KYMENLAAKSON AMMATTIKORKEAKOULU
Teknologiaosaamisen johtaminen

Mikko Partio

SCRUMIN SOVELTAMINEN OHJELMISTOJEN YLLÄPITOTYÖHÖN

Opinnäytetyö 2015

TIIVISTELMÄ

KYMENLAAKSON AMMATTIKORKEAKOULU
Teknologiaosaamisen johtaminen

PARTIO, MIKKO	Scrumin soveltaminen ohjelmistojen ylläpitotyöhön
Opinnäytetyö	69 sivua
Työn ohjaaja	Paula Posio, Principal Lecturer
Toimeksiantaja	Ilmatieteen laitos
Toukokuu 2015	
Avainsanat	Scrum, tiimityöskentely, ketterät menetelmät, johtaminen, ohjelmistojen ylläpito, käyttöönottoprojekti

Tässä tapaustutkimuksessa selvitettiin kuinka ketterä ohjelmistokehitysmenetelmä Scrum soveltuu ohjelmistojen ylläpitotyöhön. Tutkimuksessa toteutettiin ohjelmiston käyttöönottoprojekti Scrum -menetelmällä, jossa tutkija oli osa työryhmää. Työstä kerättiin kokemuksia noin yhdeksän kuukauden ajan, lopuksi työryhmän suhtautumista uuteen työtapaan kartoitettiin teemahaastattelujen avulla. Tutkimuksessa keskityttiin myös Scrum -työryhmän toimintaan ryhmänä, sekä työtavan johtamiselle asettamiin erityisvaatimuksiin. Tätä varten tutkimuksessa perehdyttiin alan yleisimpään teoriaan, ja reflektoitiin sitä työryhmään ja sen johtamiseen. Aiempia tutkimuksia verrattiin tähän tutkimukseen niiltä osin, miltä se oli mahdollista.

Tutkimuksen tulokset olivat varsin selkeät ja linjassa vertaistutkimusten kanssa. Scrum toimi käyttöönottoprojektissa hyvin sekä yrityksen johdon että työryhmän kokemana. Scrum asettaa kuitenkin tiettyjä ehtoja työryhmälle sekä sen johtamiselle joiden on täytyttävä, jotta työryhmä voi menestyä. Jotta Scrum voisi työtapana toimia, pitää organisaation sitoutua sen sääntöjen noudattamiseen, mutta toisaalta sitä pitää prosessina soveltaa kunkin yrityksen toimintaympäristöön.

ABSTRACT

KYMENLAAKSON AMMATTIKORKEAKOULU
University of Applied Sciences
Degree Programme in Technology Administration

PARTIO, MIKKO	Implementing Scrum in Software Administration
Master's Thesis	69 pages
Supervisor	Paula Posio, Principal Lecturer
Commisioned by	Ilmatieteen laitos
May 2015	
Keywords	Scrum, team dynamics, agile methods, leadership, software administration, software deployment

This case study examines how Scrum, an agile software development methodology, can be used in the framework of software administration. Scrum was used to execute a software deployment project where the researcher was a part of the working group. The practical work period lasted for nine months, during which developer experience was collected. The empirical phase of study was concluded with interviews with the working group where the interviewee could express their feelings towards Scrum. The research also focused on team dynamics and leadership in the context of Scum. This was done by familiarizing one with the common theory of team dynamics, leadership and management and then reflecting that on the working group. Previous research was reviewed and analyzed.

The results of the study were quite clear, and in line with the other studies of the same field. Scrum was concluded to fit well for software administration tasks, as experienced by management and the working group itself. This comes with certain caveats, specifically that the participants (including management) must commit to the process, but also that the process itself must be tailored to fit each individual organization.

TIIVISTELMÄ
ABSTRACT
SISÄLLYS
INDEKSI KUVISTA
1 JOHDANTO....1
1.1 Opinnäytetyön tavoite ja tarkoitus....2
1.2 Käytetyt tutkimusmenetelmät....3
1.3 Tutkimuskohde....4
2 RYHMÄ JA JOHTAMINEN....5
2.1 Ryhmän kehityskaari....6
2.2 Roolit ryhmissä....9
2.3 Ryhmän johtaminen....13
2.4 Kommunikaation merkitys....16
3 OHJELMISTOKEHITYKSEN PROSESSEJA....17
3.1 Ohjelmistokehityksen haasteita....18
3.2 Vesiputousmalli....21
3.3 Ketterä ohjelmistokehitys....24
3.4 Lean-filosofia....25
4 SCRUM....29
4.1 Roolit....30
4.2 Tuotokset....32
4.3 Tapahtumat....34
4.4 Seuranta ja mittarit....38
5 CASE: SCRUM YLLÄPITOYMPÄRISTÖSSÄ....39
5.1 Tavoite ja motivaatio....40
5.2 Scrum ylläpidon työkaluna....41
5.3 Aikataulu....44
5.4 Sprintti 1: 5-6. marraskuuta 2014....45
5.5 Sprintti 2: 17-18. joulukuuta 2014....47
5.6 Sprintti 3: 18-19. helmikuuta 2015....49
5.7 Sprintti 4: 8-9. huhtikuuta 2015....51
6 TULOSTEN TARKASTELU....52
6.1 Scrumin soveltuvuus ryhmälle....53
6.2 Scrum johtamisen kannalta....55

6.3 Kehittäjäkokemukset....58
6.4 Tulosten luotettavuus....63
6.5 Vertaistutkimuksia....64
7 JOHTOPÄÄTÖKSET....67
LÄHTEET
LIITTEET
Liite 1: Teemahaastattelun aiherunko

INDEKSI KUVISTA

Kuva 1: Ryhmäkehityksen malli (Tucker).....7
Kuva 2: Ryhmän sisäiset roolityyppiluokat (Belbin).....12
Kuva 3: Organisatorisen kehitystason mukainen johtaminen (Hersey).....14
Kuva 4: Vesiputousmalli (Royce).....22
Kuva 5: Ketterät ohjelmistokehitysmenetelmät jaoteltuna niiden sitovuuden mukaan.....25
Kuva 6: Ketterien käsitemallien laajuus.....28
Kuva 7: Scrum yleisesti.....29
Kuva 8: Scrumin määrittelemät roolit.....31
Kuva 9: Scrumin tuotokset.....33
Kuva 10: Scrumin tapahtumat.....35
Kuva 11: Kumulatiivinen edistymiskäyrä.....38
Kuva 12: Vauhtikaavio.....39
Kuva 13: Sprinttien aikataulu.....45
Kuva 14: Ensimmäisen sprintin edistymiskäyrä.....47
Kuva 15: Toisen sprintin edistymiskäyrä.....48
Kuva 16: Kolmannen sprintin edistymiskäyrä.....50
Kuva 17: Neljännen sprintin edistymiskäyrä.....51

1 JOHDANTO

Tässä opinnäytetyössä tarkastellaan Scrum-ohjelmistokehitysmenetelmän soveltuvuutta ohjelmistojen ylläpitotyöhön. Scrumin avulla tutkimuksessa toteutettiin keskeisen tietojärjestelmän käyttöönotto. Tutkimuskohteena on Ilmatieteen laitoksen operatiivisen tuotannon ylläpito- ja kehitysryhmä.

Tässä luvussa kerrotaan yleisesti opinnäytetyön motiivista sekä määritellyistä tutkimuskysymyksistä ja -metodeista.

Luvussa kaksi tutustutaan tiimien toimintaan ja niiden johtamiseen. Tiimityöskentely on arkipäivää ohjelmistoprojekteissa ja ryhmän tehokas johtaminen vaatii ymmärrystä ryhmän sisäisestä dynamiikasta sekä rooleista. Lisäksi ryhmän johtajan on pystyttävä muokkaamaan johtamistyyliään ryhmälle sopivaksi. Ryhmän maksimaalisen suorituskyvyn mahdollistamiseksi on johtajan tunnettava ryhmäpsykologian perustava teoria.

