Scrumia vai ei – Case Yritys X

Pasi Reini

Lokakuu 2008

JYVÄSKYLÄN
AMMATTIKORKEAKOULU

KUVAILULEHTI
Päivämäärä
01.11.2008

Tekijä(t) Pasi Reini	Julkaisun laji Opinnäytetyö	
	Sivumäärä 63	Julkaisun kieli Suomi
	Luottamuksellisuus ☐ Salainen _____saakka	
Työn nimi Scrummia vai ei – Case Yritys x?		
Koulutusohjelma Liiketalous / Tietojenkäsittely		
Työn ohjaaja(t) Niko Kiviaho		
Toimeksiantaja(t) Ei saa julkaista tutkimuksen kohde organisaation nimeä tässä opinnäytetyössä.		

Tiivistelmä

Erilaisista ohjelmiston kehitysmenetelmistä on tullut merkittävä ohjelmistojen laatua ja ohjelmistokehityksen tuottavuutta parantava tekijä. Perinteisiin suunnittelumenetelmiin kuuluu vesiputousmalli. Viime aikoina ohjelmistonkehityksessä huomiota on saanut agile Software Development (ketterä kehitys), jonka eräitä metodeja ovat XP (Extreme Programming) ja Scrum.

Tämän opinnäytetyön tutkimuksen kohteena on Scrum-menetelmä, joka raportoidaan yksityiskohtaisesti lukijalle. Tässä opinnäytetyössä käsitellään myös ohjelmistotuotannon ja ohjelmistoprojektin käsitteitä. Tiedonhankintamenetelminä käytetään perehtymistä ohjelmistotuotannon lähdeteoksiin ja artikkeleihin.

Toteutettiin tutkimus informaatioteknologiaorganisaatioon. Tutkimuksen kohderyhmänä oli yrityksen sovelluskehitystiimi. Tutkimusmenetelmänä käytetään kyselyä. Tutkitaan yrityksen tuotekehitysprosessien dokumentteja. Viimeinen vaihe tutkimuksessa oli haastatella sovelluskehitystiimin johtajaa.

Yhtenä tuloksena raportoitiin yrityksen tuotekehitysprosessi. Raportoitiin erot yrityksen tuotekehitysprosessin ja Scrumin väliltä. Mitä organisaation tulisi muuttaa tuotekehitysprosessissaan, jotta se olisi Scrumin mukainen. Viimeiseksi neuvoteltiin sovelluskehitystiimin johtajan kanssa, onko Scrum sopiva menetelmä yritykselle.

Tutkimuksen kohde organisaatio voi käyttää tämän opinnäytetyön tuloksia tutkimalla miten Scrumia pitäisi muuttaa, jotta se sopisi organisaation ohjelmistotuotekehitykseen. Tuloksia ei voida yleistää, koska kyseessä on tietyn organisaation toiminnan tutkiminen.

Avainsanat (asiasanat)
Ohjelmistotuotanto, ohjelmistoprojekti, Agile, Scrum

Muut tiedot

JYVÄSKYLÄ UNIVERSITY OF APPLIED SCIENCES

DESCRIPTION
Date
01.11.2008

Author(s)
Pasi Reini

Type of Publication
Bachelor´s Thesis

Pages
63

Language
Finnish

Confidential
☐ Until _____

Title
SCRUM OR NO SCRUM–CASE COMPANY X?

Degree Programme
Business Information Systems

Tutor(s)
Niko Kiviaho, Lecturer

Assigned by
No permission to publish the name of the assigner company.

Abstract

Many software development methods have begun to play an important role in the software projects for making better quality software and for increasing productivity in the software development. A traditional software planning method is the waterfall model. Lately, the agile software development has caused a great deal of new attention in the software development. Agile development methods are, for example, Scrum and Extreme Programming.

In this thesis, the main research objective is Scrum method, which is reported in a very detailed manner for the reader. The concepts of the software development and the software project are also reported. Information of these concepts can be found in the source articles and software development literature.

In this thesis, research into specified information technology organization was carried out. The target group of the research was the software development team of the company. The research method was a questionnaire and also the software development process documents of the IT company in question. The last phase of the research was an interview with the software development team leader.

The first result of this thesis was a report of the software development process. The second result was a discussion on differences between Scrum method and the software development process of the organization. The third and last result was to analyze with the technical leader if Scrum is a correct method for the software development process for this organization.

The assigner company can use the results of this thesis in the future to research what the organization must change in Scrum in order for it to be the right method for their software development process. The results of this thesis cannot be universal, because the research was a case study into a specified IT company.

Keywords
Software development, software project, Agile, Scrum

Miscellaneous

Scrumia vai ei – Case Yritys X

Pasi Reini

Lokakuu 2008

SISÄLTÖ

1 JOHDANTO 5

2 TUTKIMUSASETELMA 6

2.1 Tutkimuksen kohde ja kohderyhmä 6

2.1.1 Tutkimuksen kohde 6

2.1.2 Tutkimuksen kohderyhmä 6

2.2 Tutkimuksen tavoite 7

2.3 Tutkimusmenetelmä 7

2.3.1 Tutkimuksen luotettavuus 8

2.4 Tutkimuskysymykset 8

3 OHJELMISTOTUOTANTO JA OHJELMISTOPROJEKTI 9

3.1 Laatujärjestelmä 10

3.2 Ohjelmistojen kehittäminen projektityönä 11

3.2.1 Miksi projektia tehdään – Business Case 11

3.3 Projektinhallinta 12

3.4 Dokumentointi 13

3.5 Vaihejakomallit ja projektin elinkaari 14

3.5.1 Vesiputousmalli 14

3.5.2 Protoilumalli työtapana 16

3.6 Muutostenhallinta 17

3.7 Projektin sisäinen tiedottaminen ja kokoukset 18

3.8 Standardit ja käytännöt projektissa 19

3.9 Kuri ja vapaus projektissa 19

3.10 Riskienhallinta projektissa 20

3.11 Projektiorganisaatio 21

4 AGILE OHJELMISTOKEHITYKSESSÄ 23

4.1 Mitä Agile tarkoittaa? 23

4.2 Agilesta jalostuneet metodit 23

4.3 Agile manifesti 24

4.4 Agile manifestin neljä pääperiaatetta 24

4.4.1 Periaate 1 24

4.4.2 Periaate 2 25

4.4.3 Periaate 3 25

4.4.4 Periaate 4 26

4.5 Agile ohjelmistokehityksen muut periaatteet 26

4.5.1 Lista muista periaatteista 26

4.6 Ketterä projektiryhmä - Agile 27

4.7 Pieniä julkaisuja kerrallaan – Agile 28

5 SCRUM 30

5.1 Mikä Scrum on? 30

5.1.1 Tunnusmerkit, milloin Scrum ei ole käytössä ohjelmistoprojektissa? 31

5.2 Scrum - termejä 32

5.3 Scrum ja sen kolme seremoniaa 33

5.3.1 Sprint Planning Meeting 34

5.3.2 Daily Scrum Meeting 35

5.3.3 Sprint Review Meeting 35

5.3.4 Johtajat ajan tasalla 36

5.4 Scrum-roolit 37

5.4.1 Scrum master 37

5.4.2 Product owner 37

5.4.3 Scrum team 38

5.5. SCRUMin arvot ja ideologia 38

6 TUTKIMUKSEN TOTEUTTAMINEN 40

6.1 Tutkimusaineiston kerääminen 40

6.3 Kyselylomakkeen kysymykset 41

7 TUTKIMUKSEN TULOKSET 44

7.1 Tulosten analysointi tutkimusongelmittain 44

7.1.1 Miten tutkimuksen kohdeorganisaation tuotekehitysprosessi etenee? 44

7.1.1.1 Kohdeorganisaation tuotekehitysprosessi: toteutuksen suunnittelu 44
7.1.1.2 Kohdeorganisaation tuotekehitysprosessi: sovelluksien toteutus..... 47
7.1.1.3 Kohdeorganisaation tuotekehitysprosessi: asennuspaketin tekeminen 50
7.1.1.4 Kohdeorganisaation tuotekehitysprosessi: sovelluksien asennusprosessi 52

7.1.2 Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi? 55

7.1.3 Onko Scrum sopiva menetelmänä kohdeorganisaation projektiryhmän tuotekehitykseen? 57

8 POHDINTA 60

LÄHTEET 62

KUVIOT

KUVIO 1. Ohjelmistotuotannon osa-alueet……...9
KUVIO 2. Esimerkki vesiputousmallista…….14
KUVIO 3. Projektin sidosryhmiä…….22
KUVIO 4. Iteratiivinen ohjelmistonkehitys…….29
KUVIO 5. Esimerkkikaavio Scrum tuotekehitysprosessista……...33
KUVIO 6. Scrumin mukainen tuotekehitysprosessi……34
KUVIO 7. Scrum mahdollistaa valvonnan……..…36
KUVIO 8. Kaavio valmiista työtehtävistä sprintin aikana…….....37
KUVIO 9. Toteutuksen suunnittelu…….....46
KUVIO 10. Sovelluksien toteutus……49
KUVIO 11. Asennuspaketin tekeminen……...51
KUVIO 12. Sovelluksien asennusprosessi……...54

1 JOHDANTO

Erilaisista ohjelmiston kehitysmenetelmistä ja ohjelmistoarkkitehtuureista on tullut merkittävä ohjelmistojen laatua ja ohjelmistokehityksen tuottavuutta parantava tekijä. Perinteisiin suunnittelumenetelmiin kuuluu vesiputousmalli, jota käytetään ohjelmistoalan yrityksissä paljon. Muutamien viime vuosien aikana ohjelmistonkehityksessä huomiota on alkanut herättää myös uusi menetelmä nimeltään Agile Software Development (Ketterä kehitys). Ketterien menetelmien eräitä metodeja ovat XP (Extreme Programming) ja Scrum.

Tämän opinnäytetyön tutkimuksen kohteena on Scrum-menetelmä. Tarkoituksena on luoda yleiskäsitys siitä, millainen menetelmä on kyseessä. Tässä opinnäytetyössä käsitellään myös ohjelmistotuotannon ja ohjelmistoprojektin osa-alueita. Tiedonhankintamenetelminä käytetään perehtymistä lähdeteoksiin ja artikkeleihin.

Suunnitellaan ja toteutetaan tutkimus informaatioteknologia-alan organisaatioon. Tutkimuksen kohderyhmänä on organisaation sovelluskehitystiimi, jolle suoritetaan kyselytutkimus. Tutkitaan kyseisen organisaation tuotekehitysprosessidokumentteja, joista selviää tutkimuksen kohdeorganisaation tuotekehitysprosessi.

Tämän opinnäytetyön yhtenä tuloksena raportoidaan tutkimuksen kohdeorganisaation tuotekehitysprosessi. Raportoidaan myös erot kyseisen organisaation tuotekehitysprosessin ja Scrumin välillä. Mitä organisaation tulisi muuttaa tuotekehitysprosessissaan, että se olisi Scrumin mukainen. Lopuksi selvitetään, onko Scrum sopiva kohdeorganisaation tuotekehitykseen.

2 TUTKIMUSASETELMA

Tutkimus suunnataan informaatioteknologia-alan organisaatioon, jonka toimintaa tutkitaan tuotekehityksessä. Verrataan kyseisen organisaation tuotekehitysprosessia Scrumin mukaiseen ohjelmistokehitysmenetelmään. Tutkimuksen kohdeorganisaatio haluaa tietää, onko Scrumi sopiva menetelmä organisaation tuotekehitykseen.

2.1 Tutkimuksen kohde ja kohderyhmä

2.1.1 Tutkimuksen kohde

Yritysanalyysin kohteena on pieni suomalainen ohjelmistoalan yritys, jossa työntekijöinä on reilut 20 henkilöä. Yritys on perustettu 2000 luvun taitteessa ja sen koko on kasvanut tasaista tahtia työntekijöiden lukumäärässä mitattuna yhdellä tai kahdella henkilöllä vuodessa. Liiketoiminta on suunnattu energia- ja telesektorille ja yrityksen tarkoituksena on tuottaa palveluja asiakkailleen, jotka hyödyntävät näitä omassa liiketoiminnassaan.

Opinnäytetyössä ei saa julkaista yrityksen nimeä tai muita yhteystietoja. Tästä syystä opinnäytetyössä käytetään tutkimuksen kohteena olevasta yrityksestä nimitystä ”Yritysx” tai ”tutkimuksen kohdeorganisaatio”.

2.1.2 Tutkimuksen kohderyhmä

Tutkimus suunnataan kohdeyrityksen tietojärjestelmäprojekteissa tuotekehityksen parissa työskenteleville ihmisille. Tutkimukseen osallistuu 10 henkilöä, jotka ovat Yritysx:n palveluksessa olevia ohjelmistosuunnittelijoita, tuotepäälliköitä ja testaajia. Tutkimus halutaan nimenomaan kohdistaa kohdeyrityksen tuotekehityksen projektiryhmälle, koska opinnäytetyössä halutaan tutkia tämän ryhmän toimintaa tuotekehitysprosessissa. Opinnäytetyössä ei selvitetä, miten myyntimiehet suorittavat omaa myyntiprosessiaan asiakkaiden kanssa tai kuinka kirjanpitäjät kirjaavat tositteita.

2.2 Tutkimuksen tavoite

Tutkimukselle aloituspotkun antoi Yritysxn sovellustiimi, jolla on tarve kehittää omaa tuotekehitysprosessiaan. Kehitysmenetelmistä kiinnostavin oli Scrum, joka on tässä opinnäytetyössä tutkimuksen kohteena. Aiheen laajuutta rajattaessa muodostettiin johtopäätös, että ei ole järkevää käsitellä tässä opinnäytetyössä kaikkia mahdollisia ohjelmistotuotannon osa-alueita, projektin hallinnan menetelmiä, prosesseja, dokumenttien mallinnustapoja ja ohjelmointikieliä. Tässä tutkimuksessa tuodaan esille yksi näkökulma tutkittavaan aiheeseen.

Opinnäytetyön sisältöä rajataan niin, että tietoperusta osassa käsitellään ohjelmistotuotannon yleisiä osa-alueita ja ohjelmistoprojektiin liittyviä asioita. Tämän opinnäytetyön lukija tutustuu myös ohjelmistotuotantoa viime aikoina paljon puhuttaneeseen Agile käsitteeseen ja sen sisältämiin periaatteisiin. Lisäksi lukija tutustuu Agilesta periytyneen Scrum ohjelmistojen kehitysmenetelmän piirteisiin.

Opinnäytetyön seuraava vaihe on suorittaa tutkimus Yritysxn työskentelyyn ohjelmistotuotteen kehityksessä. Tutkimuksessa tarkastellaan heidän tuotekehitysprosessiaan ja sitä, miten se käytännössä tapahtuu. Tutkimuksessa pyritään selvittämään, mitä muutoksia tähän prosessiin pitää tehdä, että se noudattaa Scrum-menetelmää. Lopuksi analysoidaan, sopiiko Scrum menetelmänä kohdeorganisaation projektiryhmälle.

2.3 Tutkimusmenetelmä

Tutkimusmenetelmän valintaan on syytä panostaa, koska tämän opinnäytetyön tutkimus on saatava nopeasti valmiiksi. Tällöin on järkevää välttää tilanteita, joissa aineiston hankintaan kuluu suunnattomasti aikaa. Tästä syystä opinnäytetyöhön valittiin tutkimusmenetelmäksi kysely, koska sen avulla aineisto on nopeasti kerättävissä. Kysely voidaan myös lähettää yhtä aikaa monelle. Kun lomake on huolellisesti suunniteltu, kerätty aineisto voidaan nopeasti laatia tallennettavaan muotoon. Aineiston käsittelyyn on myös laadittu tilastolliset analyysitavat ja raportointimuodot, joten tutkijan ei tarvitse kehittää omia. (Hirsjärvi, Remes & Sajavaara 2007, 177–199.)

Tutkimuksen tarkoituksena on saada täsmällisiä vastauksia Yritys X:n tietojärjestelmäprojektin tuotekehitys-prosessista ja kerätä aineistoa tämän prosessin aikana tapahtuvasta projektiryhmän toiminnasta ja verrata saatuja vastauksia Scrum-menetelmän ominaispiirteisiin. Tällöin joudutaan keräämään tietoa tosiasioista, käsitteistä ja muodostamaan suoria kysymyksiä lomakkeelle Scrumin ominaispiirteistä. Täsmällisiä tosiasioitahan pitää kysyä suoraan yksinkertaisilla kysymyksillä, joko avointen kysymysten avulla tai monivalintatyyppisesti. (Hirsjärvi, Remes & Sajavaara 2007, 177–199.)

2.3.1 Tutkimuksen luotettavuus

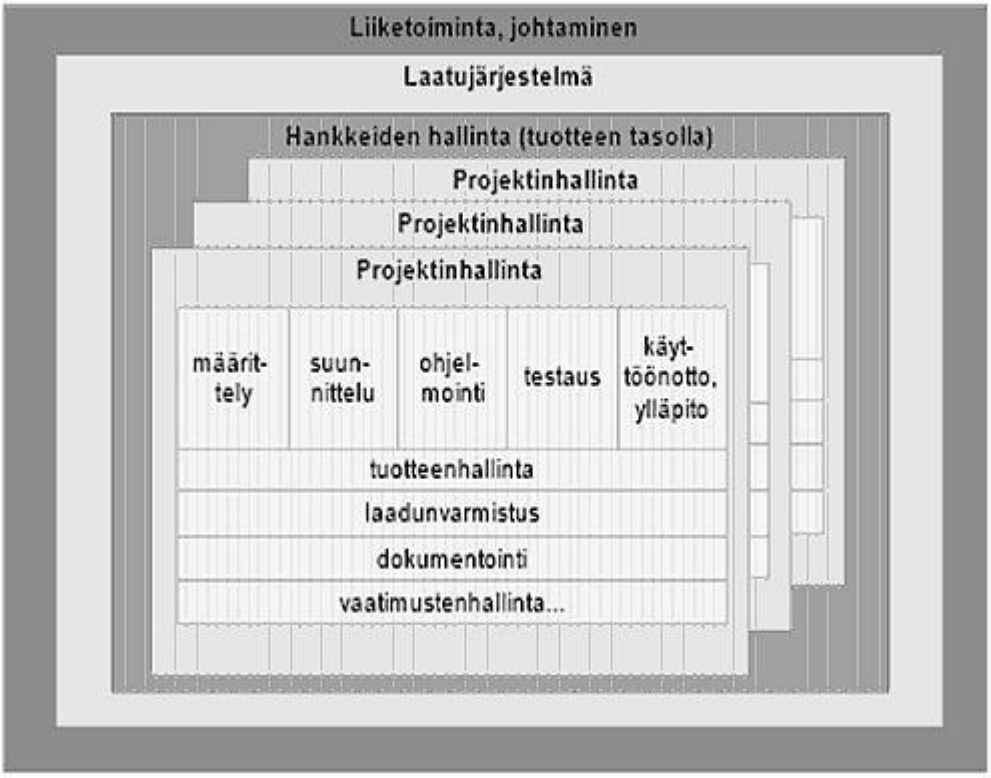
Kyselyllä kerättyyn aineistoon liittyy myös heikkouksia. Aineistoa saatetaan pitää pinnallisena ja tutkimuksia teoreettisesti vaatimattomina. Kyselyä on kritisoitu metodina: Kyselytutkimuksessa ei voida olla varmoja siitä, ovatko vastaajat suhtautuneet kyselyyn vakavissaan ja ovatko he pyrkineet vastaamaan rehellisesti. Ei voida olla varmoja, ovatko vastaajat perehtyneet aiheeseen aiemmin. Kerättävän aineiston kato voi myös nousta suureksi joissakin tapauksissa. Hyvän kyselylomakkeen laatiminenkin vaatii aikaa ja taitoa tutkijalta. (Hirsjärvi, Remes & Sajavaara 2007, 190.)

2.4 Tutkimuskysymykset

Tämän opinnäytetyön osalta nostetaan esille seuraavat tutkimusongelmat: Miten tutkimuksen kohdeorganisaation tuotekehitysprosessi etenee? Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi? Onko Scrum sopiva menetelmänä kohdeorganisaation projektiryhmän tuotekehitykseen?

3 OHJELMISTOTUOTANTO JA OHJELMISTO-PROJEKTI

Ohjelmistotuotanto voidaan jakaa osa-alueisiin kuviossa yksi esitetyllä tavalla (ks. kuvio 1). Tuotantoa ohjaa organisaation laatujärjestelmä, joka määrittelee kyseisen organisaation toimintatavat ohjelmistonkehityksessä. Ohjelmistojen kehittäminen tapahtuu projekteissa. Toisinaan projektit saattavat olla osahankkeita laajemmasta projektista. Kehitysprosessista voidaan erottaa ainakin määrittely, suunnittelu, ohjelmointi, testaus ja käyttöönotto sekä ylläpito. Taustalla ovat organisaation johtaminen ja liiketoiminta. (Haikala & Märijärvi 2004, 35.)



KUVIO 1. Ohjelmistotuotannon osa-alueet. (Haikala & Märijärvi 2004, 35.)

3.1 Laatujärjestelmä

Ohjelmiston laadulla tarkoitetaan sitä, että ohjelmistotuote täyttää loppukäyttäjän kohtuulliset toiveet ja odotukset. Laatua voidaan tarkastella sekä toiminnan että lopullisen tuotteen näkökulmasta. Nykyään ohjelmistoalalla on suuntauksena, että panostetaan toiminnan laatuun, koska sillä uskotaan olevan positiivisia vaikutuksia lopulliseen ohjelmistotuotteeseen. Ohjelmistotuotteen tekemisessä käytettäviä menetelmiä ja käytäntöjä, yrityksen toimintatapaa, kutsutaan laatujärjestelmäksi. Laatujärjestelmän tarkoituksena on varmistaa, että tuotantoprosessi tuottaa suunnitelman mukaisia ohjelmistotuotteita aikataulussa ja budjetin mukaan. Laatujärjestelmää kuvaa yleensä organisaation laatukäsikirja sekä tähän liittyvät muut dokumentit ja ohjeistukset. (Haikala & Märijärvi 2004, 48-49.)

Käsitteenä laatu on monelle kirosana, koska sen nimissä on tehty vääriä asioita. Monet organisaatiot ovatkin lopettaneet ohjelmistojen laadusta mainostamisen ja laatupäällikkö vaihtunut riskienhallintapäälliköksi. Edellä mainittu tapaus ei ole hyvä tapa toimia, koska laatu käsitteenä on varsin tärkeä ohjelmistoja valmistavalle yritykselle. Timo Lehtimäki (2006, 66-67.) esittää kirjassaan, että laatu voidaan jakaa kolmiportaiseen organisaation laadun kypsyysmalliin seuraavasti:

0. Ei hajuakaan laadusta, toimitaan miten sattuu.

1. Laatuorganisaatio, laatupäälliköt, laatukäsikirja (ISO 9000).

2. Laatu integroitu kaikkeen toimintaan.

Päästäkseen tasolle kaksi tässä kolmiportaisessa kypsyysmallissa, yrityksen on ensin käytävä tasolla yksi. Mistä sitten tietää, että yritys on mahdollisesti tasolla 2. Timo Lehtimäki (2006, 67.) kertoo kirjassaan, että kannattaa tarkastella yrityksen historiaa. Jos sieltä löytyy kokemuksia tasolta yksi, yritys voi olla tasolla kaksi. (Lehtimäki 2006, 66–69.)

3.2 Ohjelmistojen kehittäminen projektityönä

Ohjelmistojen kehittäminen tapahtuu yleisimmin projektityyppisenä työskentelynä. Monesti projekti saatetaan jakaa useampiin eri projekteihin, kun kyseessä on laaja ja pitkä projekti. Yleinen tapa on jakaa projekti osiin siten, että eriytetään määrittely- ja toteutusvaihe omiksi projekteikseen. Tämä sen takia, että on hankala toteuttaa sellaista, mitä ei ole määritelty. Määrittelyprojektin tuloksena voi myös olla, että projekti keskeytetään kannattamattomana. (Haikala & Märijärvi 2004, 53–54.)

Projektin suunnitteluvaiheessa projekti jaetaan erillisiin päävaiheisiin. Tämän jälkeen vielä kukin vaihe paloitellaan pienempiin tehtäviin, joille asetetaan työmääräarviot. Seuraavaksi tehtävät sijoitetaan kalenteriin ja niille valitaan tehtävien suorittavat henkilöt. Kun tehtävät osataan jakaa pieniin palasiin ja lyhyisiin jaksoihin, tulee projektisuunnitelmasta ja tehtävien seurannasta luotettavampaa. Projektin hyvä kesto on yksi kalenterivuosi. (Haikala & Märijärvi 2004, 53-54.)

3.2.1 Miksi projektia tehdään – Business Case

Business case nähdään monesti kammottavana asiana, mutta se on edellytys ja syy, miksi projekteja tehdään. Ei voi olla muita syitä tehdä projekteja, koska projektin lopputuloksista täytyy olla tuloksien vastaanottajalle ja projektin tekijälle hyötyä. Business case käsitteellä tarkoitetaan ”kustannus-hyöty-analyysia”. Siinä arvioidaan ja lasketaan, kuinka paljon projekti tulee maksamaan ja paljonko siitä saadaan hyötyä. Vaikuttavin business case on käännetty suoraan rahaksi. Näin saadaan suoraan tietoon, kauanko projektiin investoidun rahan saaminen takaisin kestää ja paljonko se tuottaa voittoa. (Lehtimäki 2006, 7–9.)

Monesti business caset ovat rumia ja niissä on analysoitu, miten projektin lopputulos tarjoaa tehokkuutta projektin tulokset vastaanottavalle organisaatiolle. Usein taustalla on eriteltynä henkilötyön väheneminen ja tästä johtuen saattaa tulla irtisanomisia henkilöstölle. Projektipäällikölle business casen tarjoama tieto on myös tärkeää, jotta hän voi ohjata projektia oikein ja ymmärtää liiketoiminnan ihmisiä paremmin. Oikeastaan on mahdotonta ohjata projektia, jos ei tiedetä business casea. Monesti projektipäällikön tultua mukaan projektiin, on joku jo määritellyt business casen, mutta sitä ei ole

saatavilla. Tällöin projektipäällikön täytyy itse kaivaa esille tämä tieto. (Lehtimäki 2006, 7–9.)

3.3 Projektinhallinta

Projektinhallinta voidaan jakaa seuraaviin osa-alueisiin: Projektin suunnittelu: Lähtökohtina projektinsuunnittelulle ovat projektin tavoitteet ja reunaehdot. Tavoitteet tulee määritellä projektisuunnitelmassa sellaiseen muotoon, että niiden arviointi projektin päätyttyä on mahdollista. (Haikala & Märijärvi 2004, 226–228.)

Projektin suunnittelussa on otettava huomioon, että ei saa unohtaa suunnitella itse projektia. Monesti kokematon projektipäällikkö käyttää kaiken aikansa projektisuunnitelman tekemiseen, mikä on tärkeää. Timo Lehtimäen (2006, 14.) mielestä projektisuunnittelussa on otettava huomioon neljä tärkeintä ydinkohtaa:

- Mitkä ovat projektin tehtävät?
- Kuka nämä tehtävät tekee?
- Milloin tehtävät tehdään?
- Mitä tehtävistä syntyy tuloksina?

Jos edellä mainitut kohdat ovat selvillä, projekti on suunniteltu. Jos joku näistä kohdista ei ole selvillä, dokumentoituna ja hyväksyttynä, projektia ei ole vielä suunniteltu, vaikka projektisuunnitelma olisikin olemassa. (Lehtimäki 2006, 14.)

Projektin käynnistäminen on vain pieni osa projektia, mutta sen merkitystä ei voi korostaa liikaa. Monet projektit käynnistyvät vaivihkaa ja monesti henkilöt, jotka osallistuvat projektiin, eivät tiedä, että ovat siinä mukana. Yleensä projekti käynnistetään tilaisuudella, jossa käydään projekti tarkasti läpi siihen osallistujien kanssa. Tilaisuudessa esitellään projektin tavoitteet, projektin tärkeys, osallistujat ja sidosryhmät, aikataulu ja muut käytännöt, joita projektissa mahdollisesti on. (Haikala & Märijärvi 2004, 226-228.)

Toteutuksen seuraaminen ja ohjaaminen: Projektin toteutuksen seuraamista edistävät projektipalaverit, projektikatselmukset, toteutumien kirjaaminen ja projektisuunnitelman tarkentaminen projektin edetessä. Näin pyritään estämään projektin lähteminen väärille raiteille. (Haikala & Märijärvi 2004, 226–228.)

Projektin päättämisen yhteydessä siivotaan projektin dokumentaatio ja tiedostot. Kaikki tarpeeton tuhotaan ja tarpeellinen arkistoidaan. Jos projekti avataan myöhemmin uudelleen, on dokumentaatio jo olemassa. Projektista kannattaa myös tehdä loppuraportti, joka sisältää projektin lopputuloksen, onnistumiset ja epäonnistumiset ja arvion, miten projektia kannattaa jatkossa kehittää. Projekti kannattaa myös päättää selkeästi, jotta sen vastuu siirtyy projektin ylläpitäjälle ja projektissa mukana olevat tietävät sen päättyneen. (Haikala & Märijärvi 2004, 226-228.)

3.4 Dokumentointi

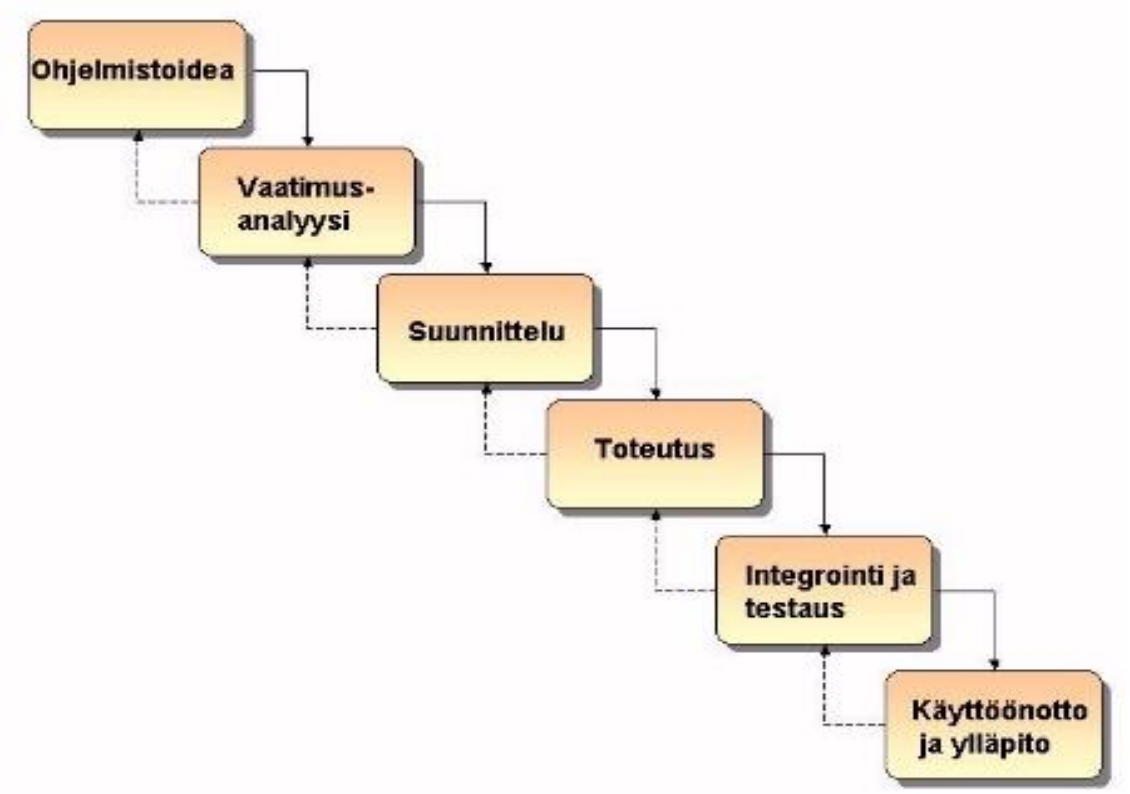
Ohjelmistojen valmistuksessa on tyypillistä, että ohjelman kehityksen aikana kertyneitä tietoja kirjataan erilaisiin dokumentteihin. Dokumentoinnin tulisi olla seuraavanlaista vähintään: projektisuunnitelma, määrittelydokumentti (toiminnallinen määrittely), suunnitteludokumentti (tekninen suunnittelu) ja testaussuunnitelma. Laatujärjestelmä ottaa myös kantaa, minkälaisia dokumentteja organisaation tulisi projekteissaan tuottaa. Ylläpidon näkökulmasta dokumentointi on ongelmallista, koska dokumentit eivät välttämättä ole ajan tasalla. Projektin päätyttyä dokumentaatio tulisikin saattaa helposti ylläpidettävään muotoon. (Haikala & Märijärvi 2004, 51-53.)

3.5 Vaihejakomallit ja projektin elinkaari

Ohjelmiston elinkaarella tarkoitetaan aikaa joka kuluu ohjelmiston kehittämisen alkamisesta sen poistamiseen. Projekti jaetaan eri vaiheisiin, joita ovat määrittely, suunnittelu, toteutus ja testaus. Tähän vaiheistamisen avuksi on kehitelty erilaisia malleja, kuten vesiputous- ja protoilumalli. (Haikala & Märijärvi 2004, 36-37.)

3.5.1 Vesiputousmalli

Tavallisin ohjelmistoprojekteissa käytettävä vaihejakomalli on vesiputousmalli ja siitä voidaankin erottaa seuraavat vaiheet (ks. kuvio 2). Vesiputousmallista on useita eri muunnelmia, mutta yleensä näistä voidaan erottaa ainakin määrittely-, suunnittelu- ja toteutusvaiheet. (Haikala & Märijärvi 2004, 36 – 37.)



KUVIO 2. Esimerkki vesiputousmallista. (Virtuaali AMK)

Vesiputousmallin mukaisessa ohjelmistonkehityksessä lähdetään liikkeelle esitutkimusvaiheesta, jossa määritellään järjestelmälle vaatimukset. Tämä vaihe vastaa kysymykseen, minkälainen ohjelma tehdään ja miksi. Esitutkimus on projektin tärkeimpiä

ja haastavampia vaiheita, koska vääristä vaatimuksista ei voida päätyä hyvään järjestelmään. (Haikala & Märijärvi 2004, 37.)

Määrittelyvaihe, joka tunnetaan myös vaatimusanalyysina tai vaatimusmäärittelynä, seuraa esitutkimusvaihetta. Tässä vaiheessa analysoidaan esitutkimusvaiheessa asetettuja vaatimuksia järjestelmälle, jolloin niistä syntyvät ohjelmistovaatimukset, jotka määrittelevät toteutettavan järjestelmän. Ohjelmistovaatimuksista käytetään myös muita termejä, kuten järjestelmävaatimukset, toiminnalliset vaatimukset ja ominaisuudet. Määrittelyvaiheesta syntynyttä dokumenttia kutsutaan toiminnalliseksi määrittelyksi. (Haikala & Märijärvi 2004, 38–39.)

Toiminnallisessa määrittelyssä kuvataan järjestelmän toiminnot eli toisin sanoen operaatiot. Ohjelmalle asetetaan myös ei-toiminnallisia vaatimuksia ja rajoituksia. Ei-toiminnallisia vaatimuksia ovat suoritusteho, vasteaika ja käytettävyys. Rajoituksia ovat käytettävissä oleva muistitila ja toteutus tietyllä ohjelmointikielellä. Tässä osiossa määritellään myös järjestelmän käyttöliittymä ja sovelluksen kommunikointi muiden järjestelmien kanssa. (Haikala & Märijärvi 2004, 38–39.)

Määrittelyvaiheen jälkeen hypätään itse toteutukseen eli ohjelmointivaiheeseen, jossa kirjoitetaan ohjelman lähdekoodi. Toteutuksen jälkeen on testausvaihe, jossa yritetään löytää koodatusta ohjelmasta virheitä. Testaus tehdään monella eri tasolla, kuten moduuli-, järjestelmän integraatio- ja järjestelmätasolla. Moduulitestauksessa etsitään vikoja yksittäisistä moduuleista, integraatiotestauksessa vikoja etsitään moduulien yhteistoiminnasta ja järjestelmätestauksessa koko järjestelmän toiminnasta ja suorituskyvystä. (Haikala & Jukka Märijärvi 2004, 40.)

Testaus on oma osaamisalueensa. Projektipäällikön täytyy ymmärtää testaamisen merkitys, koska jokaisessa projektissa testataan ja sen aiheuttama työmäärä on suuri. Huono testaaminen tai sen laiminlyönti saattavat aiheuttaa projektin lopputuloksessa ikäviä yllätyksiä. Projektin alkuvaiheessa on hyvä suunnitella strategia testaamiselle tai ainakin määritellä projektin eri vaiheille omat testaukset ja niille tavoitteet. Yleensä moduulitestauksesta huolehtivat moduulien koodanneet henkilöt, mutta laajemman kokonaisuuden testaamisen suunnitteluun ja toteuttamiseen tarvitaan siihen erikoistuneita ammattiosaajia. (Lehtimäki 2006, 170.)