Luvussa kolme paneudutaan tarkemmin ohjelmistokehityksen teoriaan. Luvussa käydään läpi ohjelmistokehityksen haasteita, jotka ovat johtaneet uusien käsitemallien syntymiseen. Tarkemmin esitellään sekä niin sanottu vesiputousmalli, joka on ensimmäinen teoreettisesti määritelty malli, sekä myös modernien mallien syntyyn vaikuttaneet filosofiat.

Luvussa neljä keskitytään opinnäytetyön kannalta oleelliseen toimintamalliin, Scrumiin. Scrumin pääpiirteet käydään läpi yksityiskohtaisesti, keskittyen enemmän kuitenkin opinnäytetyön kannalta oleellisiin ominaisuuksiin.

Luvussa viisi päästään varsinaisen case-tapauksen pariin. Luvussa kerrotaan yksityiskohtaisesti toteutettu malli, ja kuinka se erosi oppikirja-Scrumista. Lisäksi kuvataan Scrumin aikana toteutetut käytännön toimenpiteet.

Luvussa kuusi analysoidaan Scrum -projektin aikana kerättyjä tutkimustuloksia ja luvussa seitsemän näistä tuloksista vedetään johtopäätökset.

1.1 Opinnäytetyön tavoite ja tarkoitus

Opinnäytetyön tarkoituksena on selvittää kuinka ketterä ohjelmistokehitysmenetelmä Scrum sopisi täydentämään nykyistä työprosessia ja millaisia vaatimuksia Scrum asettaa ryhmän koostumukselle sekä sen johtamiselle. Tavoitteena on analysoida sekä Scrumin asettamia vaatimuksia, että sitä kuinka ne toteutuvat opinnäytetyön työryhmässä.

Työlle asetetut tutkimuskysymykset ovat:

1. Millaisia edellytyksiä ja vaatimuksia Scrum asettaa ryhmän sisäiselle dynamiikalle sekä ryhmän johtamiselle?
2. Miten Scrum sopii kehitys- ja ylläpitotyötä tekevän ryhmän työkaluksi? Arviointikriteereinä ovat erityisesti henkilöstön kokemukset työtavan tehokkuudesta ja mielekkyydestä.

Tutkimuksessa keskitytään nimenomaisesti Scrum -kehitysmalliin, vaikka myös muita ohjelmistokehitysmalleja esitellään pintapuolisesti. Ryhmädynamiikkaa ja johtamista käsitellään alan teorian kautta. Yrityksessä on aikaisemmin toteutettu vastaavankaltaisia käyttöönottoprojekteja lähinnä perinteisen vesiputousmallin mukaisesti.

1.2 Käytetyt tutkimusmenetelmät

Opinnäytetyö jakautuu edellistä lukua mukaillen kahteen osa-alueeseen:

- ryhmädynamiikkaa ja johtamista käsittelevään osioon, jossa tutustutaan muutamaan alan suosituimpaan teoriaan sekä Scrum-mallin ideologiaan ja reflektoidaan näitä työryhmään
- käytännön osioon, jossa Scrum-kehitysmallia toteutetaan työryhmässä järjestelmän käyttöönottoprojektin yhteydessä, ja tästä kerätään kokemuksia ja tuntemuksia haastatteluiden avulla

Tutkimuksen empiirisessä osiossa toteutettiin uuden ajoketjujen hallintaohjelmiston käyttöönotto. Käyttöönotto tehtiin Scrumia soveltamalla, opinnäytetyön aikana ehdittiin toteuttaa yhteensä neljä iteraatiota (sprinttiä). Työ jatkui opinnäytetyön valmistumisen jälkeenkin.

Laadullinen eli kvalitatiivinen tutkimus pyrkii ymmärtämään tutkittavaa ilmiötä sekä saamaan syvän ja kokonaisvaltaisen kuvan tutkimuskohteesta. Kvalitatiivinen tutkimus usein keskittyy yksilöön ja yksilön kokemuksiin, ja tutkimuksen tuottamat tulokset ovat usein intiimejä luonteeltaan eikä niistä voida suoraan vetää laajempia johtopäätöksiä. Laadullisen tutkimuksen vastakohta on määrällinen eli kvantitatiivinen tutkimus. Tämä tutkimustapa tuottaa numeerisessa muodossa olevaa tietoa, joka voidaan yleistää koskemaan laajempaa ihmisryhmää tai populaatiota.

Tässä tutkimuksessa päädyttiin kvalitatiivisiin tutkimuskeinoihin useasta syystä:

- tutkimusjoukko oli niin pieni, ettei kvantitatiivisilla keinoilla saavutettaisi tarpeeksi luotettavaa tulosta
- pääasiallinen tutkimuskysymys (2) keskittyy nimenomaan laadullisiin seikkoihin

- työn empiirisessä osuudessa toteutettava projekti on ainutlaatuinen eikä vastaavaa ole yrityksessä aiemmin tehty; vanhalla työskentelytavalla tehdyt projektit eivät ole yhteismitallisia Scrumin avulla toteutetun projektin tavoitteiden kanssa, joten vertailua tapojen määrällisestä tehokkuudesta ei voida tehdä

Tietoa testiryhmältä kerättiin koko tutkimusprojektin aikana tapahtuneiden havaintojen perusteella sekä teemahaastatteluin, joita tehtiin sekä ryhmämuotoisesti että henkilökohtaisesti. Haastattelut toteutettiin tutkimuksen loppuvaiheessa, kun työtavasta oli saatu tarpeellinen määrä kokemusta. Teemahaastatteluihin käytettiin aikaa puoli tuntia per haastattelu. Tutkimuksen pohjana on tutkijan pitkä kokemus yrityksen palveluksessa.

1.3 Tutkimuskohde

Työ toteutettiin yhteistyössä Ilmatieteen laitoksen Sääpalvelujen tuotantojärjestelmät-yksikön kanssa. Ilmatieteen laitos on liikenne- ja viestintäministeriön alainen palvelu- ja tutkimuslaitos. Tavoitteena laitoksella on laadukkaan ilmakehästä ja merestä tuotetun havainto- ja tutkimustiedon tuottaminen sekä ihmisten että ympäristön hyvinvointia lisäävien palveluiden tuottaminen. Ilmatieteen laitoksen päätoimipaikka on Helsingissä, ja toimipisteet Kuopiossa, Tampereella, Rovaniemellä, Sodankylässä, Jokioisissa ja Nurmijärvellä. Ilmatieteen laitoksella työskentelee noin 720 henkilöä, jotka ovat ammatiltaan mm. meteorologeja, kemistejä, fyysikoita, matemaatikoita ja insinöörejä.

Sääpalvelujen tuotantojärjestelmät -organisaatioyksikkö kehittää ja ylläpitää koko laitoksen tarvitsemaa IT-infrastruktuuria. Ohjelmistojen kehitys- ja ylläpitovastuu on Tuotantopalvelut -ryhmällä, jonka kanssa yhteistyössä tämä työ tehtiin. Ryhmän vastuulla on säätuotantojärjestelmän sekä siihen liittyvien tietovarastojen ylläpito ja kehitys. Ydintoimintaa ryhmälle on sääennusteiden prosessointi, tallennus ja jakelu, tietokantojen ylläpito ja kehitys, tutka- ja satelliittikuvien vastaanotto ja tallennus sekä

laitoksen yleisten sisäisten työkalujen ylläpito. Ryhmän toiminta on ympärivuorokautista, ja siinä työskentelee kymmenen henkilöä. Ryhmä on monikielinen (suomi, englanti).

Seuraavassa luvussa käsitellään ryhmän muodostumisen, kehittymisen ja roolien teoriaa. Lisäksi käsitellään organisatorisen yksikön johtamisen teoriaa sekä referoidaan kommunikoinnin merkitystä tehokkaassa ryhmätyöskentelyssä.

2 RYHMÄ JA JOHTAMINEN

Suuret ohjelmistoprojektit toteutetaan lähes poikkeuksetta ryhmätyönä. Suurissa ja pienemmissäkin ohjelmistoprojekteissa kehitysryhmän sisäinen toimivuus ja roolijako sekä ryhmän oikeanlainen johtaminen ovat äärimmäisen tärkeitä, jotta ryhmä saadaan suoriutumaan korkeimmalla mahdollisella tasolla. Uudet ketterät ohjelmistokehitysmenetelmät perustuvat itsenäisiin ja funktionaalisiin tiimeihin.

Ryhmien sisäistä dynamiikkaa sekä ryhmän ja yksilön johtamista on tutkittu paljon aina 1960-luvulta lähtien. Tässä luvussa esitellään kolme teoreettista viitekehystä, joita tukena käyttäen Scrum -mallia analysoidaan tarkemmin luvussa 6. Kappaleissa 2.1 ja 2.2 keskitytään tiimin toimintaan, sisäiseen tilaan ja kehitykseen sekä tiimissä esiintyviin rooleihin ja rooliryhmiin. Kappaleessa 2.3 siirrytään tarkastelemaan tiimiä johtamisen kannalta, ja kappaleessa 2.4 kommunikaatiota tiimin toiminnan edellytyksenä.

Alan kirjallisuudessa usein erotellaan ryhmä ja tiimi toisistaan, mutta tässä opinnäytetyössä termit vastaavat toisiaan.

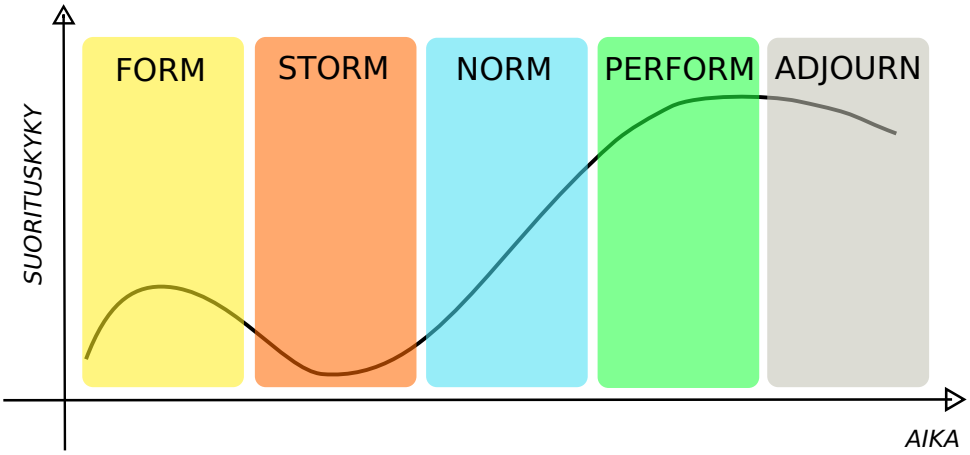
2.1 Ryhmän kehityskaari

Ketterän kehityksen ytimessä on vahva kehitystiimi. Menestyksekkäässä tiimissä on joukko motivoituneita, monenlaista osaamista omaavia toisilleen lojaaleja yksilöitä, jotka yhdessä työskennellen mahdollistavat tiimin toimimisen korkeammalla tasolla kuin sen yksittäiset jäsenet pystyisivät yksinään. Tiimi on siis enemmän kuin osiensa summa.

Vuonna 1965 psykologi Bruce Tuckman esitteli neliportaisen mallin ryhmän sisäisen dynamiikan evoluutiosta. Myöhemmin, vuonna 1977, Tuckman lisäsi malliinsa vielä yhden vaiheen. Yhteensä nämä viisi vaihetta ovat: [1, 2]

- **muodostumisvaihe** (forming)
- **kuohuntavaihe** (storming)
- **normitusvaihe** (norming)
- **suoritusvaihe** (performing)
- **lopetusvaihe** (adjourning)

Kuvassa yksi on esitetty Tuckmanin ryhmän kehitystasot sekä ryhmän suorituskyky kussakin tasossa.



Kuva 1: Ryhmäkehityksen malli (Tucker)

Ensimmäinen vaihe on ryhmän muodostuminen (forming). Tämän vaiheen tunnusmerkkejä ovat ryhmän jäsenten vahva riippuvuus auktoriteetista (ryhmän johtajasta tai ohjaajasta) sekä varovainen tunnustelu ryhmän rakenteesta, rajoista, omasta asemasta ryhmän hierarkiassa sekä muiden ryhmän jäsenten luonteesta. Ensimmäisen vaiheen aikana ei ryhmä juurikaan pysty toteuttamaan tarkoitustaan vaan energia kuluu lähinnä pelisääntöjen muodostamiseen.

Tuckmanin mallin toiseen vaiheen, kuohunnan (storming), tunnusmerkki on ryhmän sisäinen ristiriita. Ryhmän jäsenet vaalivat omaa identiteettiään vastustamalla auktoriteettia sekä ryhmän hierarkiaa. Ryhmä saattaa olla vahvasti polarisoitunut, mikä ilmenee ryhmän näennäisenä jakautumisena, etäisyytenä osapuolien välillä sekä kommunikointiongelmina. Tässäkin vaiheessa suuri osa ryhmän energiasta ja ajasta menee sisäisten ristiriitojen ratkontaan eikä ryhmä vielä toimi ryhmänä vaan joukkona yksilöitä. Ryhmän sisäinen konfrontaatio saattaa nousta niin korkealle, että suorituskyky laskee ensimmäisen vaiheen kohteliaasta kanssakäymisestä.

Kolmas vaihe tunnetaan normituksena (norming), jossa ryhmä saa ratkottua tavalla tai toisella aikaisemmat ristiriidat, ja vaiheen tunnusmerkki onkin yhteenkuuluvuuden tunne. Ryhmän jäsenet jakavat avoimesti tunteensa ja mielipiteensä asioista toisilleen. Ryhmä tuntee, että sillä yhteinen tavoite, ja ryhmän jäsenet tukevat toisiaan tämän päämäärän tavoittamiseksi. Ryhmä saattaa myös kirjoittaa itselleen omat sääntönsä tai koodistonsa. Kolmannessa vaiheessa ryhmän energia alkaa suuntautua kohti sitä työtä jota varten ryhmä on muodostettu; ryhmän sisäiset rakenteen ja suhteet tukevat tätä työtä.

Tuckmanin alkuperäisen vuoden 1965 mallin viimeinen vaihe oli suoritusvaihe (performing). Tässä vaiheessa ryhmä on kypsynyt tilaan, jossa jäsenet pystyvät toimimaan kitkatta keskenään ja kohtelemaan toisiaan objektiivisesti ilman subjektiivista ja emotionaalista latausta. Ryhmä on ehkä löysännyt kolmannessa vaiheessa luotua normistoa: luottamus ryhmän sisällä on korkea eikä eksplisiittistä säännöstöä enää tarvita. Ryhmän jäsenten roolit ovat joustavia ja tehtäväperustaisia. Pääosa energiasta suuntautuu työhön ja ryhmä toimii hyvin pragmaattisesti. Tässä vaiheessa ryhmän suorituskyky on korkeimmillaan.

Viidennen vaiheen, lopetusvaiheen (adjourning), Tuckman lisäsi malliinsa jälkikäteen vuonna 1977. Ryhmä joutuu lopetusvaiheeseen, kun se on täyttänyt tarkoituksensa, eikä sen toiminnan jatkamiselle ole enää perusteita. Jäsenille ryhmän lopetus tuottaa negatiivisia tuntemuksia: ahdistusta, vetäytymistä, surullisuutta. Ryhmän suorituskyky luonnollisesti laskee, kun toimintaa ajetaan alas.