Testausta voidaan lähteä viemään läpi siten, että suunnitellaan projektin alussa systeemitesti. Se tarkoittaa, että testitapaukset dokumentoidaan ja kuvataan tarkasti, miten järjestelmän tulee reagoida mihinkin syötteeseen. Kun nämä on suunniteltu ja kuvattu hyvin, tiedetään jälkeenpäin, mitä on testattu, ja testaaminen vaatii vähemmän asiantuntemusta. Jos edellä mainittuja testitapauksia ei suunnitella kunnolla, menee testaaminen helposti kokeilemiseksi ja sen laajuutta on vaikea seurata. Testitapausten suunnittelun jälkeen on vielä hyvä arvioida, montako virhettä mahdollisesti ohjelmasta löytyy testejä ajettaessa. Jotta virhemäärä osataan arvioida oikein, tulee kerätä tiedot aiempien projektien laajuudesta ja verrata testiajojen pituuksia niissä ilmenneiden virheiden määriin. (Lehtimäki 2006, 170.)

Ylläpito on sovelluksen käytössä ilmenneiden ongelmien ratkomista, virheiden korjaamista ja uusien ominaisuuksien lisäämistä. Ylläpito voidaan jakaa ainakin seuraaviin osa-alueisiin: korjaava ylläpito, adaptiivinen ylläpito ja täydentävä ylläpito. Korjaavassa ylläpidossa suoritetaan ohjelmatuotteeseen korjaavia toimenpiteitä havaittuihin virheisiin. Adaptiivisessa ylläpidossa tehdään muutoksia muuttuneiden vaatimusten osalta. Täydentävässä ylläpidossa lisätään ohjelmaan uusia ominaisuuksia. (Haikala & Märijärvi 2004, 41.)

3.5.2 Protoilumalli työtapana

Protoilutyyppisellä lähestymistavalla tarkoitetaan, että on olemassa sellainen työskentelymalli, jossa valmistettavaa tuotetta ja sen piirteitä kokeillaan ennen varsinaisen tuotteen valmistamista. Prototyypit soveltuvat silloin, kun ollaan tekemisissä uuden teknisen ratkaisun kanssa, jolloin on välttämätöntä tehdä kokeiluja ennen päätöstä siirtyä käyttämään uutta ratkaisua. Ollaan myös tekemisissä epäselvien asiakasvaatimusten kanssa. Tässä vaihejakomallissa on kaksi käyttövaihtoehtoa: siirrytään prototyypin valmistuttua sen perusteella määrittelemään uutta järjestelmää, joka toteutetaan alusta alkaen uudestaan, tai valmis prototyyppi kehitetään valmiiksi tuotteeksi. Prototyyppinen työtapa on osoittautunut hyödylliseksi käyttöliittymiä määriteltäessä ja tehtäessä. Ongelmia prototyypin kanssa syntyy, kun sitä parannellaan ja hiotaan loputtomasti. Sen takia prototyypistä kannattaa tehdä varsin yksinkertainen ja viimeistelemättömän näköinen. Tällöin ohjelman käyttäjät eivät luule ohjelmaa valmiiksi, vaikka siitä on vielä suurin osa tekemättäkin. (Haikala & Jukka Märijärvi 2004, 42-43.)

3.6 Muutostenhallinta

Projekteissa tulee muutoksia. Tästä syystä jokaisen projektipäällikön tulee asennoitua projektissaan niin, että suunnitelmat saattavat vaihtua. Ainahan ei voida ennalta tehdä täydellisiä määrityksiä ja suunnitelmia. Näihin muuttuviin tapahtumiin ja tilateisiin vastaa käsite muutostenhallinta, jolla tarkoitetaan tässä luvussa projektin laajuudenhallintaa. Pahimmassa tapauksessa projekti lähtee rönsyilemään hallitsemattomasti, jos projektin laajuus on epäselvästi rajattu. Tästä päästään siihen, että tähän pisteeseen johtanut projekti yleensä epäonnistuu. Eikä tällaista projektia voida enää edes kutsua projektiksi, koska projektin tunnusmerkkeihin kuuluu se, että projektin laajuus on määritelty, sitä tehdään rajallisessa ajassa ja rajallisin resurssein. Hyvä ja toimiva muutostenhallinta vaatii kunnolla kuvatun projektin laajuuden, koska muutoin projektin päälle on vaikea rakentaa toimivaa muutostenhallintaa. (Lehtimäki 2006, 47-50.)

Projektin muutostenhallinta lähtee siitä periaatteesta, että projektiryhmä ja kaikki projektin kanssa tekemisissä olevat henkilöt tuntevat projektin laajuuden ja ymmärtävät sen. Lisäksi heidän tulee ymmärtää, että projektin laajuutta ei saa päästää leviämään. Tämän takia kannattaa määritellä prosessi muutostenhallinnalle, jota kaikki projektin tahot noudattavat. Tämän prosessin kuvaus kannattaa liittää projektisuunnitelmaan niin, että se on kaikkien saatavilla ja myöhemmin ongelmien ilmentyessä voidaan tarvittaessa palata siihen. (Lehtimäki 2006, 48.)

Muutoksia on järkevää dokumentoida ja ylläpitää, että myöhemmin nähdään, miten tiettyjä asioita on muutettu. Muutospyyntöihin kannattaa määritellä lomake, jolla muutoksia haetaan. Tässä lomakkeessa olisi hyvä olla seuraavat tiedot:

- Muutospyynnön numero (juokseva)
- Muutospyynnön nimi
- Ehdottaja ja ehdotuspäivämäärä
- Muutoksen tarkka kuvaus
- Perustelu, miksi muutos tarvitaan
- Työmääräarvio
- Vaikutus projektin aikatauluun
- Vaikutus projektin hintaan ja laskutukseen

Projektipäällikön on hyvä pitää lokia muutoksista ja siitä, mitä niistä on päätetty. Lokissa voidaan luetella muutospyynnöt (nimi ja numero), päivämäärä ja muutospyynnön tila. Tiloja muutokselle voivat olla: uusi, käsittelyssä, hylätty ja hyväksytty. (Lehtimäki 2006, 48–49.)

3.7 Projektin sisäinen tiedottaminen ja kokoukset

”Tiedotusta ei voi olla liikaa, mutta kokouksia voi olla”. (Lehtimäki 2006: 56.)

Projektipäällikön tulee jakaa tietoa projektiryhmälle projektin tilasta ja siihen liittyvistä asioista sekä muutoksista. Kun projektin alussa järjestetään perusteelliset projektin käynnistämiset (kick-off) ja käydään projekti läpi sekä järjestetään tarpeelliset perehdytykset, tarve käskyjen jakamiselle pienenee ja projektiryhmän motivaatio paranee. On myös yleistynyt, että projektiryhmä laitetaan työskentelemään samoihin tiloihin. Tätä varten on kehitetty ”flexi-space-malli”, jossa työntekijällä ei ole vakio työpistettä, vaan työpisteet vaihtuvat sen mukaan, miten projekti elää ja kenen kanssa milloinkin tekee yhteistyötä. Tästä on etuna se, että kommunikaatio ja tiimihenki paranevat. (Lehtimäki 2006, 56–59.)

Projektiin liittyvien kokouksien kannalta täytyy miettiä, milloin on järkevää pitää kokouksia ja minkä takia. Usein näitä kokouksia on turhan paljon ja tehokkuutta ei lisää se, että ollaan pitämässä palavereja, vaan tuloksia pitää syntyä, kuten Timo Lehtimäki kirjassaan ”Ohjelmistoprojektit käytännössä” toteaa:

”Koko projektiryhmän ei tarvitse kokoontua viikoittain. Toisaalta onhan se hieno ajatus, että kaikki kuulevat, miten kaikilla menee ja mitä ongelmia kenelläkin on ollut, mutta se vaan nyt on liian tehotonta.” (Lehtimäki 2006, 58.)

Tärkeää on myös järjestää kokouksia, jotta projektin parissa työskentelevien henkilöiden tieto ja mielipiteet saadaan selville. Lisäksi kokouksen rakenne täytyy suunnitella huolellisesti etukäteen, että nämä voidaan viedä vikkelästi läpi. On myös varmistettava, että kokouksessa käsitellään oikeita asioita ja niillä on tavoite sekä esityslista (agenda), jonka mukaan kokouksessa edetään. Monesti on hyötyä, että kokouksen

materiaalit jaetaan osallistujille ajoissa ennen varsinaista kokoontumista. Tällä varmistetaan, että kokoukseen osallistujilla on tarvittava tieto etukäteen, mikä nopeuttaa asioiden käsittelyä. (Lehtimäki 2006, 58.)

3.8 Standardit ja käytännöt projektissa

”Ei hosuta, vaan sovitaan heti aluksi, miten työt tehdään.” (Lehtimäki 2006, 94.)

Projektien aluksi on hyvä sopia standardeista ja käytännöistä, miten tietyissä tilanteissa menetellään. On erityisen tärkeää, kun ohjelman lähdekoodia kirjoitetaan, että se näyttää samalta ja sitä on mahdollista ylläpitää. Ohjeet ja standardit voivat liittyä myös arkkitehtuuriratkaisuihin tai käyttöliittymän ulkoasuun. Jos asioita ei sovita etukäteen, ihmiset tekevät omanlaisiaan ratkaisuja, jolloin tuloksena on epäyhtenäisiä toteutuksia ja sähläystä. Monet saattavat ajatella, että näin kielletään ohjelmoijia ja suunnittelijoita olemasta luovia, mutta tämä ei pidä paikkaansa. Luovuudelle on runsaasti käyttöä, kun mietitään ratkaisuja. Näiden ratkaisujen on sitten vain oltava yhdenmukaisia järjestelmässä, jota ollaan toteuttamassa. (Lehtimäki 2006, 94-95.)

Standardit ja ohjeet otetaan yleensä positiivisesti vastaan, kunhan ne vain ovat järkeviä ja johdonmukaisia. Tällöin ne auttavat projektissa työskenteleviä henkilöitä toimimaan yhdenmukaisesti ja sooloilujen aiheuttama sähläys vähenee. Projektipäällikön on hyvä heti projektin alussa kerätä joukko teknisiä asiantuntijoita palaveriin, jossa hyväksytään projektin standardit. Monessa projektissa standardien luominen on jopa projektin tuloksissa tavoitteena. Standardit liittyvät ohjelmistokoodin kirjoittamisen ohella projektin dokumentaatioon, koska siinä sovitaan, minkälaisia dokumentteja projekti tuottaa ja miten niitä hallitaan. (Lehtimäki 2006, 94-96.)

3.9 Kuri ja vapaus projektissa

Tietojärjestelmäprojektissa on hyvä olla tiukka kuri, mutta sen tulee kohdistua oikeisiin asioihin, kuten ohjelmointistandardeihin, versionhallinta- ja dokumentointiohjei-

siin, raportointimenettelyihin ym. vastaaviin. Kuria ei saa kohdistaa sellaisiin asioihin, jossa sitä ei tarvita. Näitä ovat pukeutuminen, hiustyyli, työajat ja vastaavat. Työajat sen takia, että joustoa pitää löytyä, jos työt hoituvat. Projekteissa työtä tekevät ihmiset. Täytyy muistaa, että ihmisille tulee inhimillisiä tilanteita, jolloin henkilö ei voi työtä suorittaa säännöllisten työajan puitteissa. (Lehtimäki 2006, 122-123.)

3.10 Riskienhallinta projektissa

Aina projekti ei mene suunnitellulla tavalla ja jotakin odottamatonta voi tapahtua kesken projektin sen sisällä tai ulkopuolella. Tällöin on vaarana, että projekti epäonnistuu jonkin aiheutuneen riskin takia. Tämän takia projektipäällikön täytyy hallita projektin riskit ja varautua näihin mahdollisimman hyvin. Varautumista on kahdenlaista: voidaan yrittää estää riskin syntyminen tai lieventää sen haittoja. Projektin alussa on hyvä järjestää riskianalyysi. Mukaan tähän tilaisuuteen kannattaa tietenkin kutsua teknisiä-osaajia mutta myös ohjelmistotuotantoon yleisesti liittyvien asioiden asiantuntijoita, kuten henkilöitä, jotka ymmärtävät liiketoimintaa. Riskianalyysi voidaan tehdä seuraavalla tavalla:

- Ensin ideoidaan mahdolliset riskit.
- Sitten poistetaan riskit, jotka eivät ole relevantteja projektin näkökulmasta (esimerkiksi tulivuorenpurkaus Jyväskylässä).
- Jäljelle jääneet riskit analysoidaan siten, että arvioidaan niiden toteutumisaste ja vaikutukset projektille.
- Lopuksi kirjataan toimenpiteet riskin toteutumisen estämiseksi sekä sen vaikutuksen minimoimiseksi.

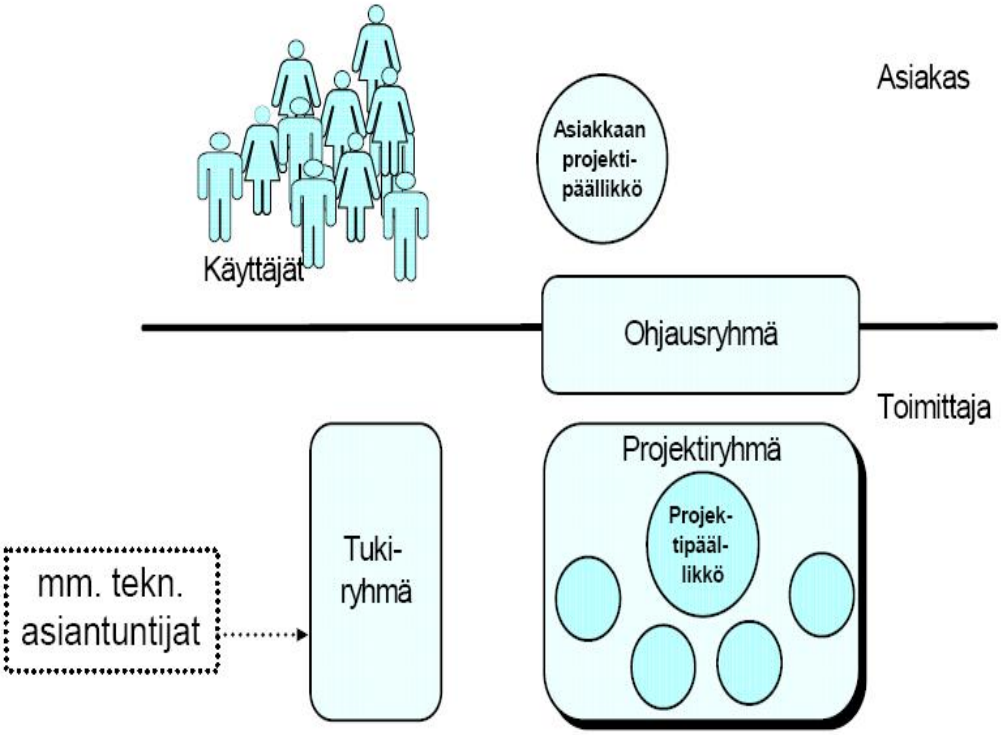
(Lehtimäki 2006, 79–81.)

Pelkkä riskianalyysi ei kuitenkaan riitä, vaan projektipäällikön on tartuttava toimeen ja valvottava, että riskianalyysissä määritetyt tarvittavat toimenpiteet tehdään riskien estämiseksi ja vahinkojen minimoimiseksi. On nimenomaan projektipäällikön vastuulla valvoa, että näin tapahtuu. Projektipäällikön on myös syytä valvoa, että projektin ohjausryhmässä tilannetta seurataan. Riskianalyysilistaa on päivitettävä; eli sieltä on

poistettava riskit, jotka ovat muodostuneet jo ongelmiksi. Ongelma, joka on jo tapahtunut, ei voi olla enää riski, joka tapahtuu tulevaisuudessa. (Lehtimäki 2006, 82.)

3.11 Projektiorganisaatio

Projekti koostuu sen jäsenistä ja projektipäälliköstä, eli projektiryhmästä. On hyvin yleistä, että sama henkilö joutuu toimimaan monen eri projektin johtajana samanaikaisesti. On myös yleistä, että projektin jäsenillä on muita projektin ulkopuolisia työtehtäviä. Lisäksi projektiryhmä on saatettu kasata siten, että Pekka työskentelee Tampereella, Matti Espoossa ja muut ryhmän jäsenet toimivat Jyväskylässä. Projektiin liittyy yleensä tukiryhmä, joka kootaan teknisistä asiantuntijoista, esimerkiksi jonkin työkalun erityisosaajia. Tukiryhmän jäsenet antavat neuvoja projektiryhmälle teknisten ratkaisujen suunnittelussa ja tarkastavat niiden toteutuskelpoisuuden. Projektilla on myös ohjausryhmä, joka koostuu ohjelman toimittajan edustajista ja lopullisen tuotteen vastaanottajien edustajista (ks. kuvio 3). Ohjausryhmän tarkoituksena on seurata projektin edistymistä ja auttaa mahdollisten ongelmien ratkaisussa. Ohjausryhmä myös hyväksyy projektisuunnitelmaan tehtävät muutokset. (Haikala & Märijärvi 2004, 228–229.)



KUVIO 3. Projektin sidosryhmiä. (Haikala & Märijärvi 2004, 229.)

4 AGILE OHJELMISTONKEHITYKSESSÄ

4.1 Mitä Agile tarkoittaa?

Ketterät ohjelmistokehitysmenetelmät (Agile Methods, Agile Modeling) ovat syntyneet vaihtoehtoliikkeen tavoin ratkaisemaan raskaiden ohjelmistokehitysmallien ongelmia. Perinteiset ohjelmistoprosessit ovat monesti raskaita dokumentaation ja kankeiden kontrollimekanismien vuoksi. Sen takia niitä onkin joskus vaikea soveltaa erilaisiin ohjelmistoprosesseihin. Menetelmiä kutsutaan ”ketteriksi”, koska ne ovat mukautumiskykyisiä nopeasti muuttuviin asiakasvaatimuksiin ja vaikeaan ennustettavuuteen toisin kuin perinteiset ohjelmistoprosessit. (Sirviö 2002.)

4.2 Agilesta jalostuneet metodit

Agile itsessään ei ole metodi, vaan kokonaisuus keskeisiä käsitteitä ja periaatteita, miten ohjelmistoja tulisi kehittää. Agilesta on periytynyt seuraavia ohjelmistonkehitysmetodeja, jotka ovat jalostuneet kasvavan tyytymättömyyden tunteen aiheuttamasta muutoksista jäykkään ja raskaaseen ohjelmistokehitykseen:

- Adaptive Software Deveplopment (ASD).
- The Crystal Methodologies.
- Dynamic Systems Development Method (DSDM).
- Extreme Programming (XP).
- Feature-Driven Development (FDD).
- Lean Software Development.
- SCRUM.

Kaikilla edellä listatuilla metodeilla on samoja piirteitä, jotka ovat periytyneet Agilen sisältämistä periaatteista ja käytännöistä. Kuitenkin jokainen näistä metodeista on omanlainen menetelmä kehittää ohjelmia. Nämä menetelmät eroavat toisistaan siten, että niissä on panostettu enemmän tiettyihin eri piirteisiin kuten hallintaan ja kommunikaatioon (SCRUM ja ASD). Lisäksi Feature-Driven Development (FDD) menetel-

mässä otetaan suoraan kantaa siihen, mitä prosesseja ohjelmistokehitystiimin tulisi noudattaa. (Schuh 2005, 3.)