Ketterien kehitysmenetelmien kannalta toimiva ryhmä on normitus- tai suoritusvaiheessa. Mikäli ryhmä on vasta perustettu eikä se ole vielä asettunut, ei voida olettaa Scrumin tai muiden kehitysmenetelmien olevan kovin tehokkaita, ja tässä vaiheessa esimiehelle olisikin tärkeämpää ryhmän sisäisen dynamiikan kehityksen edistäminen (englanniksi *team building*).

2.2 Roolit ryhmissä

Kaikki ryhmät rakentuvat yksilöistä. Ryhmät usein kootaan yksilöiden osaamisprofiilien mukaisesti, niin että ryhmällä on kokonaisuutena osaaminen kaikissa tarvittavissa osa-alueissa. Tässä jaottelussa ei kuitenkaan usein oteta huomioon kunkin yksilön roolia tiimin *sisällä*. Osittain se on mahdotontakin sillä vaikka henkilön osaamisprofiili on usein hyvin staattinen, saattaa hänen roolinsa eri tiimeissä vaihdella tiimien välillä sekä ajallisesti, yksilön osaamisen ja kokemuksen karttuessa.

Kukin yksilö on interaktiossa muun ryhmän kanssa omien luonteenpiirteidensä mukaisesti, ja tästä muodostuu myös henkilön rooli tiimin sisällä. Tiimin vetäjän on hyvä tunnistaa nämä roolit ja pyrkiä vahvistamaan suotuisia rooleja ja heikentämään epäsuotuisia, jotka eivät ole hedelmällisiä ryhmän kehityksen ja suorituskyvyn kannalta.

Yksi tunnetuimmista tiimiroolien luokitteluista on R Meredith Belbinin esittelemä yhdeksänkohtainen malli. [3] Useimmat näistä rooleista ovat osittain yhteensopivia toistensa kanssa ja yksi henkilö pystyy helposti hallitsemaan monta roolia samanaikaisesti; toisaalta jotkut roolit ovat selkeästi poissulkevia eli niiden omaaja voi toimia yhdessä ryhmässä vain siinä kyseisessä roolissa. Täten Belbinin ideaaliryhmä, jossa esiintyy kaikkia yhdeksää roolia, voidaan käytännössä toteuttaa neljän - viiden hengen ryhmissä.

Belbinin malliin kuuluvat seuraavat roolit:

- **innovoija** (plant)

  Innovoija on henkilö, joka pystyy luomaan uutta epätavallisin tavoin, hakemaan ratkaisuja vaikeisiinkin ongelmiin ja löytämään uusia yhteyksiä eri kokonaisuuksien välille. Hän pystyy tarjoamaan tiimille täysin mullistavia ja innoittavia ideoita, mutta työskentelee usein yksin eikä välttämättä osallistu idean käytännön toteutukseen. Innovoija ei välttämättä saa vastakaikua

ideoilleen ryhmän käytännönläheisemmän roolin omaavilta henkilöiltä, mikä saattaa herättää sisäisiä ristiriitoja henkilöiden välille.

- **verkostoituja** (resource investigator)
  Verkostoituja on ekstrovertti, jonka kontaktilista on laaja. Hän pystyy löytämään tiimin tarvitsemat ulkopuoliset resurssit. Verkostoituja käyttää paljon aikaa tiimin ulkopuolella eikä ole täten aina tiimin käytettävissä. Verkostoituja on luonteeltaan helposti innostuva mutta ailahteleva.

- **mahdollistaja** (co-ordinator)
  Tiimin mahdollistaja voi olla samalla sen johtaja, mutta yhtä hyvin roolin omaava voi toimia tiimin jäsenenä. Mahdollistaja työskentelee tiimin edun eteen ja koordinoi päätöksentekoa niin että kaikki tiimin jäsenet saavat sanoa sanottavansa. Mahdollistaja on usein konsensushakuinen henkilö mikä saattaa ryhmän jakavissa kysymyksissä viedä tehon hänen työtavaltaan. Mahdollistajan mielenkiinto on työn organisoinnissa, ja hän jättääkin substanssityön suunnittelun ja toteutuksen muiden tehtäväksi.

- **vauhdittaja** (shaper)
  Vauhdittaja toimii ryhmän katalysaattorina ja työskentelee hyvin kovassakin työpaineessa. Hänellä on korkea energiataso, joka tarttuu usein myös muihin ryhmän jäseniin. Vauhdittaja on kärsimätön jos asiat eivät etene odotetulla tavalla ja vastoinkäymisissä siirtää syyn usein muun ryhmän niskoille.

- **analysoija** (monitor-evaluator)
  Analysoijan tärkein tehtävä on toimia ryhmän jarruna silloin, kun mennään liian lujaa. Analysoija pystyy katsomaan asioita objektiivisesti ja tekemään harkittuja päätöksiä, vaikka kiire olisi kovakin. Analysoijan heikkous on hänen aiheuttamansa hitausmomentti, mikä saattaa turhauttaa muita ryhmän jäseniä ja vaikuttaa negatiivisesti työlle asetettuihin aikatauluihin.

- **sovittelija** (team worker)

  Sovittelija on ryhmän ihmisten välisten suhteiden erikoisosaaja, joka on kiinnostunut eniten ryhmän sisäisestä dynamiikasta ja henkilöiden välisistä suhteista. Hän toimii välimiehenä ryhmän sisäisissä riidoissa, kannustaa ryhmän muita jäseniä ja on ylipäätään sosiaalisesti hyvin taitava. Sovittelija kuitenkin karttaa ristiriitatilanteita ja tuntee olonsa ulkopuoliseksi ryhmissä jotka ylikorostavat substanssin osaamista.

- **toteuttaja** (implementer)

  Toteuttaja on henkilö joka tekee sen raa'an työn, jota varten ryhmä on perustettu. Hän nauttii työmyyrän roolistaan ja pitäytyy tiukasti realismissa, antamatta ajatusten lentää sfääreihin. Toteuttaja on usein näkymättömämpi kuin muut ryhmän jäsenet, sillä hän ei korosta omaa rooliaan. Hänen pragmatisminsa ja kykenemättömyytensä korkealentoiseen ideointiin saattaa turhauttaa ryhmän muita jäseniä silloin kun ryhmä kohtaa ongelman, jonka ratkaiseminen vaatii uuden luomista.

- **viimeistelijä** (finisher)

  Viimeistelijä vastaa ryhmän toiminnan laadusta ja aikataulusta. Viimeistelijä pyrkii toimimaan ennalta määrätyn prosessin mukaisesti eikä pidä, jos säännöistä poiketaan. Viimeistelijä on kiinnostunut yksityiskohdista, mutta saattaa takertua niihin liiallisestikin muiden ryhmän jäsenten harmiksi.

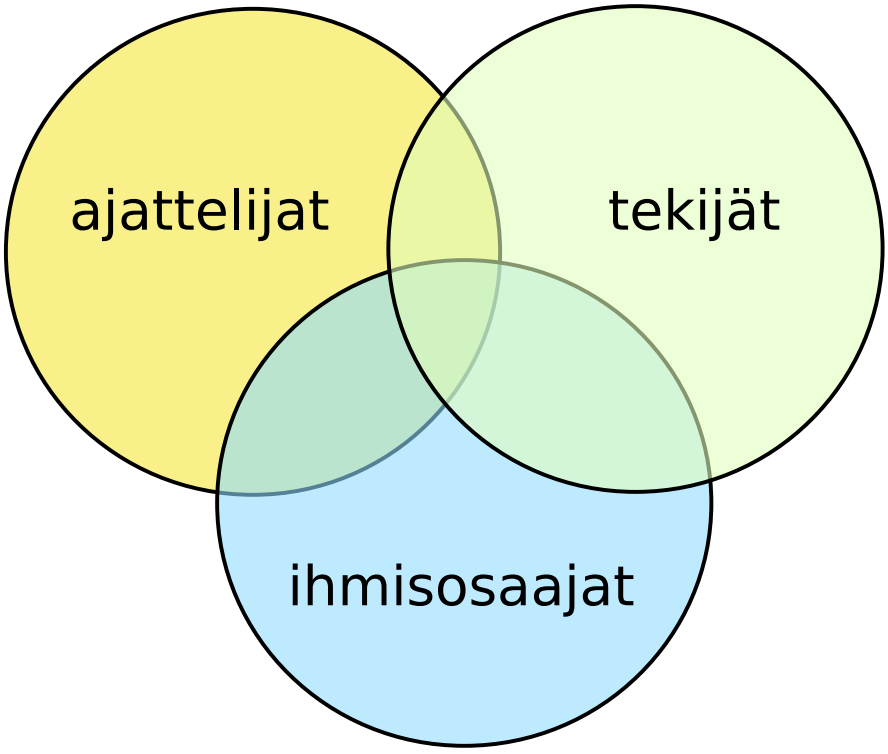
- **erikoisosaaja** (specialist)

  Erikoisosaaja omaa jonkun tietyn kapeahkon sektorin syväosaamisen ja tuo tämän osaamisen tiimin käyttöön. Hän rakentaa itsetuntoaan oman osaamisensa kautta ja on hyvin omistautunut asialleen. Erikoisosaaja ei useinkaan halua nähdä kokonaisuutta ja saattaa olla mustasukkainen omasta vastuualueestaan ja verkostostaan. Erikoisosaajan rooli voi olla myös ryhmää konsultoiva, jossa hänen ammattitaitoaan tarvitaan harvoissa mutta merkityksiltään tärkeissä tilanteissa.

Nämä yllä kuvatut yhdeksän roolia voidaan jakaa kolmeen arkkityyppiin, jotka ovat

- **ajattelijat** (innovoija, erikoisosaaja, analysoija)
- **tekijät** (vauhdittaja, toteuttaja, viimeistelijä)
- **ihmisosaajat** (mahdollistaja, sovittelija, verkostoituja)

Kuvassa kaksi on kuvattu arkkityyppien keskinäinen päällekkäisyys. Tiimin yleinen suorituskyky ja hyvinvointi ovat korkeimmillaan silloin, kun kaikkien kolmen osa-alueen edustajat ovat tasapainoisesti edustettuina. Käytännössä jo se, että ryhmässä on kunkin osa-alueen edustaja edistää ryhmän kehitystä, ryhmähengen muodostumista sekä ryhmän tavoitteiden saavuttamista.



Kuva 2: Ryhmän sisäiset roolityyppiluokat (Belbin)

Ryhmän sisäiset roolit ja niiden tunnistaminen ovat sekä ryhmän johtajan työkalu, että myös erittäin hyödyllinen taito ryhmän jäsenelle. Oman roolin tunnistaminen helpottaa omien vahvuuksien hyödyntämistä ja heikkouksien tunnistamista. Ryhmän johtajan näkökulmasta on hyvä tunnistaa ryhmän jäsenten roolit, sillä ne usein heijastuvat suoraan ryhmähenkeen ja suorituskykyyn.

Ryhmä, jossa on pelkästään suorittavia henkilöitä saa aluksi tehtävät tehtyä, mutta ei pysty nousemaan korkeammalle suoritustasolle, koska ryhmähenkeä ei muodostu. Ryhmä, josta puuttuvat ideoijat pystyy suoriutumaan normaaleista työtehtävistään hyvällä tasolla, mutta odottamattomat ongelmat saattavat pysäyttää koko ryhmän toiminnan. Ryhmä ilman tekijöitä jää pelkäksi keskustelukerhoksi.

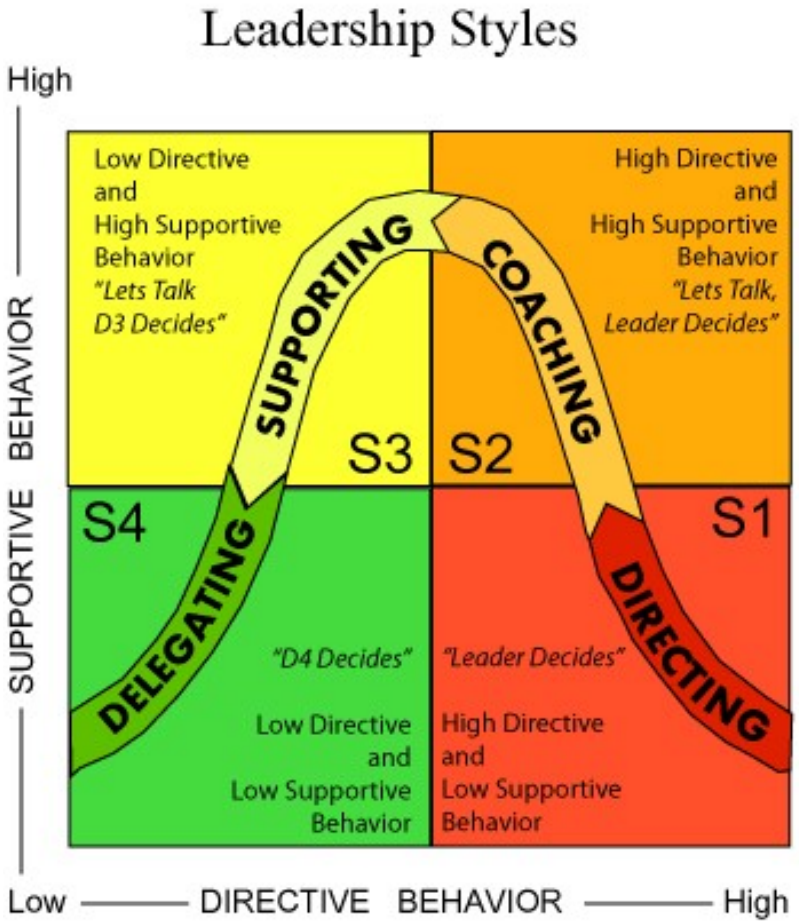
2.3 Ryhmän johtaminen

Jokainen ryhmän kehitysvaihe vaatii omanlaisensa johtamistavan. Mikäli ryhmä ja sen vetäjä eivät ole synkronissa, on vaarana, että ryhmän kehityskaari ei saavuta huippuaan eikä ryhmä ikinä pysty suoriutumaan sillä tasolla, joka sille saattaisi olla mahdollista. Ryhmän johtamisessa tulee siis keskittyä ryhmän sisäisen dynamiikan kehittämiseen ja ryhmän yhtenäisyyden luomiseen.

Hersey ja Blanchard julkaisivat vuonna 1972 teorian ihmisen organisatorisesta kehitystasosta ja siitä, miten se tulisi ottaa huomioon henkilöä johdettaessa. Teoria perustuu aikaisempiin tutkimuksiin,ja siinä johdettavan henkilön tai ryhmän kehitystaso voidaan jakaa neljään eri vaiheeseen. Näin saadaan yhteensä neljä erilaista johtamismallia, jotka ovat järjestyksessä vähiten kehittyneestä eniten kehittyneeseen: [4, 5]

- **saneleva** (directing); johtamistyyli korkea tehtäväkeskeisyys, alhainen ihmiskeskeisyys
- **myyvä** (coaching); johtamistyyli korkea tehtäväkeskeisyys, korkea ihmiskeskeisyys
- **osallistuva** (supporting); johtamistyyli alhainen tehtäväkeskeisyys, korkea ihmiskeskeisyys
- **delegoiva** (delegating); johtamistyyli alhainen tehtäväkeskeisyys, alhainen ihmiskeskeisyys

Kuvassa kolme on esitetty neljä johtamismallia, jotka on jaoteltu kahdelle akselille. On huomattava, että mikään neljästä mallista ei ole paras; jokaiselle mallille on oma käyttötapauksensa riippuen johdettavien kehitystasosta. Johtamistyyli siis mukautuu johdettaviin eikä toisinpäin.