4.3 Agile manifesti

Vuonna 2001 seitsemäntoista eri ketterän menetelmän edustajaa tapasivat Utahissa ja keskustelivat vaihtoehtoisista työskentelytavoista dokumentointilähtöisille ohjelmistokehitysmenetelmille. He neuvottelivat arvoista ja periaatteista koskien uudentyyppisiä ohjelmistonkehitysmenetelmiä parantaakseen ohjelmistokehitystiimien kykyä vastata muutoksiin ja työskennellä nopeasti. Näistä henkilöistä muodostui Ketterä Allianssi (engl. Agile Alliance). Ketterä Allianssi muotoili parin päivän aikana arvoista ja periaatteista neljä pääkohtaa, joita kutsutaan Ketterän Allianssin manifestiksi (engl. Manifesto for Agile Alliance). (Agile Alliance 2002.)

4.4 Agile manifestin neljä pääperiaatetta

4.4.1 Periaate 1

Arvostetaan yksilöitä ja vuorovaikutus taitoja enemmän kuin prosesseja tai työkaluja.

Ihmiset tekevät yhdessä töitä valmistaakseen ohjelmia ja Agile kehitys asettaa suuren arvon näille ammattilaisille, jotka muodostavat projektiryhmän, koska ryhmän yksilöiden taitoihin tulee luottaa. Lisäksi, että projekti onnistuisi, ryhmälle tulee valita oikeat prosessit ja työkalut, jotka sopivat heidän ympäristöönsä, teknologioihin ja projektin tavoitteisiin. Projektiryhmän projekti väärillä prosesseilla ja työkaluilla varustettuna voi epäonnistua. (Schuh 2005, 3.)

4.4.2 Periaate 2

Arvostetaan toimivaa ohjelmaa enemmän kuin perusteellista dokumentaatiota.

Kehitettävän ohjelman on tarkoitus antaa käyttäjälleen työn suorittamiseen lisäarvoa. Agile-kehityksen mielestä ei ole hyötyä hyvin suunnitelluista yksityiskohtaisista vaatimuksista, arkkitehtuurista ym. suunnitteludokumenteista, jos niitä ei pystytä yhdistämään toimivaan järjestelmään. Toimivan järjestelmän tulisi olla projektin toiminnan päätarkoituksena. Lisäksi liiallinen dokumentointi projektin aikaisessa vaiheessa voi olla hukkaan heitettyä aikaa, jos sitä ei ylläpidetä myöhemmin. Jos dokumentoinnista ja sen ylläpitämisestä tulee toiminnan päätarkoitus, voi se aiheuttaa projektin myöhästymisen. (Schuh 2005, 3-4.)

4.4.3 Periaate 3

Arvostetaan yhteistyötä asiakkaan kanssa enemmän kuin hankalia sopimus neuvotteluita.

Agile-kehitys uskoo, että yhteistyö asiakkaan kanssa pitää projektin oikeilla raiteilla. Liian usein prosessit, työkalut ja dokumentaatio voivat aiheuttaa, että projektin päämäärä häviää projektiryhmän sisällä työskentelevien ihmisten mielestä. Tämä on seurausta siitä, että ihmiset rupeavat tulkitsemaan eritavalla sopimukseen painettua tekstiä. Agile projektiryhmillä on käytäntöjä, jotka ohjaavat niitä olemaan avoimia asiakkaan tarpeille. Lisäksi Agile edistääkin vain sopimussuhteita siten, että se rohkaisee työskentelemään asiakkaan kanssa ja asiakkaalle. (Schuh 2005, 4.)

4.4.4 Periaate 4

Arvostetaan muutoksiin vastaamista enemmän kuin suunnitelmien orjallista seuraamista.

Muutoksia tapahtuu tietojärjestelmäprojekteissa ja jokaisen projektiryhmän tulee muuntaa itsensä työskentelemään muutosten parissa, koska muuten toimitetaan alaarvoista tuotetta asiakkaan näkökulmasta katsottuna. Agile-kehitys tuo tähän käytäntöjä jotka kytkevät tiimit mukautumaan projektissa muuttuviin vaatimuksiin ja ympäristötekijöihin. Esimerkiksi: nopeasti saatava palaute, nopea reagointi ja päätökset muutoksiin sekä korkealaatuinen helposti ylläpidettävä ohjelman lähdekoodi ovat tällaisia menetelmiä, joilla voidaan vaikuttaa muutoksien jouhevaan käsittelyyn. Lisäksi Agile kannustaa siihen, että projektiryhmä voi tarvittaessa räätälöidä prosesseja, suunnitelmia, dokumentaatiota ja sopimuksia saavuttaakseen päämäärän. (Schuh 2005, 4.)

4.5 Agile ohjelmistokehityksen muut periaatteet

Agile manifestin osallistujat kirjailivat myös lyhyen listan joka tunnetaan nimellä ”Principles behind the Agile Manifesto.” Kyseinen lista on suositeltu ohjekirja kaikille, jotka hyödyntävät Agile kehityksen käytäntöjä ja periaatteita projekteissaan. (Schuh 2005, 5.)

4.5.1 Lista muista periaatteista

- Korkein tavoite on tuottaa asiakkaalle mahdollisimman aikaisessa vaiheessa korkealaatuisia ohjelmistoja.
- Hyväksytään muutokset jopa projektin myöhäisessä vaiheessa, mikä tuo kilpailuetua organisaatiolle.
- Julkaistaan toimiva ohjelma usein muutamasta viikosta muutamiin kuukausiin olevalla aikavälillä.
- Myynnin ja kehittäjien tulee työskennellä yhteistyössä päivittäin läpi projektin keston.
- Rakenna projekti hyvin motivoituneiden yksilöiden ympärille.

- Jotta työ saataisiin tehtyä, anna projektiryhmälle ympäristö ja tutki, mitä he tarvitsevat ja luota heihin.
- Tehokkain tapa jakaa informaatiota projektiryhmälle on kasvotusten tapahtuva palaveri.
- Toimiva ohjelmisto on projektin todellinen edistymisen mittari.
- Agile edustaa jatkuvaa kehitystä.
- Sponsorien, kehittäjien ja käyttäjien tulisi voida pitää tasainen työtahti jatkuvasti.
- Panostetaan jatkuvasti laadukkaaseen tekniseen suunnitteluun ja toteutukseen.
- Parhaat arkkitehtuuriratkaisut ja suunnitelmat syntyvät itse ohjautuvassa projektiryhmässä.
- Säännöllisin väliajoin projektiryhmän tulisi arvioida, miten se voisi tehostaa toimintaansa.
- Pidä suunnittelu ja toteutus yksinkertaisena, jolloin työn tekeminen tehostuu.

(Schuh 2005, 5-6.)

4.6 Ketterä projektiryhmä - Agile

Agile-metodit vaativat projektitiimien yksilöitä olemaan avoimia, kommunikoivia, yhteistyökykyisiä ja halukkaita jatkuvaan oppimiseen. Menestyvä projekti myös edellyttää, että ohjelmoijat haluavat työskennellä toistensa kanssa läheisessä yhteistyössä. Yhteistyö on oleellinen tapa jakaa tietoa ohjelman lähdekoodista ja toteuttaa optimaalisesti ratkaisuja.

Sellaisia ohjelmoijia ei arvosteta Agilen metodien mukaan, jotka haluavat työskennellä yksin suljetun oven takana. Agile metodien keskeinen piirre on siis kommunikaatio tiimin jäsenten välillä. Työn tekeminen ei voi olla tehokasta, jos kaikki ryhmän jäsenet eivät kysy, kun ongelmia tulee, anna palautetta toisilleen ja käsittele ongelmia monelta eri näkökulmalta. Asiakkaan kanssa kommunikoidessaan ohjelmistosuunnittelijoiden tulee olla sellaisia ihmisiä, jotka mielellään aloittavat keskusteluita, esittävät kysymyksiä ja keskustelevat vaihtoehdoista sen sijaan, että arvuuttelevat, milloin olisi tyhjää tilaa kalenterissa. (Schuh 2005, 9.)

Projektiryhmien koko on myös tärkeä seikka Agilen periaatteiden mukaan, koska pienemmät ryhmät voivat työskennellä tehokkaammin. Tähän perusteluina on, että pienet projektiryhmät voivat työskennellä samoissa tiloissa, jolloin kommunikointi helpottuu, ja ryhmän jäsenien ei tarvitse noudattaa kokoajan jotakin tiettyä muodollista prosessia. Projektiryhmä voi saada aikaan yhteisiä päätöksiä nopeammin ja helpommin. Ryhmä voi myös reagoida muutoksiin nopeammin. Isojen projektiryhmien kanssa suurin aika voi mennä organisointiin ja kommunikointiin. (Schuh 2005, 9.)

4.7 Pieniä julkaisuja kerrallaan – Agile

Agilesta muodostetuille metodeille ominainen piirre on, että järjestelmän kehitystyö jaetaan jaksoihin (iteroiva ohjelmistonkehitys) (ks. kuvio 4). Jokainen näistä jaksoista sisältää määrittely-, suunnittelu-, toteutus- ja testivaiheen. Tällä prosessimallilla yritetään varmistaa, että säännöllisin väliajoin julkaistaan jokaisen jakson lopussa uusi toimiva järjestelmä uusilla ominaisuuksilla varustettuna. Iteroivasta kehityksestä on listattu olevan seuraavaa hyötyä ohjelmistonkehitykselle:

1. Jaksot antavat asiakkaalle mahdollisuuden määritellä projektin suunta jokaisen jakson alussa ja määritellä, mitkä toiminnallisuudet ovat milloinkin tärkeimpiä toteuttaa.

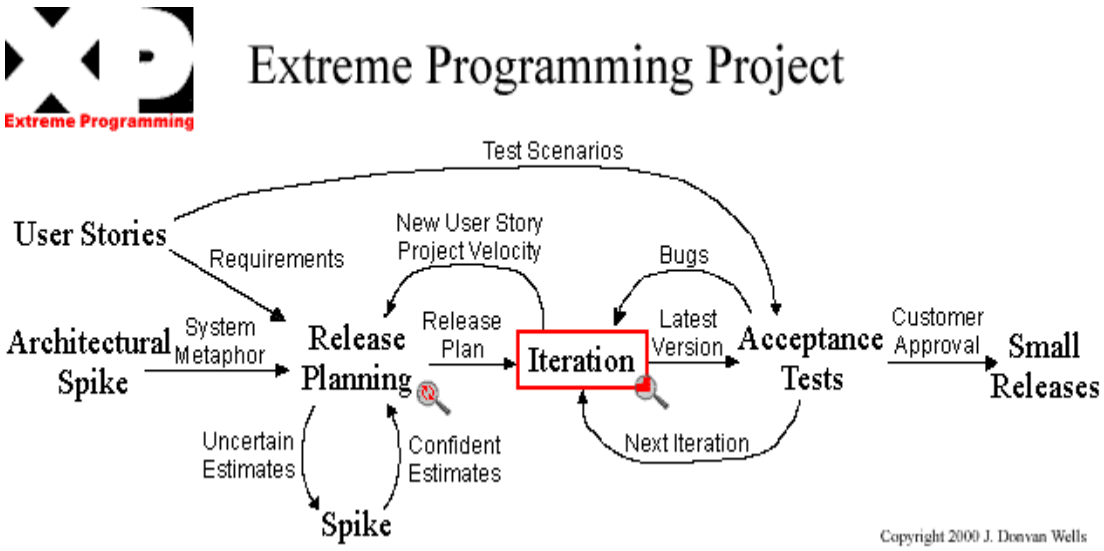
2. Iteroiva kehitys varmistaa lyhyillä ajanjaksoilla, että tuotekehitys ei jää polkemaan paikoilleen.

3. Iteroiva kehitys tarjoaa säännöllisiä kehitys ajanjaksoja ja näistä saatuja tuloksia voidaan käyttää helposti projektin edistymisen ja projektiryhmän tuottavuuden seuraamiseen.

4. Jokaisen jakson lopussa julkaistaan uusi järjestelmä uusilla ominaisuuksilla.

5. Asiakkaat saavat mahdollisuuden tarjota palautetta projektiryhmälle jokaisen jakson jälkeen.

(Schuh 2005, 209-210.)



KUVIO 4. Iteratiivinen ohjelmistonkehitys. (Extreme Programming 2006.)

5 SCRUM

5.1 Mikä Scrum on?

”Scrum on iteratiivinen ja inkrementaalinen ohjelmistonkehitysmenetelmä, joka tuottaa jokaisen iteraation päätteeksi valmiin osakokonaisuuden tuotteesta. Scrumia käytettäessä ohjelmistoprojektin edistymistä eri osa-alueilla (kustannukset, aikataulut, laatu, laajuus) on helppo seurata. Oikean tiedon avulla pystytään tekemään koko projektia koskevia päätöksiä, kehittämään toimintaa jatkuvasti ja vaikuttamaan projektia hidastaviin asioihin. Uusiin vaatimuksiin ja toimintaympäristössä tapahtuviin muutoksiin pystytään reagoimaan nopeasti ja hallitusti.” (Reactor Innovations 2006.)

Tavallisesti yksi iteraatio ilman suurempia muutoksia kestää yhdestä neljään viikkoa ja tätä jaksoa nimitetään sprintiksi. Scrum voidaan implementoida projektiin mukaan heti alussa tai se voidaan ottaa käyttöön myös kesken projektin. Siihen voidaan turvautua, kun huomataan, että tuotekehitys on vaikeuksissa. Useat organisaatiot ovatkin onnistuneesti ottaneet Scrumin käyttöön projekteissaan. Sitä on käytetty taloushallinnon-, internet- ja lääketieteellistenprojektien menetelmänä. Scrum sisältää seuraavia ominaisuuksia:

- Parantaa kommunikaatiota ja maksimoi yhteistyötä.
- Maksimoi tuottavuutta.
- Tekee työstä mukavaa ja saa ihmiset tekemään parhaansa.
- Poistaa esteitä, jotka haittaavat ohjelmiston kehitystä ja tuotteiden toimitusta.

(Control Chaos 2008.)

5.1.1 Tunnusmerkit, milloin Scrum ei ole käytössä ohjelmistoprojektissa?

Kysymyksiä herättää, minkälainen ohjelmistoprojekti ei käytä Scrumia apuna tuotekehityksessä. Melanie G. Silver (Scrum alliance 18.3.2007) esittää internet artikkelissaan kolme skenaariota tunnusmerkkinä, milloin ohjelmistoprojekti ei ole Scrumin mukainen.

Ensimmäinen skenaario: Ohjelmistoprojektin jäsenet päättävät ottaa Scrum-menetelmän käyttöön ja projektiorganisaation johto antaa tähän luvan. Kyseisen organisaation johto vaatii kuitenkin projektiryhmää noudattamaan tiettyä ohjelmistonkehitysprosessia. Kyseisen organisaation johto pyrkisi tätä vaatimalla varmistamaan sen, että projektiryhmä tekee, mitä heidän odotetaan tekevän. (Silver 2007.)

Scrum-menetelmän mukainen projektiryhmä on itse organisoituva ja päättää itse miten työt tehdään (ks. 5.4.3 Scrum team).

Toinen skenaario: Ohjelmistoprojektin jäsenet päättävät käyttää Scrumia apuna muussa kuin ohjelmistoprojektissa. Projektiryhmää ei ole olemassa tai resursoituna ja product ownerilla ei ole aikaa priorisoida ohjelmistotuotteen ominaisuuslistaa. Tai product ownerin mielestä kaikki tuotteen ominaisuudet on priorisoitava tärkeiksi samalla kertaa. (Silver 2007.)

Sprint planning meeting – palaverissa priorisoidaan toteutettavan tuotteen ominaisuudet. Tiimi valitsee tehtävät, jotka se uskoo pystyvänsä suorittamaan seuraavan sprintin aikana (ks. 5.3.1 Sprint Planning Meeting).

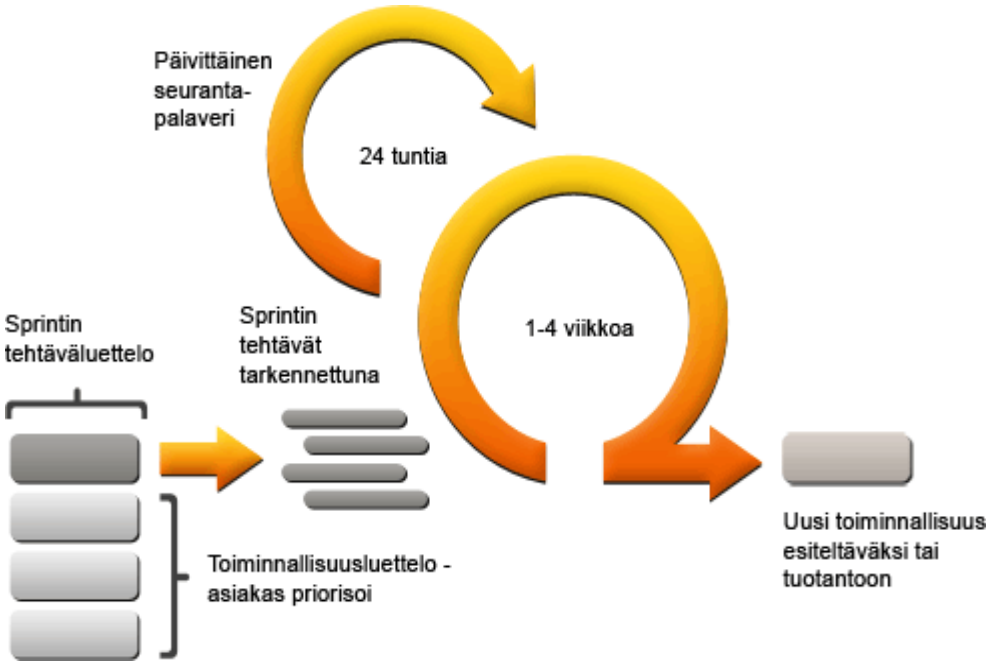
Kolmas skenaario: Ohjelmistoprojektin jäsenet päättävät käyttää Scrum-menetelmää apuna projektissaan, mutta projektiryhmä päättääkin pitää päivittäiset ryhmän palaverit vain kahdesti viikossa. Scrum rooleja ei ole otettu käyttöön projektiryhmässä. Projektiryhmä hyödyntää vain joitakin Scrumin periaatteita ohjelmistoprojektissaan. (Silver 2007.)