Kuva 3: Organisatorisen kehitystason mukainen johtaminen (Hersey)

Sanelevalla johtajalla (kuvassa S1) on korkea tehtäväkeskeisyys ja matala ihmiskeskeisyys, tarkoittaen, että johtaminen keskittyy vahvasti työhön ja sen sisältöön. Johtamisen kommunikaatio on yksisuuntaista: johtaja kertoo, mitä toivoo tehtävän ja työntekijä toimii ohjeiden mukaan. Johtaja tekee kaikki päätökset ja ilmoittaa päätöksistä alaisille, työntekijä on haluton tai kyvytön sitoutumaan työhön eikä halua ottaa vastuuta itselleen. Saneleva johtamistyyli on toimiva aloilla, joilla työvoima on vähän koulutettua ja työ on yksinkertaista ja suorittavaa (esimerkiksi

perinteinen tehdastyö) tai silloin, kun työntekijä on epävarma ja tarvitsee paljon tukea työssään (esimerkiksi vasta rekrytoitu henkilö, jolla ei ole aiempaa työkokemusta).

Kehitystasossa seuraava johtamismalli on myyvä (S2). Myyvässä tyylissä sekä tehtäväkeskeisyys että ihmiskeskeisyys ovat korkealla tasolla. Työntekijä ei pysty ottamaan kokonaisvastuuta tehtävästä työstä, mutta on halukas osallistumaan osaamisensa puitteissa. Kuten sanelevassa tyylissä johtaja edelleenkin vahvasti ohjaa ryhmän tai yksilön toimintaa, mutta sanelevasta tyylistä poikkeavasti pyrkii kuuntelemaan työntekijää. Työntekijän motivaatiota ylläpidetään osallistamalla tätä työtehtävien valintaan ja suunnitteluun. Taas on huomioitava, että johtamismalli seuraa johdettavan henkilön kehittymistä eli johtamismallia voidaan vaihtaa vasta henkilön siirtyessä kypsyystasolta toiselle.

Kolmas johtamistyyli on osallistuva johtaminen (S3), jonka tunnusmerkkejä ovat alhainen tehtäväkeskeisyys ja korkea ihmiskeskeisyys. Johtaja jakaa päätösvaltaansa työntekijän kanssa ja kohdistaa energiansa tämän tukemiseen työn tekemisessä. Työntekijä omaa riittävän asiantuntemuksen substanssityöstä, mutta on epävarma tai arka ottamaan vastuuta työstä. Jos tarkastellaan lähemmin osallistuvaa ja sanelevaa johtamistyyliä, voidaan havaita niiden vaikuttavan ryhmän kehitykseen eri tavoin. Osallistuvan johtamistyylin on osoitettu vaikuttavan heterogeenisen ryhmän reflektointikykyyn positiivisesti. Heterogeenisellä ryhmällä tarkoitetaan tässä yhteydessä ryhmää, jonka jäsenten ammatilliset taustat ovat erilaisia. Reflektointikyvyn parantuessa ryhmä pystyy kehittämään itseään havaitessaan vajavaisuuksia tai puutteita. Samalla ryhmän innovaatiokyvyn todettiin kasvavan.

Jos ryhmä on homogeeninen – sen jäsenten ammatilliset taustat ovat samankaltaiset – todettiin sanelevan johtajan pystyvän parantamaan ryhmän reflektointia osallistuvaa paremmin. Samanlaisen taustan omaavat henkilöt asettuvat ryhmässä helposti heille sosiaalisesti odotetuille paikoille ja ovat vähemmän valmiita ryhmän kriittiseen tutkimiseen. Samalla kuitenkin todettiin, että osallistuvalla johtamisella oli negatiivinen vaikutus ryhmän suorituskykyyn, verrattuna sanelevaan johtamiseen. Sanelevalla johtamisella saatiin ryhmien suorituskyky kasvamaan mutta samalla

innovatiivisuus oli laskevalla trendillä. Todellisuudessa nämä kaksi johtamistyyliä eivät siis ole toisensa poissulkevia vaan enemmänkin täydentäviä. [6]

Viimeinen ja kehittynein johtamistyyleistä on delegoiva johtaminen (S4), jossa sekä tehtäväkeskeisyys että ihmiskeskeisyys ovat alhaisia. Työntekijällä on vahva osaaminen työstään sekä korkea itseluottamus, joka mahdollistaa vastuun ottamisen ja itsenäisen työskentelyn. Johtaja jakaa osan päätösvallastaan henkilölle tai ryhmälle, ja osallistuu jokapäiväiseen työhön enemmän vertaisena kuin esimiehenä. Esimies pystyy käyttämään aikaansa ryhmän kehitykseen ja suurempien linjojen ja strategioiden miettimiseen ja toteuttamiseen.

2.4 Kommunikaation merkitys

Menestyksekkään tiimin menestyksen analysoiminen on monimutkaista. Menestykseen vaikuttavia tekijöitä on paljon: ryhmän jäsenet ja heidän henkilökohtainen elämäntilanteensa, johtaminen, asetetut tavoitteet, organisaation tuki, työn luonne. Yksi keskeisimmistä menestyksen mahdollistajista on kuitenkin tehokas kommunikaatio.

Alex Pentland MIT:stä työryhmineen on tutkinut kommunikaatiota ryhmien sisällä ja varsinkin sitä, kuinka kommunikaatio on jakautunut ryhmän jäsenten välillä erilaisissa ryhmissä ja kuinka se on vaikuttanut ryhmä suorituskykyyn. Tutkimustuloksista on pystytty nostamaan esille seuraavat viisi menestyksekkään ryhmän kommunikoinnin tunnusmerkkiä: [7]

- kaikki ryhmän jäsenet puhuvat ja kuuntelevat; kukaan ryhmään jäsen ei dominoi keskusteluja ja palavereja
- ryhmän jäsenet pystyvät kommunikoimaan energisesti ja kasvokkain
- ryhmän jäsenet kommunikoivat suoraan keskenään käyttämättä ryhmän johtajaa välittäjänä

- ryhmän sisällä käydään keskusteluja myös virallisten palaverien ulkopuolella
- ryhmän jäsenillä on kontakteja myös ryhmän ulkopuolelle

Näihin viiteen kommunikaation avaintekijään vaikuttaa merkittävästi kommunikoinnin energia ja sen jakautuminen ryhmän jäsenten kesken sekä ryhmien välinen kommunikointi. Pentland ja kumppanit toteavat, että mikäli kommunikaatio on matalaenergistä (vähäistä tai ei-innostavaa) tai mikäli se on jakautunut vain muutaman ryhmän yksilön kesken (ryhmä ryhmän sisällä), viestintä ei ole toimivaa, mikä herättää yleisen tunteen viestinnän epäonnistumisesta.

Toinen Pentlandin esiin nostama mielenkiintoinen havainto liittyy ryhmien väliseen kommunikointiin: ne ryhmät, joiden jäsenet osallistuivat aktiivisesti ryhmän ulkopuoliseen toimintaan ja toivat uusia ideoita takaisin omaan ryhmäänsä, omasivat huomattavasti tehokkaamman ja tasapuolisemman viestintäkentän.

3 OHJELMISTOKEHITYKSEN PROSESSEJA

Yhteiskunnan digitalisoituessa tietojärjestelmien merkitys kasvaa entisestään. Lähes kaikki tekniset laitteet jääkaapista ydinvoimalaan ovat jollakin ohjelmistolla ohjattavissa. Toiminnallisuuden lisääntyessä ohjelmistoista tulee yhä monimutkaisempia mikä johtaa hyvin nopeasti suuriin haasteisiin ohjelmistokehityksessä. Ohjelmistosta tulee liian suuri kokonaisuus yhdelle henkilölle tai tiimille käsitettäväksi, mikä johtaa kehitystyön pirstaloitumiseen. Tässä luvussa kuvataan tarkemmin minkälaisia ongelmia suomalaiset ohjelmistokehitystalot ja heidän asiakkaansa kokevat, sekä millaisia filosofioita ja viitekehyksiä ohjelmistojen kehityksen tueksi on olemassa.

Kappaleessa 3.1 kuvataan ohjelmistokehityksen haasteita, jotka ovat johtaneet uusien mallien ja prosessien kehitykseen ja käyttöönottoon. Kappaleessa 3.2 esitellään

vesiputousmalli, joka on ensimmäinen ohjelmistokehitysmalli ja jota käytetään yleisesti edelleenkin. Kappaleissa 3.3 ja 3.4 esitellään Agile- ja Lean-filosofioiden perusteet. Nämä filosofiat ovat erittäin vahvasti vaikuttaneet moderneihin ohjelmistokehityksen malleihin, joten näiden mallien syvällisempi ymmärtäminen vaatii myös taustatiedon omaamisen. Tämän opinnäytetyön kannalta oleellisimmalle mallille Scrumille on omistettu kokonaan oma lukunsa 4.

3.1 Ohjelmistokehityksen haasteita

Miksi ohjelmistokehitys on niin vaikeaa? Mediaa seuraamalla päätyy helposti johtopäätökseen, että mikään viime vuosien laaja ohjelmistoprojekti ei ole onnistunut. Sosiaali- ja terveysalan sekä yleisestikin kunta-alan tietojärjestelmät ovat auttamattoman pirstaleiset, e-reseptin ja sähköisen äänestysjärjestelmän osittaisetkin käyttöönotot joko kestivät vuositolkulla tai epäonnistuivat täysin ja valtion monopoliyhtiö VR ei saa verkkosivujaan toimimaan, vaikka apuna on Suomen suurimmat ohjelmistoyritykset valtavine resursseineen. Kyse ei voi olla ohjelmointitaidon puutteesta, mutta mistä sitten?

Erään tutkimuksen mukaan vain 45% ohjelmistotyön tilaajista koki onnistuneensa hankinnoissa hyvin. Aiheesta tehtyjen tutkimusten pohjalta esiin nousee kaksi pääsyytä hankkeiden epäonnistumiseen: toimimaton kommunikointi tilaajan ja toimittajan välillä sekä työn ja sen tulosten vääränlainen määrittely. [8]

Viestintähaasteet

Tilaajan ja toimittajan välisen onnistuneen viestinnän merkitystä tietojärjestelmähankkeissa ei voi aliarvioida. Samalla se on osa-alue, joka *Tietojärjestelmien hankinta Suomessa 2013* -tutkimuksen mukaan aiheuttaa eniten ongelmia hankkeiden osapuolien välille. Toimittajaosapuolen mukaan se on suurin yksittäinen syy, joka johtaa projektin kriisiytymiseen. Kommunikaatio-ongelmista on

tunnistettu kommunikaation puute sekä eriävät näkemykset osapuolten välillä. Sama ongelma on havaittu kansainvälisesti jo vuonna 1995, jolloin *The Standish Group* julkaisi raportin, jossa todetaan käyttäjän (tilaajan) aktivoinnin olevan suurin yksittäinen tekijä joka johtaa tietojärjestelmäprojektin onnistumiseen, ja toisaalta suurimman syyn epäonnistumiseen olevan liian vähäisen kommunikoinnin tilaajan ja toimittajan välillä. [9]

Kommunikaation puute on systemaattista, koska sen juuret sijaitsevat itse ohjelmistokehitysprosessissa. Perinteisessä vesiputousmallissa tilaajan ja/tai käyttäjän palaute prosessiin on mahdollista vain alun analyysivaiheessa, kun tietoa kerätään tulevaa ohjelmistoa varten sekä operatiivisessa vaiheessa, kun tuote on valmis käyttöön. Mikäli käyttäjä haluaa muuttaa jotain ohjelmiston osaa, tulee koko kehitysprosessi käynnistää alusta ja palautteen pohjalta tehtyjen muutosten saapuminen operatiiviseen tuotteeseen saattaa isoissa ohjelmistoprojekteissa kestää vuosia. Tämän perinteisen lineaarisen kehitysmallin ongelmia on pyritty ratkaisemaan nykyaikaisemmissa ketterissä kehitysmenetelmissä kuten Kanban tai Scrum, sekä DevOps- ajattelumallissa, jossa yhdistetään yrityksen kehitys- ja operatiiviset henkilöstöt yhdeksi ryhmäksi. Kaikissa tavoissa pyritään aktivoimaan tilaajaosapuoli ottamalla tämä mukaan kehitykseen sekä määrittämään selkeästi ja eksplisiittisesti mitkä ovat ne komponentit, mitä kehitystyöllä tuotetaan. [10]

Työn ja tulosten määrittely

The Standish Groupin tekemän tutkimuksen mukaan tärkeysjärjestyksessään toinen asia, joka mahdollistaa tietojärjestelmäprojektin onnistumisen – tai epäonnistumisen – on vaatimusten tarkka määrittely. *Tietojärjestelmien hankinta Suomessa 2013* -tutkimuksessa käsiteltiin myös määrittelyprosessia sekä toimittajan että tilaajan näkökulmasta. Tulokset olivat ristiriitaisia ja mielenkiintoisia: sekä tilaaja että toimittaja olivat sitä mieltä, että he tekevät suurimman osan tietojärjestelmähankintaan tai -kehitykseen liittyvistä määrityksistä.

Perinteisessä ohjelmistokehitysprosessissa määritysten tekeminen on ehkä tärkein osa koko prosessia. Tässä tavassa ohjelmiston määrittelyt tehdään jo prosessin alkuvaiheessa eikä niihin enää jotain poikkeuksia lukuun ottamatta palata myöhemmin. Tämä niin kutsuttu vesiputousmalli on siis hyvin jäykkä ja lineaarinen tässä suhteessa.

Yksi aspekti tietojärjestelmähankkeiden määrittelyssä on onnistumisen määrittely. *Tietojärjestelmien hankinta Suomessa 2013* -tutkimuksen mukaan tilaajat pitävät tietojärjestelmähankkeen tärkeimpänä arvona sen merkitystä liiketoiminnalle. Tätä mitataan usein aikataulussa ja budjetissa pysymisellä. Nämä mittarit ovat selkeitä ja helppokäyttöisiä mutta ne eivät mittaa kaikkein tärkeintä asiaa: kuinka uusi järjestelmä soveltuu tilaajan käyttöön. Valmis järjestelmä on hyödyllinen, vain jos se oikeasti täyttää tilaajan tarpeet.

Ohjelmistokehityksen arki ja kokemus on osoittanut, että yhtään suuremmissa hankkeissa ei lopullisen tuotteen (ohjelmiston) ominaisuuksia pystytä määrittämään riittävällä tarkkuudella ennen itse kehityksen aloittamista. Käytännössä on usein niin, että alussa tilaaja esittää hyvinkin suuripiirteisiä toivomuksia tuotteelle, ja tarkempi määrittely tapahtuu kehitysprosessin aikana, kun asiakas näkee demoversiota tuotteesta ja huomaa tarvitsevansa sellaisia ominaisuuksia, joita ei aiemmin tullut ajatelleeksi. Varsinkin ohjelmistoja, joissa on rikas graafinen käyttöliittymä, joudutaan usein määrittelemään ja uudelleenkehittämään monta kertaa, ennen kuin saadaan käyttäjiä tyydyttävä versio.

Tämä arjen realiteetti on niin kutsuttujen ketterien ohjelmistokehitysmallien taustalla. Ketterillä ohjelmistokehitysmalleilla pyritään ratkaisemaan kaikki yllä mainitut tilaajan ja toimittajan väliset ongelmat:

- iteraatioihin perustuva toimintamalli mahdollistaa muutosten tekemisen koko kehitysprosessin aikana
- asiakkaan edustaja on aktiivisesti mukana kehitysprojektissa, mikä parantaa kommunikaatiota

- asiakkaan edustaja saa priorisoida kehityskohteita, jolla parannetaan liiketoiminnalle kertyvää arvoa
- tuotteesta julkaistaan useita kehitysversioita, jotka esitellään tilaajalle ja/tai käyttäjille, jolloin kehitysprosessiin saadaan käyttäjäpalautetta jo varhaisessa vaiheessa.