5.2 Scrum - termejä

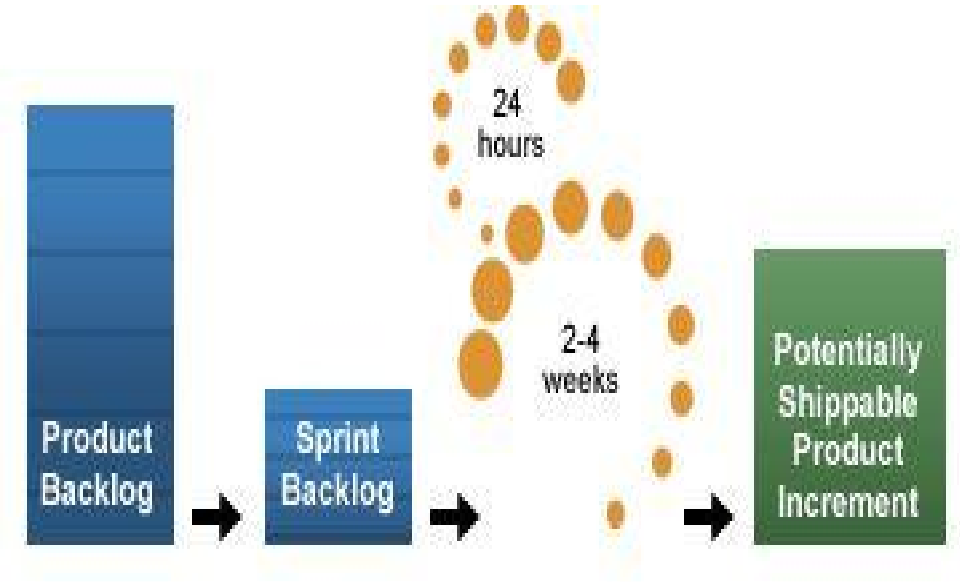
- Product Backlog: on lista, johon kerätään kaikki tuotteeseen kuuluvat vaatimukset ja tiedot sekä toiminnallisuudet. Listan vaatimukset priorisoidaan ja sitä päivitetään tarvittaessa koko tuotannon ajan. (SCRUM Alliance 2007.)
- Product backlog item: Product backlogissa olevia kirjauksia, jotka ovat tuotteeseen liittyviä työtehtäviä tai tämän ominaisuuksia. (SCRUM Alliance 2007.)
- Sprint planning meeting: Suunnittelupalaveri, jossa seuraava tuotekehitysjakso (iteraatio) suunnitellaan ja pyyhkäistään käyntiin. (SCRUM Alliance 2007.)
- Daily scrum meeting: Päivittäinen palaveri, jossa jokainen työntekijä kertoo omista työtehtävistään, mitä aikoo seuraavaksi tehdä ja onko työn etenemistä haittaavia ongelmia. (SCRUM Alliance 2007.)
- Sprint review meeting: Palaveri, jossa esitellään ja hyväksytään kuluneen jakson tehtävät. (SCRUM Alliance 2007.)
- Sprint: on tuotteen kehitysjakso Scrumin mukaisessa tuotekehitysprojektissa, joka kestää yleensä 30 vuorokautta. (SCRUM Alliance 2007.)
- Sprint backlog: Tiimit pitävät toiminnastaan kirjaa, johon kirjaavat jakson aikana onnistuneita ja epäonnistuneita tapahtumia. Sprit review meeting – palaverissa nämä kirjaukset käydään läpi. (SCRUM Alliance 2007.)
- Release: Jakson jälkeen julkaistaan uusi versio kehitettävästä tuotteesta ja tapahtumasta käytetään käsitettä release. (SCRUM Alliance 2007.)
- Product owner, Scrum master, team ovat Scrumin mukaisessa tuotekehityksessä rooleja. (SCRUM Alliance 2007.)

5.3 Scrum ja sen kolme seremoniaa

Scrum sisältää kolme erillistä päätapahtumaa, jotka ovat uuden jakson suunnittelupalaveri, päivittäiset tiimin palaverit ja jakson lopussa katselmointi- palaveri (ks. kuvio 5 ja kuvio 6). (SCRUM Alliance 2007.)



KUVIO 5. Esimerkkikaavio Scrum tuotekehitysprosessista. (Reactor Innovations 2006.)



KUVIO 6. Scrumin mukainen tuotekehitysprosessi. (SCRUM Alliance 2007.)

5.3.1 Sprint Planning Meeting

Kaikki alkaa, kun product owner toimittaa suunnitelman tuotteesta tai projektista sen tekijöille. Product owner voi olla asiakkaan edustaja tai valtuutettu henkilö. Hänellä pitää olla näkemys siitä, minkälainen tuotteen pitäisi olla. Varsinainen Scrum alkaa, kun tuotteen ominaisuuksia on määritelty tarpeeksi ja priorisoitu tuotteesta olevaan backlogiin. Tämän jälkeen product owner esittelee vision tuotteesta, etenemis- ja julkaisusuunnitelman projektitiimille. Tiimi määrittelee töihin kuluvan tarkan ajan tuotteen backlogin mukaan ja valitsee, kuinka paljon se pystyy seuraavan 30 päivän aikana tuotetta kehittämään. Arvioinnissa tiimin täytyy ottaa huomioon oma kokonsa ja tuottavuutensa. Tämän jälkeen Scrum Master johdattaa tiiminsä suunnittelemaan ja analysoimaan valittuja ominaisuuksia, joista muodostetaan työtehtäviä jakson ajalle. Sprint planning meeting ei saisi kestää kauemmin kuin neljä tuntia. (SCRUM Alliance 2007.)

Sprint on määritelty ajanjakso Scrum-menetelmän mukaisessa tietojärjestelmäprojektissa, jonka aikana kehitetään ohjelmistotuotetta (Scrumissa oletuksena 30 päivää). Jokainen sprint päättyy jonkin tuotteeseen kuuluvan toiminnallisuuden julkaisemiseen. Tätä tapahtumaa kutsutaan käsitteellä release. Tiimit pitävät toiminnastaan kirjanpitoa

(sprint backlog), joka esitellään jakson päätöspalaverin yhteydessä. (SCRUM Alliance 2007.)

5.3.2 Daily Scrum Meeting

Daily Scrum meeting on päivittäinen tilaisuus, joka järjestetään aina samaan aikaan ja samassa paikassa. Tämä tilaisuus on oivallinen tilanne tarkastaa, missä vaiheessa tiimi on kulloinkin menossa. Palaverissa jokainen tiimin jäsen kertoo, mitä on tehnyt ja mitä seuraavaksi aikoo tehdä. Jos tiimillä on erityisiä ongelmia, ne voidaan huomioida nopeasti ja niiden ratkaisemiseksi voidaan järjestää erikseen uusi palaveri. Daily Scrum–palaveriin saavat osallistua muutkin kuin tiimiin kuuluvat jäsenet, mutta he eivät saa kommentoida mitään. Jokaiselle tiimin jäsenelle tulee varata vähintään 15 minuuttia aikaa vastata kolmeen kysymykseen:

- Mitä tiimin jäsen on tehnyt viimeisimmän palaverin jälkeen?
- Mitä hän aikoo tehdä seuraavaksi tämän palaverin jälkeen?
- Mitkä ongelmat haittaavat työn edistymistä?

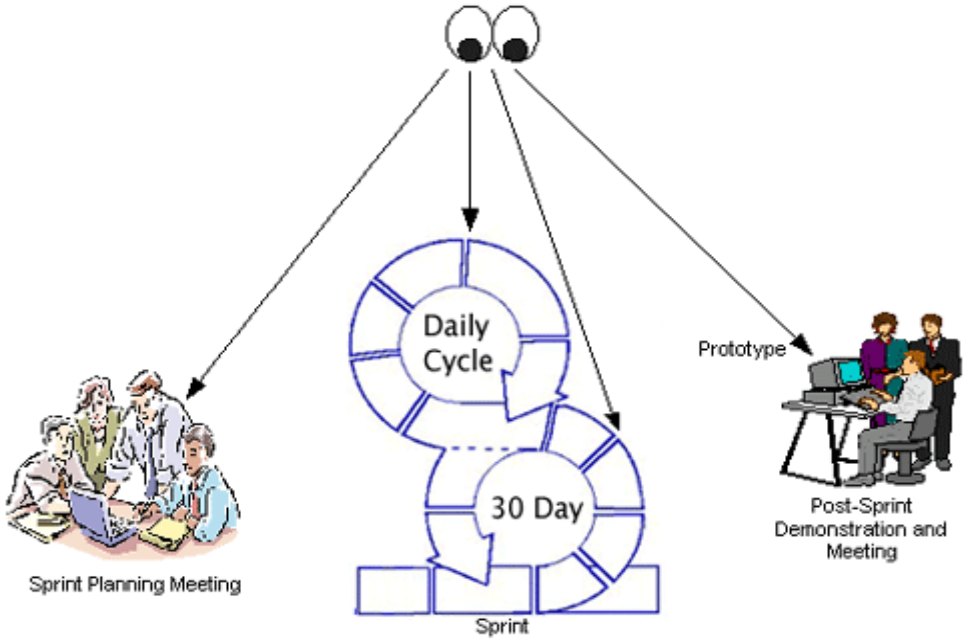
(SCRUM Alliance 2007.)

5.3.3 Sprint Review Meeting

Jokaisen sprintin lopussa järjestetään palaveri, joka kestää noin neljä tuntia. Tämä palaveri on jaettu kahteen osaan siten, että ensimmäistä puoliskoa johtaa product owner. Hän on mahdollisesti kutsunut asiakkaan puolelta tuotteesta kiinnostuneet muut johtajat tai asiakas organisaation muita henkilöitä mukaan palaveriin. Palaverin ensimmäisellä puoliskolla esitellään tuotteeseen tehdyt ominaisuudet, jotka olivat valittuina jaksolle. Tällöin myös hyväksytään tai hylätään toteutuneet tulokset. Product owner neuvottelee tiimin kanssa, miten seuraavalle jaksolle priorisoidaan tehtävät, tavoitteet ja mitä aletaan tehdä. Toista puolikasta Sprint Review Meeting-palaverista johtaa Scrum master, joka käy tiimin kanssa läpi seuraavia asioita sprint backlogista: Mitä positiivisia tapahtumia oli ilmennyt sprintin aikana, joita tiimi voisi hyödyntää seuraavalla sprintilla. Lisäksi käsitellään asiat, jotka olisivat voineet mennä paremmin. Tämän jälkeen aloitetaan ohjelmiston-kehityksessä uusi sprint. (SCRUM Alliance 2007.)

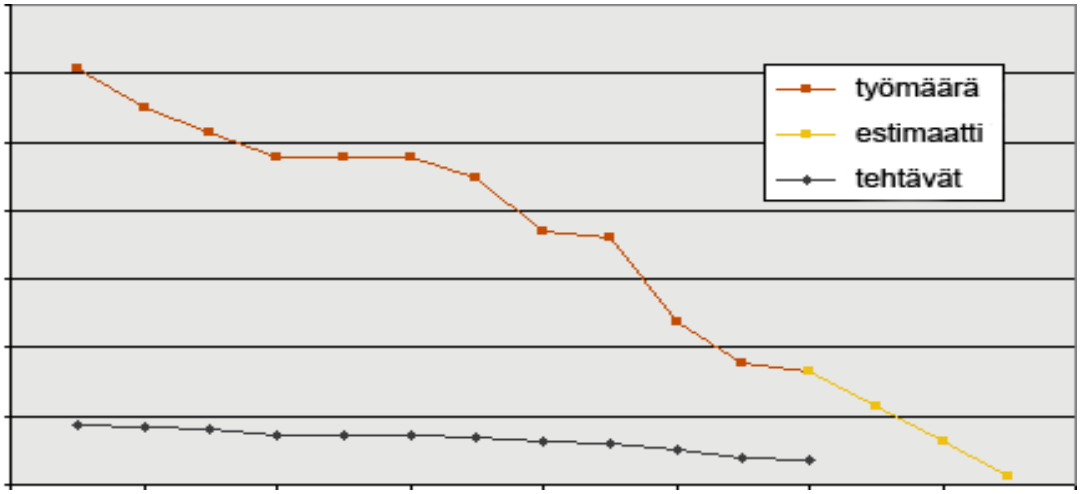
5.3.4 Johtajat ajan tasalla

Organisaation johtajat voivat valvoa projektin edistymistä Scrumissa seuraavasti: johtajat voivat osallistua päivittäisiin palavereihin jakson aikana, tulevan uuden kehitysjakson suunnittelupalaveriin ja jakson päättymispalaveriin (ks. kuvio 7). Näiden palaverien kuluessa he voivat tarkkailla projektiryhmän henkistä tilaa, jokaisen ryhmän jäsenen osallistumista ja vuorovaikutustaitoja, töitä, jotka on saatettu päätökseen ja projektin edistymistä haittaavia ongelmia. Näin organisaation johtajat tietävät, mitä projektissa tapahtuu. He voivat myös tarvittaessa ryhtyä toimenpiteisiin, jos ilmenee ongelmia ja aihetta huoleen. (Control Chaos 2008.)



KUVIO 7. Scrum mahdollistaa valvonnan. (Control Chaos 2008.)

Tuotekehitystiimi julkaisee usein sprintin työmäärän vähentymistä kuvaavan kaavion. Kaaviosta voidaan seurata, onko valmistuminen arvioitua nopeampaa tai hitaampaa, ja pystytään toimimaan sen mukaan (ks. kuvio 8). (Reactor Innovations 2006.)



KUVIO 8. Kaavio valmiista työtehtävistä Sprintin aikana. (Reactor Innovations 2006.)

5.4 Scrum-roolit

5.4.1 Scrum master

Scrum master vastaa tiimien työskentelystä ja siihen liittyvistä päätöksistä tuotekehitysjakson aikana. Hänen tärkeimpinä tehtävinään ovat seuraavat asiat:

- Yrittää poistaa muurit projektiryhmän ja product ownerin väliltä, että tämä voi ohjata tuotteen kehitystyötä.
- Kehittää projektiryhmää kaikin keinoin, että tiimi olisi parempi ja tuottavampi.
- Päivittää projektin edistymistä ja projektiryhmää koskevia asioita kokoajan ja pitää ne kaikkien tiimin jäsenien saatavilla.
- Hankkia tarvittavat työkalut tiimin käyttöön, jotta kehitystyö olisi tehokasta.

(SCRUM Alliance 2007.)

5.4.2 Product owner

Product owner on vastuussa projektin tuotteesta ja hänen vastuualueisiin kuuluvat seuraavat asiat: määritellä kehitettävän tuotteen ominaisuudet, päättää julkaisupäivä ja sisältö, huolehtia kannattavuudesta, lisätä ominaisuuksia ja tarvittaessa priorisoida

niitä joka 30 vuorokauden kuluttua (sprint) ja hyväksyä tai hylätä jakson tulokset. (SCRUM Alliance 2007.)

5.4.3 Scrum team

Scrum tiimin koko on yleensä viidestä kahdeksaan ihmistä. Tiimi valitsee jokaisen sprintin alussa tavoitteet ja määrittelee työn tulokset. Kunhan vain pysytään projektisuunnitelman rajoissa, tiimillä on valtuudet tehdä kaikki toimenpiteet, jotta tavoitteisiin päästään. Tiimi voi organisoitua omalla tavallaan ja päättää, kuinka työt tehdään. Sprintin lopussa tiimin tulee esitellä jakson tulokset product ownerille. (SCRUM Alliance 2007.)

5.5 Scrumin arvot ja ideologia

Scrum-ideologia perustuu kokemukseen ja innovatiivisuuteen sekä oppimiseen ja osittain myös psykologiseen näkökulmaan. Scrum-tiimit ovat vapaita tekemään itse päätöksiä ja suunnittelemaan seuraavan jakson työn itse sopimallaan tavalla. Pääasia on, että saadaan tulosta aikaiseksi sekä työskennellään keskittyneesti tavoitteen saavuttamiseksi. Schwaber (2002) toteaa kirjassaan, että tiimin on hänen mielestään turhaa rakentaa mallinnusmenetelmän mukaisia kaavioita, jos sitä tehdään vain sen vuoksi, koska sitä on tehty viimeiset kymmenen vuotta. Jos tiimi haluaa ohjata ja suunnitella toimintaansa ja ajatuksiansa mallinnuksen avulla, on siihen siinä tapauksessa hänen mielestään järkevä syy. Tärkeää on, että keskitytään olennaiseen eli mietitään ennemminkin sanojen merkityksiä, kuin pelkästään sitä, mikä sana on. (Schwaber & Beedle 2002, 29, 90.)

Scrum-ideologiaan kuuluvat myös seuraavat asiat:

- Commitment: pitää olla halua päästä tavoitteeseen. Scrum tarjoaa kaiken mahdollisen päätäntävallan tiimille, jotta se saavuttaa maalin.
- Focus: pitää keskittää kaikki huomiosi siihen, mitä olet tekemässä äläkä huolehdi mistään muusta.
- Opennes: Scrum pitää kaiken projektissa avoimena kaikille osapuolille.

- Respect: yksilöt ovat muovautuneita erilaisten taustojen ja kokemusten myötä. Tiimin erilaisia yksilöitä tulee kunnioittaa.
- Courage: pitää olla rohkeutta toimia, olla avoin ja kunnioittaa muita saavuttaaksesi tavoitteet.

(Schwaber & Beedle 2002, 29, 90.)

6 TUTKIMUKSEN TOTEUTTAMINEN

Tutkimukseen valmistauduttiin keräämällä tarvittavia taustatietoja ohjelmistotuotannon osa-alueista ja ohjelmistoprojektiin läheisesti liittyvistä käsitteistä. Lisäksi selvitettiin, mitä ohjelmistoalaa paljon kiinnostava Agile käsitteenä tarkoittaa ja mitä ominaispiirteitä se sisältää. Opinnäytetyön tarkoituksena on selvittää tutkimuksen kohteena olevan informaatioteknologiaorganisaation tuotekehitysprosessin ominaisemmat piirteet; miten ohjelmistotuotetta kehitetään kyseissä organisaatiossa ja verrata tätä prosessia Scrum-menetelmän mukaiseen ohjelmiston-kehitykseen. Tarkoituksena on myös raportoida tulokset, mitä organisaation tulisi muuttaa tuotekehitysprosessissaan, jotta se olisi Scrumin mukainen. Tavoitteena on tutkia hieman sovelluskehitystiimin asenteita, miltä kyseinen menetelmä tuntuisi heistä. Tutkimuksen suorittaminen onnistuneesti vaatii myös selvitystä siitä, minkälainen ohjelmiston kehitysmenetelmä Scrum on ja mitä ominaispiirteitä se sisältää.

6.1 Tutkimusaineiston kerääminen

Tutkimusmenetelmäksi valittiin kysely (ks. 2 Tutkimusasetelma) ja aineiston kerääminen suoritettiin tutkimukseen osallistuvilta kyselylomakkeen avulla. Kohderyhmänä oli Yritys X:n ohjelmistoprojektissa työskentelevät henkilöt ja heille lähetettiin kyselylomakkeen www-osoite henkilökohtaisesti sähköpostilla. Samalla ohjeistettiin vastaajia, kuinka lomakkeen kysymyksiin tulee vastata. Kohderyhmää käytiin myös henkilökohtaisesti informoimassa, kuinka tärkeä kysely on opinnäytetyön näkökulmasta. Kyselyyn osallistuva kohderyhmä sai viisi vuorokautta aikaa vastata kysymyksiin ja lähettää vastaukset. Vuorokausi ennen aineiston keräämisajan päättymistä kohderyhmälle lähetettiin muistutus kyselyyn vastaamisesta.

Kyselylomake oli koodattu internetissä olevalle www - sivulle elektroniseen muotoon, koska tällöin kyselyyn vastaajat voivat työnsä ohella tietokoneella istuessaan naputella kysymyksiin vastaukset. Ideana oli, että vastaajien on helppo pelkällä napin painalluksella lähettää vastaukset, eikä heidän tai tutkijan tarvitse käyttää aikaa ja vaivaa lo-

makkeiden postittamiseen. Tällä tavalla ajateltiin saada kyselyvaihe nopeammin ja helpommin läpi sekä päästä heti analysoimaan kohderyhmän vastauksia.

Aineiston keräämistä harkittiin myös Microsoft Excel taulukkolaskenta-ohjelmalla tehdyn kyselylomakkeen avulla. Ideana oli, että toimitetaan vastaajille tiedosto, joka sisältää kysymykset. Vastattuaan kohderyhmä lähettäisi sen takaisin tutkijalle sähköpostin välityksellä. Idea tiedoston käsittelystä ja sähköpostin välityksellä vastaamisesta hylättiin, koska tällöin vastaajat saattaisivat jättää vastaamatta kyselyyn helpommin. Tähän on syynä se, että ihmiset haluavat todennäköisesti vastata tutkimuksiin anonyymisti. Muutenhan tutkija tietäisi, mitä kukin henkilö on vastannut tiedoston ja sähköpostin lähettäjän perusteella. Kyselyyn vastaajat saattaisivat myös tuntea tiedoston ja sähköpostin käsittelyn turhan työläänä. Vastaukset saattaisivat myös hävitä jonnekin sähköpostin lähetyksen yhteydessä ja niitä voisi joutua karhuamaan vastaajilta.

Elektronisessa muodossa olevan kyselylomakkeen oikein toimiminen testattiin, jotta vastaukset tulisivat kunnolla perille. Testauksella tarkoitetaan tässä tapauksessa sitä, että tutkimukseen osallistuvan henkilön antamat vastaukset kohdistuvat oikeisiin kysymyksiin tietoja tallennettaessa, kun vastaukset kyselylomakkeelta lähetetään.

6.2 Kyselylomakkeen kysymykset

1. Miten tietojärjestelmäprojekti käynnistettiin projektissa, jossa olet mukana?

Kysymyksellä yritetään selvittää, minkälainen palaveri järjestetään ja mitä kyseisessä palaverissa tapahtuu, kun kohdeorganisaatiossa aloitetaan uusi tietojärjestelmäprojekti. Tarkoituksena on verrata tämän kysymyksen avulla saatuja vastauksia Scrum-menetelmän mukaiseen ”Sprint Planning Meeting”-palaveriin, jolla aloitetaan Scrumin mukainen ohjelmistonkehitys. Tämän kysymyksen avulla pitäisi pystyä analysoimaan vastauksia seuraavaan tutkimusongelmaan: ”Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi?”

2. Miten uudet työtehtävät jaetaan projektiryhmälle?

Kysymyksen vastauksilla pyritään myös selvittämään tutkimusongelmaa: ”Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi?” Tässä ohjelmiston kehitys-menetelmässä sprintin jälkeen järjestetään ”Sprint Review Meeting”–palaveri ja tällöin hyväksytään kuluneen kehitysjakson tulokset ja aloitetaan uusi sprint uusien tehtävien kanssa. Tässä tapahtumassa verrataan, miten kohdeorganisaatio toimii tilanteessa, kun sovelluskehitystiimille jaetaan uusia työtehtäviä.

3. Miten palaverit tuotteen kehitykseen liittyen järjestetään projektiryhmän kanssa?

Scrumin mukaan työskentelevä sovelluskehitystiimi pitää joka päivä lyhyitä palavereja liittyen tiimin jäsenten työtehtäviin ja näitä palavereja kutsutaan ”Daily Scrum Meeting”–tapahtumiksi. Tällä kyselylomakkeen kysymyksellä selvitetään, miten tutkimuksen kohdeorganisaation sovelluskehitystiimi järjestää palavereja tuotteen kehittämiseen liittyen ja kuinka usein. Vastausten analysoinnilla pitäisi saada vastaus tutkimusongelmaan: ”Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi?”

4. Miten projektiryhmän tehdyt työt hyväksytään?

Scrumissa sprintin jälkeen järjestetään ”Sprint Review Meeting”-palaveri, jossa käydään kuluneen tuotekehitysjakson tehtävät läpi ja hyväksytään tuotokset. Tällä kysymyksellä pyritään selvittämään, miten tutkimuksen kohdeorganisaatio toimii samassa tilanteessa. Kysymyksen vastauksia analysoitaessa pitäisi saada vastaus tutkimusongelmaan: ”Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi?”

5. Miten sovellustuotteen tuotekehitysprosessi etenee?

Kysymyksen vastausten analysoinnilla pitäisi saada vastaus tutkimusongelmaan: ”Miten tutkimuksen kohdeorganisaation tuotekehitysprosessi etenee?”

6. Valitse seuraavasta listasta kohdat, joista ajattelisit olevan hyötyä projektissa, jossa olet mukana:

6.1 Avoimuus ja asioista tiedottaminen.

6.2 Yhteistyö- ja vuorovaikutustaidot.

6.3 Rohkeus tehdä omia teknisiä ratkaisuja.

6.4 Päivittäinen palaveri, jossa käytäisiin edellisen päivän työt läpi.

6.5 Suunnitellaan vähän kerrallaan, koodataan, testataan ja julkaistaan uusi versio ohjelmasta.

7 TUTKIMUKSEN TULOKSET

Kysely suoritettiin kohdeorganisaation (Yritys X) sovelluskehityksessä työskentelevil-le ihmisille, jotka ovat ohjelmistosuunnittelijoita ja tuotepäälliköitä. Kyselyyn vastasi seitsemän ihmistä, kun kyselylomake lähetettiin kymmenelle. Kyselyn avoimien ky-symyksien vastauksien analysoiminen tehdään tarinamuotoon tässä opinnäytetyössä, koska näin koetaan tuloksien raportointi mielekkäämmäksi. Kohdeorganisaatio on myös dokumentoinut oman tuotekehitysprosessimallinsa, johon tutustutaan lähemmin tässä opinnäytetyössä.

7.1 Tulosten analysointi tutkimusongelmittain

7.1.1 Miten tutkimuksen kohdeorganisaation tuotekehitysprosessi etenee?

Tämän tutkimusongelman analysoimisessa apuna on käytetty tutkimuksen kohdeor-ganisaation tuotekehitysprosessimallien dokumentteja. Kyseisen organisaation tuote-kehitystä tapahtuu neljässä päävaiheessa, joita ovat toteutuksen suunnittelu (ks. kuvio 9), sovelluksien toteutus (ks. kuvio 10), asennuspaketin tekeminen (ks. kuvio 11) ja asennus tuotantoon (ks. kuvio 12).

7.1.1.1 Kohdeorganisaation tuotekehitysprosessi: toteutuksen suunnittelu

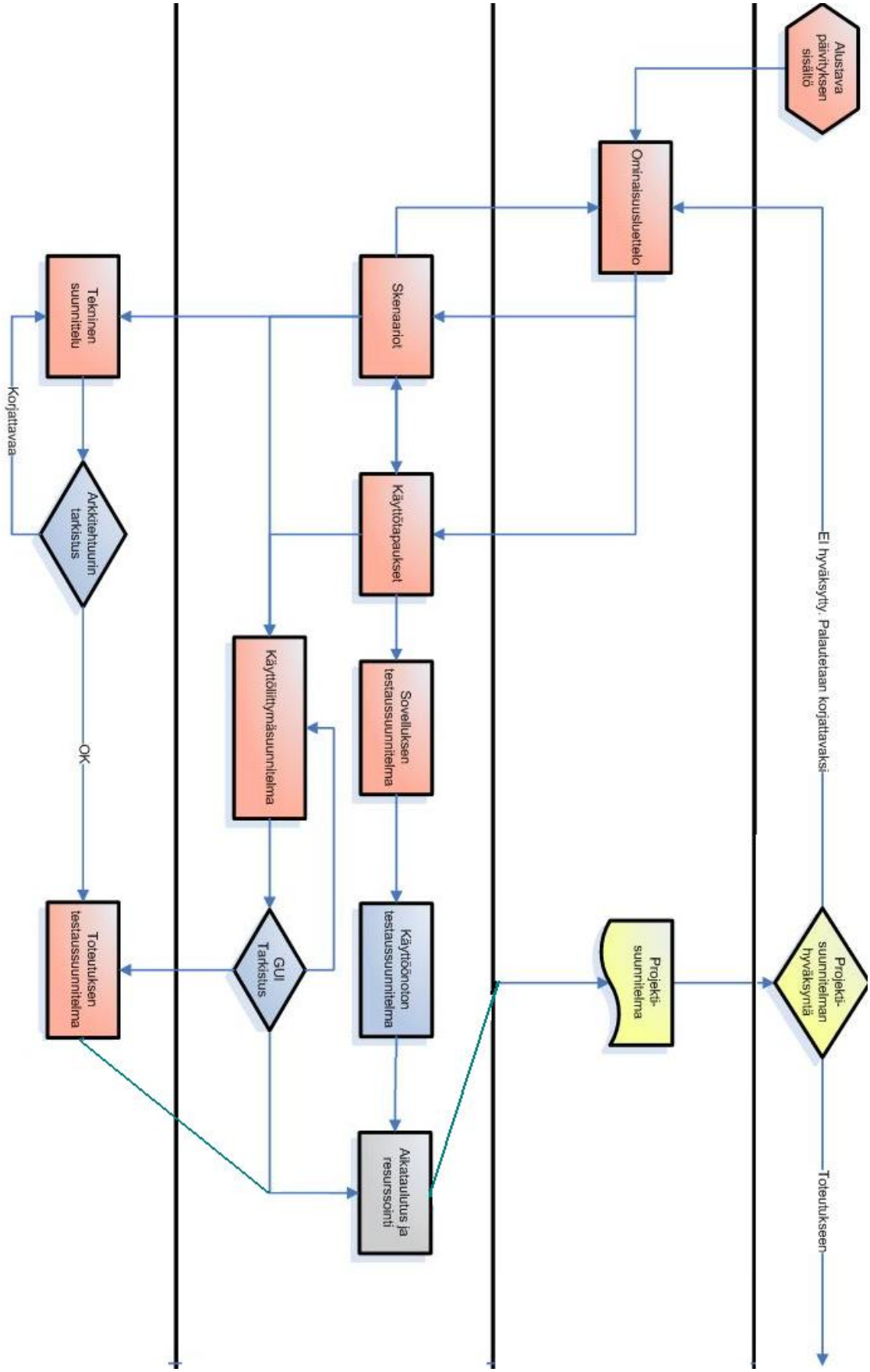
Kohdeorganisaation tuotekehitysprosessi etenee neljässä päävaiheessa, jossa ovat to-teutuksen suunnittelu (ks. kuvio 9), sovelluksen toteutus, asennuspaketin tekeminen ja asennus tuotantoon.

Yritysxn ohjelmistojen toteutuksen suunnitteluvaihe lähtee käyntiin sovelluskehityk-sen ohjausryhmän palaverista, jossa ovat mukana ohjelmistotuotekehityksen johtaja, tekninen arkkitehti, ohjelmisto-suunnittelijoita, ohjelmistotuotteen tuotepäällikkö, projektiorganisaation myyntipäällikkö ja asiakkaan edustajat. Tässä palaverissä käy-dään tulevan ohjelmistotuotteen alustava sisältö läpi sovellusohjausryhmään osallistu-jien kesken. Palaverin tuotoksena syntyy ominaisuusluettelo ohjelmistotuotteesta, jota ollaan tekemässä.

Perustavan sovellusohjausryhmän palaverin jälkeen tuotepäällikkö tekee käyttötapaukset, joiden perusteella ohjelmistosuunnittelijat laativat teknistä määrittelyä ja testaussuunnitelman. Projektin aikataulu muodostetaan näiden määrittelyiden perusteella. Tuotepäällikkö ja ohjelmistosuunnittelijat käyvät määrittelyitä läpi ja arvioivat, paljonko mihinkin ohjelmistotuotteen ominaisuuden toteuttamiseen kuluu aikaa.

Seuraavaksi tuotepäällikkö tekee projektisuunnitelman käyttötapausten ja ohjelmistosuunnittelijoiden määritysten perusteella. Projektisuunnitelma hyväksytään sellaisenaan tai korjauksien jälkeen organisaation omassa sovelluskehityksen ohjausryhmäpalaverissa, jossa ovat ainoastaan projektiorganisaation omat työntekijät mukana. Taloushallinnon ammattilaiset ovat myös mukana tässä palaverissa, koska ohjelmistonkehityksessä pitää huomioida myös projektiorganisaation oma- ja asiakkaan liiketoiminta. Projektisuunnitelmassa on tarjous siitä, mitä ohjelmistoprojekti tulee asiakkaalle maksamaan. Projektisuunnitelma sisältää myös aikataulun, milloin ohjelmistoprojektin tuloksena syntyvä ohjelmistotuote on valmis luovutettavaksi asiakkaan käyttöön. Palaverissa viilatut tarvittavat korjaukset suoritetaan projektisuunnitelmaan ja korjaukset vielä katselmoidaan.

Tämän jälkeen projektisuunnitelma hyväksytään esitettäväksi asiakkaalle. Järjestetään palaveri, jossa ovat mukana projektiorganisaation tekninen johtaja, tuotepäällikkö, myyntipäällikkö ja asiakkaan edustajat. Tässä palaverissa asiakkaalle esitellään projektisuunnitelma, aikataulut, kustannukset ja hyötysuhdeanalyysit. Tämä on ohjelmistoprojektin tärkein vaihe, ja tässä ei voida epäonnistua. Muuten ohjelmistoprojektia tuskin aloitetaan kyseisen asiakkaan kanssa. Tähän Yritysx on panostanut paljon kouluttamalla asiakasrajapinnassa toimivia henkilöitä.



KUVIO 9. Toteutuksen suunnittelu. (Yritysx prosessikaaviot.)

7.1.1.2 Kohdeorganisaation tuotekehitysprosessi: sovelluksien toteutus

Toteutuksen suunnitteluprosessin jälkeen alkaa toteutusprosessi (ks. kuvio 10), joka lähtee liikkeelle hyväksytystä projektisuunnitelmasta. Toteutuksen alussa järjestetään ”KickOff”-palaveri, jossa pidetään perehdyttämiset ja muut tarvittavat toimenpiteet. Tämän jälkeen pidetään palavereja säännöllisesti viikoittain. Viikkopalavereihin on määritelty tarkasti noudatettava esityslista sen takia, että palaverit eivät veny turhan pitkiksi. Viikkopalaverin esityslista etenee seuraavasti:

1. Tarkistetaan edellisen viikon tehtävien suoritukset.

2. Päivitetään projektin status.

    2.1 Verrataan jäljellä olevaa työmäärää jäljellä olevaan työaikaan.

    2.2 Negatiivisesta poikkeamasta ilmoitetaan mahdollisimman nopeasti ohjelmistoprojektin ohjausryhmälle.

3. Käsitellään esiin nousseet ongelmat ja sovitaan toimenpiteet ongelmien ratkaisua varten.

    3.1 Projektipäällikön tehtävänä on hoitaa ongelmien ratkaisut.

    3.2 Koodaajien ensisijainen tehtävä on toteuttaa projektin tehtäviä eteenpäin.

4. Sovitaan seuraavan viikon tehtävät.

Toteutusta viedään eteenpäin siten, että koodaajat suorittavat viikkopalavereissa sovittuja tehtäviä ja tekevät korjauksia testauksesta palautuneisiin virheellisiin tehtäviin. Kun toteutus on valmis, vaihdetaan sille tilaksi testaus ja siirretään se varsinaiseen testaukseen. Luodaan testityökaluun testit ja linkitetään testiin vaikuttavat tiedostot. Varmistetaan, että testaajalla on myös avoin pääsy työtehtävään.

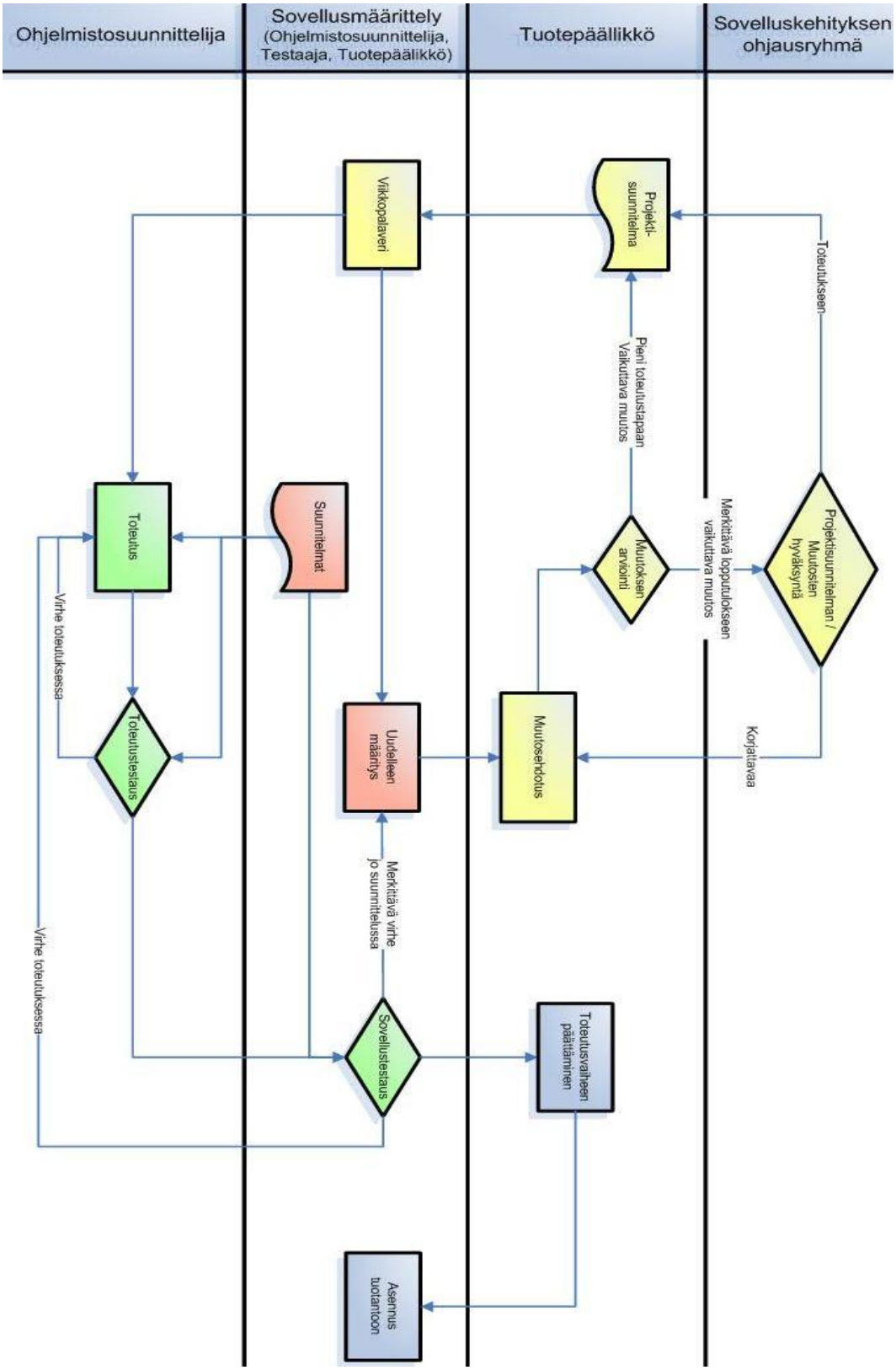
Toteutuksen testaaminen tapahtuu seuraavasti: käyttötapauksia testataan, onnistuuko käyttötapauksen mukainen toiminta. Tarkistetaan alkuehdot, päästäänkö lopputulokseen ja onko virheiden hallinta oikein.

Seuraavaksi testataan syöttökentät, onko jokainen syöttökenttä määritelty ja toteutettu määritelmän mukaisesti ja tallentuuko kaikki kirjoitettu data. Voiko kenttään syöttää virheellisen arvon ja voiko kentän jättää tyhjäksi. Kokeillaan myös, tallentuvatko syötetyt arvot oikein tietokantaan.

Web-sovelluksien ollessa kyseessä, tarkistetaan html:n validointi. Toimintoja toteuttavien sivujen pitää läpäistä html-validointi 100-prosenttisesti tuetuilla selaimilla. Sivujen lähdekoodin kommentointi pitää myös tarkistaa.

Sovelluksen toimintoja toteuttavien luokkien lähdekoodi pitää myös tarkistaa ja testata. Toteutuksen tekijä tekee luokalle testit, mitä luokka palauttaa ja miten virheiden käsittely onnistuu. Pitää myös testata toteutettuun luokkaan liittyvien muiden luokkien oikein toimiminen. Projektin dokumentaatio pitää myös katselmoida ja tarkastaa, että esimerkiksi tietokannan jokainen sarake on dokumentoituna.

Kun sovellusta on riittävästi testattu ja virheitä ei enää havaita sovellustestauksen jälkeen suoritettujen korjaavien toimenpiteiden ja dokumenttien katselmointien jälkeen, päätetään toteutusvaihe. Seuraavaksi aloitetaan asennuspaketin tekeminen, minkä jälkeen suoritetaan tuotantoasennuksen mukainen prosessi.



KUVIO 10. Sovelluksien toteutus. (Yritysx prosessikaaviot.)

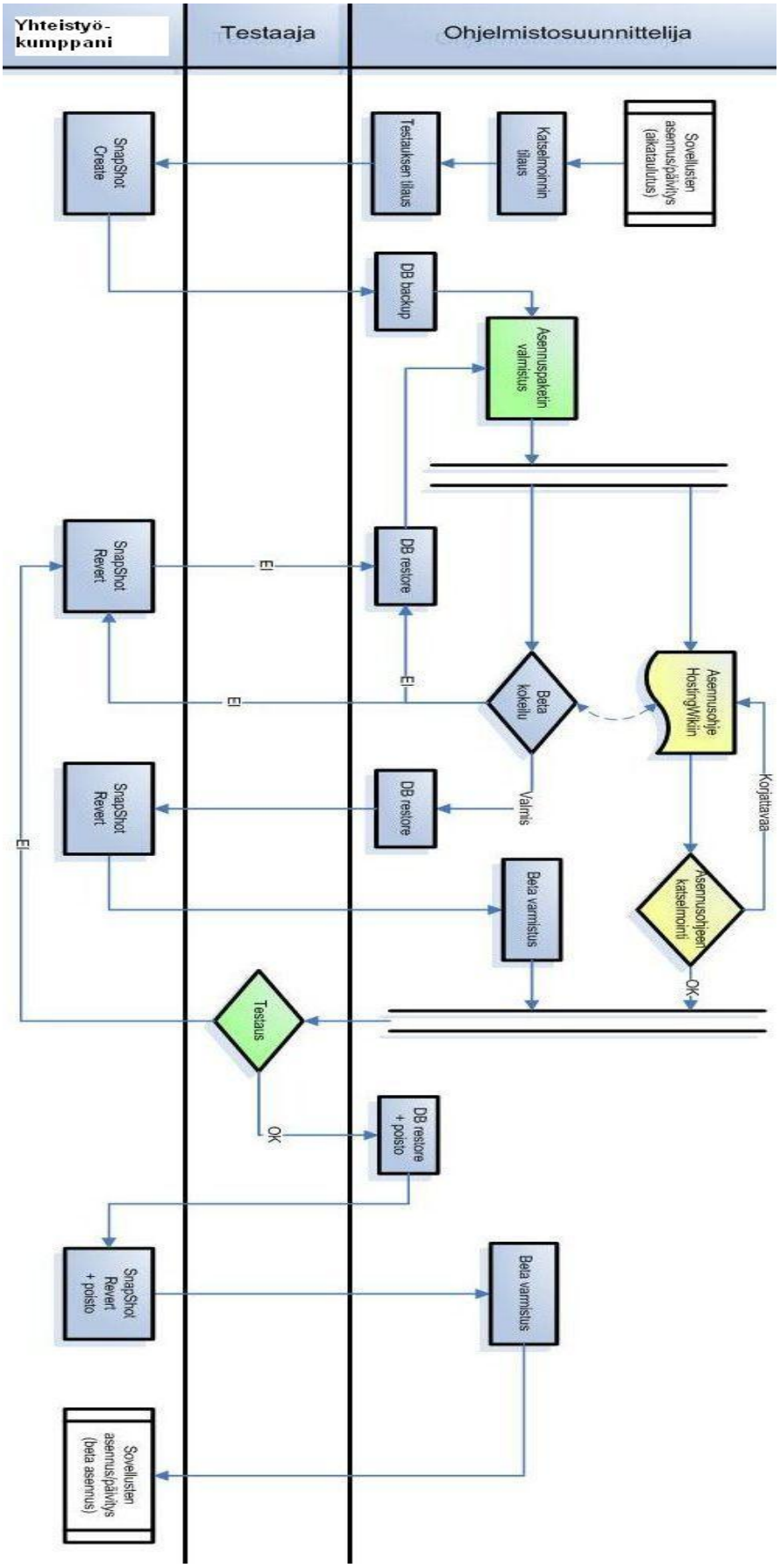
7.1.1.3 Kohdeorganisaation tuotekehitysprosessi: asennuspaketin tekeminen

Lähtötilanne asennuspaketin tekemiselle on, että testipalvelimen pitää olla samassa tilassa kuin tuotantopalvelimen. Tämä on varmistettu sillä, että kohdeorganisaation palvelimia ylläpitävä yhteistyökumppani suorittaa viralliset asennukset tuotanto- ja testipalvelimille. Tietokannoista ja sovelluskoneista ei saa olla olemassa ylimääräisiä varmuuskopioita. Oletusarvoisesti asennuspaketin tekijä on myös tuotantoasennuksen yhdyshenkilö.

Kohdeorganisaation mukaan onnistunut asennuspaketin tekoprosessi alkaa siten, että tilataan asennuspaketille ja asennusohjeelle katselmoija ja testaaja. Asennuspaketin tekijä ei saa itse olla testaajana eikä katselmoijana. Testaus pitää tilata ja sopia mahdollisimman ajoissa, että se saadaan tehtyä ajallaan.

Seuraavaksi testipalvelimen sovelluskoneesta ja tietokannasta otetaan varmuuskopio. Asennuspaketin tekijä kasaa asennuspaketin tarvittavineen tiedostoineen ja kirjoittaa asennusohjeen. Seuraavaksi asennuspaketin tekijä testaa itse asentamalla uuden sovellusversion testipalvelimelle. Jos testiasennus menee läpi, palautetaan tietokanta ja sovelluskone alkuperäiseen tilaan ennen testiasennusta. Jos testiasennus tässä vaiheessa epäonnistuu, palautetaan myös sovelluskone ja tietokanta alkuperäiseen tilaan ennen asennusta. Tätä toistetaan, kunnes asennuspaketti on asennettu onnistuneesti.

Kun asennuspaketin tekijä on onnistunut asentamaan uuden sovellusversion onnistuneesti, asennusohje on kirjoitettu ja sovelluskone ja tietokanta on palautettu alkuperäiseen tilaan. Varsinainen katselmoija ja testaaja suorittavat asennuksen testipalvelimelle. Jos asennus menee läpi, tilataan sovelluksien tuotantopäivitys yhteistyökumppanilta, mikä on sovelluskehitysprosessin viimeinen vaihe. Jos asennus ei mene läpi ja asennusohjeessa on jotakin epäselvää, asennuspaketin tekijä selvittää ongelmaa ja korjaa virheet. Sama kierto toistuu, kunnes asennus on suoritettu onnistuneesti.



KUVIO 11. Asennuspaketin tekeminen. (Yritysx prosessikaaviot.)

7.1.1.4 Kohdeorganisaation tuotekehitysprosessi: sovelluksien asennusprosessi

Sovelluksien asennusprosessi päättää tuotekehitysjakson. Prosessi on määritelty tarkasti, koska Yritysx haluaa varmistaa sovelluksien asennuksen- ja päivityksen oikein sujumisen sekä niiden toimimisen asennuksen jälkeen. Yritysx on tiukka tässä prosessissa ja vaatii, että sovelluskehityksessä työskentelevät ohjelmistosuunnittelijat noudattavat tarkasti tätä prosessia. Lisäksi yrityksen sovelluksia palvelimillaan ylläpitävän yhteistyökumppanin teknisien asiantuntijoiden on myös noudatettava tätä prosessia. Tämä kaava onkin määritelty yhdessä yhteistyökumppanin asiantuntijoiden kanssa helpottamaan sovelluksiin tapahtuvaa päivitystä ja varmistamaan, ettei mitään kriittistä virhettä pääse tapahtumaan. Tällainen vakava virhe voi esimerkiksi olla tietokannan korruptoituminen ja tässä prosessissa onkin huomioitu esimerkiksi varmuuskopioiden ottaminen itse www-koneesta ja tietokannasta. Tällöin voidaan helposti palauttaa alkuperäinen tilanne tuotantopalvelimella, jos jokin menee asennuksessa vikaan. Tällä prosessilla estetään mahdolliset kriittiset virheet sovelluksien päivityksessä ja asennuksessa.

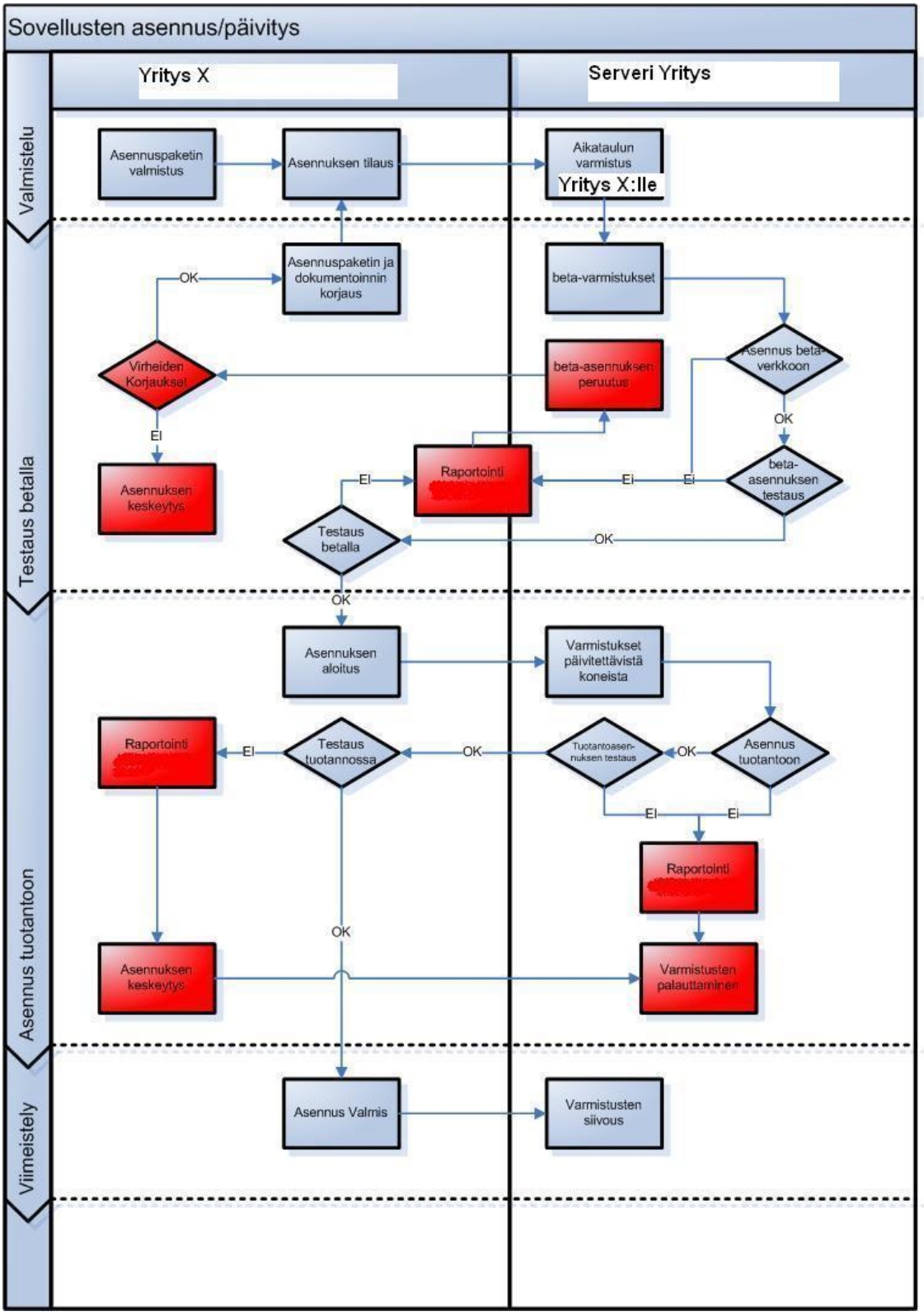
Molemmat osapuolet, sekä Yritysx, että palvelimia ylläpitävä yhteistyökumppani, vaativat että prosessin mukaisesti pitää toimia. Saattaa tapahtua, että Yritysx on toiminut prosessin mukaisesti, mutta yhteistyökumppani ei ole, ja varmuuskopio tietokannasta puuttuu. Tämän jälkeen jostakin syystä asennus epäonnistuu ja aiempaa tilannetta ennen sovelluksen asennusta ei pystytä palauttamaan tuotantopalvelimella yhteistyökumppanin virheen takia, kun varmuuskopioita ei ole otettu. Edellä mainitusta tilanteesta yhteistyökumppani voi joutua vaikeaan asemaan ja korvaamaan Yritysxn asiakkaille menetetyistä tiedoista aiheutuneet tappiot. Tämän prosessin mukaan toimiminen ja sen tarkka noudattaminen ovat molempien osapuolien kannalta tärkeää. Tässä prosessissa ei voi toimia luovasti ja oikoa polkuja esimerkiksi säästääkseen aikaa jossakin prosessin tietyssä vaiheessa. Jos toimitaan muutoin kuin prosessi vaatii, seuraukset voivat olla kohtalokkaat.

Prosessissa lähdetään liikkeelle siitä, että Yritysxn ohjelmistosuunnittelija on koonnut asennuspaketin ja laatinut ohjeen asennusta tai päivitystä varten. Tämän jälkeen lähetetään tilausviesti sovelluksen päivityksestä yhteistyökumppanille. Tilausviestin sisältö on tarkoin määritelty. Se sisältää päivityksen kohteena olevan sovelluksen ja toivo-

tun aikataulun, sekä Yritysxn puolelta yhteyshenkilön, johon yhteistyökumppani ottaa yhteyttä, jos päivityksessä ilmenee jokin ongelma.

Tilausviestin jälkeen yhteistyökumppani vahvistaa Yritysxlle, että tilaus on vastaanotettu ja aikataulu päivitykselle hyväksytty. Tähänkin on määritelty molemminpuoliset vastausajat. Tämän jälkeen yhteistyökumppani ottaa testi- ja tuotantopalvelimista varmuuskopiot sekä ilmoittaa Yritysxlle, että suoritetaan testiasennus. Testiasennuksen jälkeen testataan sovellusta palvelimella ja jos asennus on onnistunut ja sovellus toimii oikein, annetaan lupa asentaa päivitys tuotantoon. Jos testiasennuksessa jokin menee pieleen, korjataan ongelmat ja tarkennetaan asennusohjetta tarvittaessa. Mikäli ongelmia ei pystytä ratkaisemaan, perutaan asennustilaus.

Asennusten jälkeen tehdään testaukset ja lähetetään varmennusviestit onnistumisesta toisille osapuolille ja luodaan raportit. Jos asennus ei onnistu, palautetaan testi- ja tuotantopalvelimet alkuperäiseen tilaan ennen asennusta. (ks. kuvio 12)



KUVIO 12. Sovelluksien asennusprosessi. (Yritysx prosessikaaviot.)

7.1.2 Mitä muutoksia kohdeorganisaation projektiryhmän tuotekehitysprosessi vaatii tullakseen Scrumin mukaiseksi?

Tämän tutkimusongelman analysoiminen suoritetaan vertailemalla kohdeorganisaation toimintaa tuotekehitysprosessissa Scrumin mukaiseen tuotekehitysprosessiin ja käsitteisiin.

Product backlog: Yritysxn toteutuksen suunnittelu dokumentista (ks. kuvio 9) käy ilmi, että ohjelmistotuotteesta on olemassa ominaisuusluettelo. Kohdeorganisaation tuotteen ominaisuuksia priorisoidaan aina viikoittain ja listaa päivitetään sitä mukaa, kun tehtävät ovat valmiita.

Sprint backlog: Yritysxn tuotekehitysprosessin mukaan tietylle tuotekehitysjaksolle, joka on viikon mittainen ajanjakso, listataan tuotteen ominaisuudet erilliselle listalle. Tätä tuotteen ominaisuuslistaa ja projektin tilaa päivitetään sen mukaan, miten ohjelmistosuunnittelijat saavat omaisuudet valmiiksi. Viikkopalavereissa käydään viikoittain läpi, mitä on tehty ja mitä on tekemättä. Viikkopalaveri on myös starttipalaveri seuraavalle työviikolle.

Scrumin ideologia mahdollistaa organisaation johdon päivittäisen ajan tasalla pysymisen, koska päivittäiset seurantapalaverit paljastavat joka päivä ohjelmistoprojektin statuksen (ks. 5 Scrum). Sprintin työtehtävistä myös julkaistaan graafinen tehtävien suoritusta kuvaava kaavio. Tästä kaaviosta voidaan seurata, ylittyykö vai alittuuko sprintin tehtävien suorittamiseen arvioitu aika. Kaksi tai neljä viikkoa kestävän sprintin jälkeen järjestettävät katselmointipalaverit (sprint review meeting) avittavat myös projektin statuksen seuraamista palavereissa, joissa julkaistaan uusi versio ohjelmasta edellyttäen asiakkaan hyväksymistä. Periaatteessa Scrumin pitäisi mahdollistaa jokaisen sprintin jälkeen kaupattava tuote. Kaksi – neljä viikkoa kestävän jakson jälkeen myös sovelluskehitystiimille tarjoutuu mahdollisuus arvioida omaa tekemistä ja kehittyä paremmaksi seuraavalle sprintille. Edellyttäen tietenkin, että sprintista on pidetty kirjanpitoa, mikä on mennyt kehitysjakson aikana huonosti ja hyvin.

Kohdeorganisaation tuotekehitysprosessi etenee palavereissa mitattuna viikoittaisissa sykleissä. Jokaisen syklin jälkeen ei julkaista uutta versiota ohjelmasta tuotantoon. Ohjelma julkaistaan tuotantoon neljä kertaa vuodessa kohdeorganisaation ”sovelluksi-

en toteutusprosessin” ja ”asennuspaketin valmistusprosessin” jälkeen. Tähän vaikuttaa kohdeorganisaation yhteistyökumppani, jolla on ainoastaan pääsyoikeudet tuotantopalvelimille. Jokainen sovelluksen asennuskerta, joka nostaa kustannuksia tutkimuksen kohdeorganisaation tuotekehityksen osalta, jonka tämä sovelluspalvelimia ylläpitävä yhteistyökumppani suorittaa. Sovelluspalvelimia ylläpitävä yhteistyökumppani ja kohdeorganisaatio ovat yhdessä luoneet molempien pakollisesti noudatettavan sovelluksien asennusprosessin, jotta asennuksessa ei tule virheitä. Yhteistyökumppani on luvannut tarjota 24 tuntia vuorokaudessa katkeamatonta palvelua ja tästä syystä vaatii, että vain heillä on tuotantopalvelimiin ja tietokantoihin pääsy. He vaativat myös, että sovelluksien asennukset suoritetaan heidän toimestaan.

Daily Scrum meeting verrattuna tutkimuksen kohdeorganisaation tapaan hoitaa palavereita tuotekehityksen aikana. Kyselylomakkeella oli kysymys palaverien järjestämiseen liittyen: ”Miten palaverit tuotteen kehitykseen liittyen järjestetään projektiryhmän kanssa?” Kysymykseen tutkimuksen kohderyhmä vastasi seuraavasti:

Vastaukset koosteena: Projektin alussa järjestettiin starttipalaveri (kick off). Tuotepäällikkö on yhteyksissä asiakkaan kanssa aina kun mahdollista. Projektiin osallistuvat henkilöt keskustelevat päivittäin keskusteluohjelman ja puhelimen välityksellä. Tuotepäällikkö kutsuu projektiryhmän koolle kerran viikossa ja palaverista tehdään muistio.

Vastauksista voidaan nähdä, että tutkimuksen kohdeorganisaatiossa palavereja järjestetään viikoittain kerran säännöllisesti projektiryhmän sisäisesti. Jos organisaatio haluaisi noudattaa Scrum-menetelmän mukaista palaverointi tapaa, tulisi sen ottaa käyttöön ”Daily Scrum Meeting”-palaverit.

Scrumin mukaisessa ohjelmistoprojektissa projektiryhmä julkaisee työmääriä kuvaavan graafisen kaavion, jota käytetään apuna projektin edistymisen seuraamisessa. Tutkimuksen kohdeorganisaatiossa projektin statusta päivitetään viikoittain kohdeorganisaation ohjelmistoprojektissa. Tämä tapahtuu, kun ohjelmistosuunnittelijoiden viikon aikana tehdyt työt kirjataan tehdyiksi.

Yhteenveto: tutkimuksen kohdeorganisaation tuotekehitysprosessin tuleminen täysin Scrumin mukaiseksi vaatisi sen, että kohdeorganisaatio rupeaa käyttämään ”Daily

Scrum Meeting”-palavereja apuna projektin seuraamisessa. Viikoittainen palaverointi tulisi muuttaa päivittäiseksi.

Kohdeorganisaatio julkaisee ohjelmistotuotteesta neljä kertaa vuodessa uuden version. Scrumissa uusi versio ohjelmasta julkaistaan jokaisen sprintin jälkeen. Scrumin mukainen sprintti kestää kahdesta neljään viikkoa. Tutkimuksen kohdeorganisaation kehitysjakso (sprintti) kestää kolme kuukautta, jos tuotteesta julkaistaan neljä kertaa vuodessa uusi versio.

7.1.3 Onko Scrum sopiva menetelmänä kohdeorganisaation projektiryhmän tuotekehitykseen?

Kappaleen 7.2.2 yhteenvedossa on esitelty, mitä muutoksia kohdeorganisaation tulisi tehdä noudattaakseen Scrum-menetelmän mukaista tuotekehitystä. Nämä muutokset käytiin yhdessä kohdeorganisaation kanssa läpi seuraavin tuloksin.

Daily Scrum meeting: päivittäiset palaverit eivät saaneet kannatusta tutkimuksen kohderyhmän taholta. Kyselylomakkeella oli kysymys, jolla kyselytutkimus suoritettiin: auttavatko päivittäiset palaverit (Daily Scrum meeting) työn suorittamisessa? Tutkimuksen kohdeorganisaation sovelluskehitystiimissä työskentelevät ohjelmistosuunnittelijat pitävät päivittäisiä palavereja turhina ja resursseja syövinä. Heidän mielestään palavereissa istuminen ei tehosta työn tekemistä, vaan jatkuva kommunikaatio projektiryhmän sisällä.

Sen sijaan, että kulutetaan aikaa palavereissa istumiseen, kannustetaan ohjelmistosuunnittelijoita kysymään heti ja kommunikoimaan paljon ongelmatilanteissa. Kyselylomakkeella oli myös kysymys, jossa piti valita, mitkä seuraavista asioista auttavat ohjelmistoprojektissa, jossa olet mukana. Ideana oli se, että ohjelmistosuunnittelijat valitsevat ne kohdat, joista uskovat olevan hyötyä ohjelmistoprojektissaan, ja perustelevat valintansa. Kohtina olivat avoimuus, asioista tiedottaminen ja rohkeus tehdä omia teknisiä ratkaisujaan. Vastaajat eivät arvostaneet rohkeutta tehdä omia teknisiä ratkaisuja. He arvostavatkin sitä, että tehdään yhteisesti sovittuja standardin mukaisia juttuja (ks. 3.8 Standardit ja käytännöt projektissa). Ihmisten omista ratkaisuista tulee vain ongelmia myöhemmin. Yhteistyö ja vuorovaikutus-taitoihin ohjelmistosuunnittelijat vastailivat, että ihmisten oma sooloilu johtaa huonoon lopputulokseen. Tulisikin

ylläpitää jatkuvaa kommunikaatiota ja tiedottamista projektin jäsenien välillä, että pysytään oikeassa asiassa.

Uuden ohjelma version julkaiseminen: Tutkimuksen kohdeorganisaatio julkaisee ohjelmasta uuden version neljä kertaa vuodessa. Tällöin yksi tutkimuksen kohdeorganisaation sprintti kestää kolme kuukautta. Tähän on yhtenä syynä kohdeorganisaation ja sovelluspalvelimia ylläpitävän yhteistyökumppanin määrittelemä asennusprosessi, jolla pyritään estämään virheet tuotantoasennuksissa. Jokainen asennuskerta maksaa kohdeorganisaatiolle. Jos sovelluksesta asennetaan vuoden aikana 12 kertaa päivitys Scrumin mukaan, ylimääräisiä asennuksia tulisi kahdeksan asennuskertaa enemmän. Tällöin nousevat tuotekehityksen kustannukset merkittävästi, kun tutkimuksen kohdeorganisaation ja yhteistyökumppanin määrittelemän asennusprosessin suorittamiseen vaaditaan tutkimuksen kohdeorganisaatiolta vähintään kolmen ihmisen työpanos yhtä asennusta kohti. Yksi työntekijä tekee asennuspaketin, toinen työntekijä testaa asennuksen ja kolmas katselmoi.

Kohdeorganisaatiolta tiedusteltiin: mitkä tekijät ovat johtaneet siihen, että ohjelmistotuotteesta julkaistaan uusi versio neljä kertaa vuodessa. Syitä tähän oli seuraavia:

Aiemmin kaikilla ohjelmistosuunnittelijoilla oli oikeudet päästä tuotantopalvelimille tekemään päivityksiä. Tästä johtuen päivityksien tekeminen ei ollut luotettavaa, koska kuka tahansa pääsi tekemään päivityksiä tuotantoon. Vaarana oli se, että joku tekee nopeasti jonkun muutoksen tuotepäällikön pyynnöstä ja tämän jälkeen selvitellään kauan rikki menneitä ominaisuuksia ohjelmasta. Tiukasti noudatettava sovelluksien asennusprosessi on määritelty myös sen takia, että halutaan sovelluksiin toteutettavat päivitykset ja uudet ominaisuudet suorittaa organisoidusti. On myös tiukasti päätetty käsitellä muutokset ja uudet ominaisuudet projektin ohjausryhmässä, koska projektin laajuus halutaan pitää hallinnassa. Organisaatio ei halua sitä, että muutoksia tehdään hallitsemattomasti ja projektin laajuuden rajaus tulee epäselväksi muutoksista johtuen. Muutokset ovat kuitenkin tervetulleita ja niitä ei pyritäkään estämään, mutta ne pitää käsitellä hallitusti. Toteutettavat ominaisuudet aikataulutetaan ja niistä laaditaan tarjous asiakkaalle. Jos asiakas hyväksyy tarjouksen, muutokset toteutetaan.

Asiakkaat ja sovellustuotteen käyttäjät saattavat tuntea myös ohjelman käyttämisen epämääräiseksi, jos uusia ominaisuuksia lisätään paljon jatkuvasti lyhyen ajan sisällä

tai jonkun toiminnon alkuperäinen logiikka muuttuu. Organisaation tavoite onkin julkaista ominaisuuksia hallitusti, että käyttäjät pysyvät mukana. Käyttäjillehän pitää järjestää koulutus uusista ominaisuuksista, koska muuten he tuskin osaavat hyödyntää niitä. Uusien ominaisuuksien arvostus on myös tärkeää tuotteen elinkaaren näkökulmasta, koska tyytymätön käyttäjä tuskin käyttää niitä. Usein tyytymätön käyttäjä saattaa olla henkilö, joka ei osaa käyttää tuotetta oikein.

Yhteenveto: tutkimuksen kohdeorganisaation johto ei tyrmää Scrumia menetelmänä täysin, vaikka se nostaa tuotekehityksen kustannuksia organisaation määrittelemän sovelluksien asennusprosessin johdosta. Pitäisi jatkotutkia, miten Scrumia pitäisi muuttaa, että se olisi sopiva kohdeorganisaatiolle.

8 POHDINTA

Tämän opinnäytetyön aihe sopii hyvin tietojenkäsittelyn opiskelijoiden opinnäytetyön aiheeksi, koska tässä työssä käsitellään ohjelmisto-projektimenetelmiä ja ohjelmiston tuotekehitysprosessia.

Tämän opinnäytetyön tarkoituksena on kehittää tutkimuksen kohdeorganisaation tuotekehitysprosessia selvittämällä, onko Scrum sopiva kohdeorganisaation tuotekehitykseen. Aiheen rajaaminen saattaa olla vaikeaa tällaisessa opinnäytetyössä, kun taustamateriaalia on todella paljon saatavilla. On myös paljon mielenkiintoisia ohjelmistotuotantoon ja ohjelmistoprojektiin läheisesti liittyviä käsitteitä, joita ei voinut tässä opinnäytetyön yhteydessä käsitellä.

Tämä opinnäytetyö oli luonteeltaan kehittämisprojekti informaatioteknologia-alan organisaation tuotekehitysprosessiin, jolloin tutkimus perustuu pääasiassa tutkijan havaintoihin. Jostakin tutkimus on kuitenkin aloitettava ja yhdeksi tutkimusmenetelmäksi tähän opinnäytetyöhön valittiin kysely. Se suoritettiin kyselylomakkeen avulla internetin kautta. Kysely valittiin tutkimus-menetelmäksi, koska vastausmateriaali tutkimukseen haluttiin kerätä nopeasti ja vaivattomasti. Kysely lähetettiin kahdeksalle ohjelmistosuunnittelijalle ja kahdelle tuotepäällikölle. Kyselyyn vastasi 70 prosenttia tutkimuksen kohderyhmästä. Kohderyhmä oli tarkasti rajattu, koska vain ohjelmistonkehityksen parissa työskentelevät ihmiset oli kelpuutettu mukaan tutkimukseen.

Tähän opinnäytetyöhön oli vaikea valita tutkimusmenetelmää, kun pelkästään kyselyn muodostaminen on jo haastavaa. Haastattelu on myös tutkimus-menetelmänä varsin haastava, koska haastattelijan on kouluttauduttava haastattelutilanteeseen. Haastattelut voivat myös kestää kauan, joten kyselyllä lähdettiin etenemään tutkimuksessa. Tutkimusmenetelmän valinnan vaikeudesta johtuen oli pakko harkita tutkimus-strategiaa tarkemmin. Tutkimusstrategia oli tässä opinnäytetyössä seuraava: ensin suoritettiin taustatiedon hankkiminen, suoritettiin kysely, tutkittiin kohdeorganisaation tuotekehitysprosessidokumentteja ja lopuksi käytiin keskustelua kohdeorganisaation johdon kanssa mahdollisista muutoksista tuotekehitysprosessiin.

Tässä opinnäytetyössä käytettyä tutkimusmenetelmää olisi voinut kehittää tekemällä tutkimus yksityiskohtaisemmin. Tutkimuslomakkeella olisi voitu kysyä myös yksi-

tyiskohtaisemmin ohjelmistotuotteen kehitysprosessista. Liiketoiminnan korostaminen ohjelmistotuotannossa olisi myös saattanut antaa uusia näkökulmia tutkittavaan aiheeseen. Prosesseja ja menetelmiä voidaan aina viilata loputtomasti, mutta ohjelmistotuotanto on yritys-maailmassa liiketoimintaa. Ilman kannattavaa liiketoimintaa informaatio-teknologia-alan yritys tuskin on kauan olemassa. Liiketoiminnan näkökulmasta tätä tutkimusta kannattaisi myös jatkossa kehittää. Tuotekehitys-prosessiin ja siihen valittuihin menetelmiin vaikuttaa myös ohjelmistoprojektin laajuus ja kesto. Näistä näkökulmista tutkimusta voitaisiin kehittää.

Opinnäytetyön yhtenä tuloksena odotettiin tutkimuksen kohdeorganisaation tuotekehitysprosessin raportoimista, jonka selvittäminen onnistui hyvin tässä työssä. Pystyttiin myös selvittämään, mitä yhteneväisyyksiä ja eroavaisuuksia Scrum-menetelmän ja tutkimuksen kohdeorganisaation tuotekehitys-prosessissa on. Tämän opinnäytetyön tuloksena odotettiin myös saatavan tietoa, onko Scrum-menetelmä sopiva tutkimuksen kohdeorganisaation tuotekehitysprosessiin. Tässä opinnäytetyössä kyseiseen tutkimusongelmaan vastauksena saatiin se, että periaatteessa Scrum sopii menetelmänä tutkimuksen kohdeorganisaation tuotekehitykseen. He eivät kuitenkaan halua toteuttaa Scrumia täysin samalla tavalla, kuten se on tämän opinnäytetyön lähdeviitteissä kuvattuna. Tuloksia ei kannata yleistää, koska kyseessä on tietyn organisaation tuotekehitysprosessin kehittäminen. Tämän opinnäytetyön tarjoama informaatio ei välttämättä sovellu yleistettäväksi.

LÄHTEET

Agile Alliance 2002. Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D. Viitattu 10.8.2008. http://www.agilealliance.org, Manifesto for Agile Software Development.

Control Chaos 2008. SCRUM. Viitattu 3.9.2008. http://www.controlchaos.com/.

Extreme Programming 2006. Viitattu 4.9.2008. http://www.extremeprogramming.org/.

Ilkka Haikala ja Jukka Märijärvi 2004. Ohjelmistotuotanto. 10.painos. Karisto Oy, Hämeenlinna 2004.

Melanie G. Silver. Am I, or Am I Not, Using Scrum? That is the Question 18.3.2007. Viitattu 22.9.2008. http://www.scrumalliance.org/, Scrum.

Reactor Innovations 25.10.2006. Viitattu 15.9.2008. http://www.ri.fi/web/fi/teknologia-ja-tutkimus/scrum, Scrum.

Peter Schuh 2005. Integrating Agile Development in the Real World. ISBN:1-58450-364-5. 1. painos. Charles River Media, INC 2005.

Schwaber, K., Beedle, M. 2002. Agile Software Development with Scrum. New Jersey: Prentice Hall.

Scrum Alliance 2007. Scrum. Viitattu 13.8.2008, http://www.scrumalliance.org.

Sirkka Hirsjärvi, Pirkko Remes, Paula Sajavaara. Tutki ja kirjoita. 13. painos. Otavan Kirjapaino Oy Keuruu 2007.

Sirviö, P. 2002. Ketterät menetelmistöt – aidosti asiakkaan asialla, nettiartikkeli HETKY 3/2002. Viitattu 24.4.2007.

Timo Lehtimäki. 2006. Ohjelmistoprojektit käytännössä. 1.painos. Gummerus Kirjapaino Oy, Jyväskylä 2006.

Yritysx. Prosessikaaviot. Viitattu 25.9.2008.

Virtuaali AMK. Ohjelmistotuotanto. Viitattu 1.9.2008. http://tekniikka.ncp.fi/ohjelmistotuotanto/otp/sivusto/materiaalit.html, Jyväskylän ammattikorkeakoulu ja Pohjois – Karjalan ammattikorkeakoulu.