Ohjelmistokehityksen avuksi on kehitetty useita malleja joista varhaisimmat periytyvät 1960-luvulta, ketterien menetelmien vallankumous alkoi 1990-luvulla. Keskeistä metodiikkaa on myös haettu kovan teollisuuden puolelta; ketterien ohjelmistometodien keskeinen käsite *lean* sekä yksi sen käytetyimmistä prosesseista *kanban* on alun perin kehitetty autoteollisuuden parissa. Lean-ajattelumallissa pyritään karsimaan tuotantoprosesseista kaikki turha ja epäoleellinen, ja täten maksimoidaan tuottavuus sekä asiakastyytyväisyys sekä minimoidaan toimitusaika. Lean-mallia käsitellään tarkemmin luvussa 3.4. Kanban on tuotannonohjausprosessi, joka auttaa määrittämään optimaalisen tuotantokapasiteetin kussakin ympäristössä ja tilanteessa. Kanbania on paljon käytetty ohjelmistojen kehityksessä, mutta sitä ei käsitellä enempää tässä opinnäytetyössä.

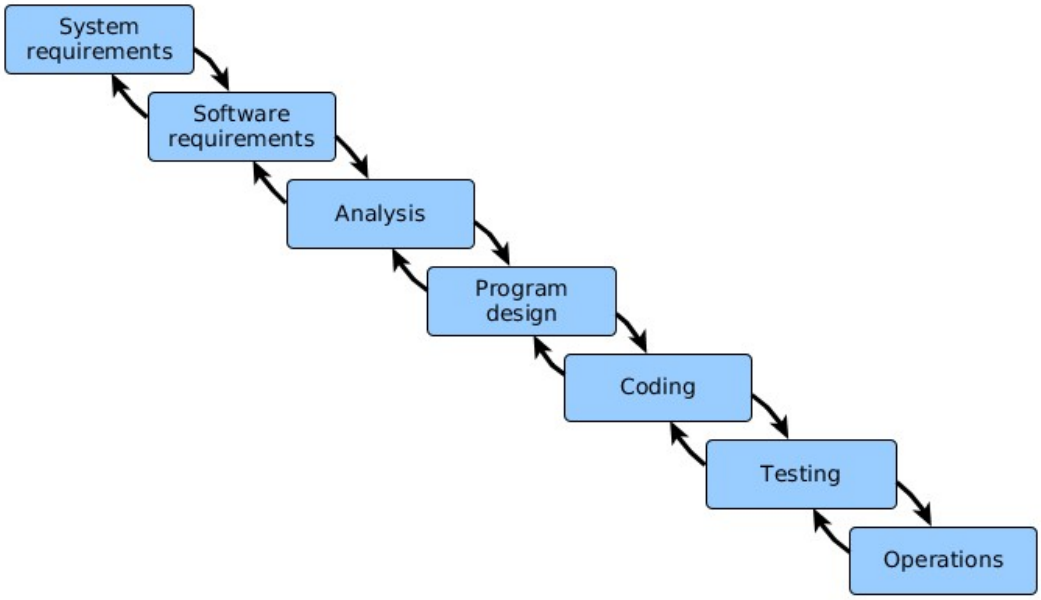
Seuraavassa kappaleessa käsitellään jo tässä kappaleessa pikaisesti esiteltyä vesiputousmalliksi kutsuttua ohjelmistokehitysprosessia

3.2 Vesiputousmalli

Vesiputousmalli on perinteinen ohjelmistokehitysprosessi joka edeltää ketteriä kehitysmalleja. Se on ensimmäinen malli, joka on muodollisesti määritelty, ja sitä ovat käyttäneen mm. IBM, Microsoft ja NASA. Vesiputousmallin ongelmat tunnistettiin jo varhain, mutta ongelmista huolimatta sitä käytettiin suurissakin ohjelmistoyrityksissä aina viime vuosituhannen loppuun saakka. Esimerkkinä vesiputousmallin käytöstä laajoissa ympäristöissä on Microsoft, jonka nimellinen ohjelmiston julkaisusykli oli 12-48 kuukautta. [11]

Vesiputousmallin – joka vasta paljon sen ensiesittelyn jälkeen nimettiin vesiputousmalliksi – määritti ensimmäisen kerran Winston Royce vuonna 1970. Mallin ominaisuuksia ovat: [12]

- ohjelmiston kehitys on jaettu seitsemään osaan analyysistä operatiiviseen käyttöön
- kehitys etenee suoraviivaisesti alusta loppuun ja iteratiivista kehitystä voi tapahtua vain kahden vierekkäisen vaiheen välillä
- kattava dokumentaatio on avainasemassa siirryttäessä kehitysvaiheesta toiseen
- jokainen kehitysvaihe on selkeästi määritelty
- tietojärjestelmän osien kokonaisintegraatio tapahtuu vasta prosessin lopussa



Kuva 4: Vesiputousmalli (Royce)

Kuvassa neljä on esitelty vesiputousmalli Roycen kuvaamana. Mallin seitsemän osaa ovat:

1. Järjestelmävaatimukset: Tilaajalta kerätään vaatimukset uudelle ohjelmistolle

2. Ohjelmistovaatimukset: Järjestelmävaatimukset muutetaan ohjelmistovaatimuksiksi, joissa kuvataan, mitä ohjelmiston pitää tehdä
3. Analyysi: Ohjelmistovaatimuksista muodostetaan yleinen arkkitehtuuri sekä toiminnalliset vaatimukset
4. Suunnittelu: Arkkitehtuurin perusteella muodostetaan yksityiskohtainen ohjelmistokuvaus ja tekninen määrittely
5. Ohjelmointi: Ohjelmistokuvauksen perusteella kehitetään varsinainen ohjelmisto
6. Testaus: Varmistetaan, että ohjelmisto toteuttaa kaikki siltä edellytetyt vaatimukset
7. Käyttöönotto: Ohjelmisto on operatiivisessa ympäristössä asiakkaan käytössä

Jo esitellessään mallin Royce totesi ongelmalliseksi sen, että ohjelmiston testaus tapahtuu vasta prosessin viimeisessä vaiheessa ja ennusti, että käytännössä kehityksessä saatetaan siirtyä myöhemmästä kehitysvaiheesta aikaisempaan, esimerkiksi testausvaiheessa löytyneen vakavan ongelman johdosta kehitys palaa suunnitteluvaiheeseen. Jos näin käy, Royce toteaa, saatetaan odottaa jopa 100% ylitystä projektin aikatauluun ja budjettiin.

Ratkaisuksi vesiputouksen ongelmiin Royce esittää mallia täydennettävän piirteillä, jotka muistuttavat paljon nykyisiä ketteriä kehitysmalleja: iteratiivisuuden lisääminen, asiakkaan parempi huomioiminen ja aktivoiminen koko prosessin ajan. Roycen lopullisessa mallissa on siltikin piirteitä, joita nykyisessä ketterässä ohjelmistokehityksessä ei tunnisteta: iteratiivisuus on hyvin rajoitettua, jokaisesta kuudesta kehitysvaiheesta on tuotettava hyvin kattava dokumentaatio, perusluonteeltaan lineaarinen prosessimalli sekä varsinkin oletus siitä, että prosessin vaiheet tunnetaan täydellisesti jo suunnitteluvaiheessa. [13]

3.3 Ketterä ohjelmistokehitys

Vaikka ketterä ohjelmistokehitys ei terminä ole levinnyt suuren yleisön tietoisuuteen, on tämän ideologian synnyttänyt julistus julkaisu jo vuonna 2001, eli oikeastaan puhutaan jo kohtalaisen vanhasta ja koetellusta metodiikasta. Ketterä ohjelmistokehitys on sateenvarjotermi, joka kattaa joukon metodeja, joilla pyritään torjumaan varsinkin suurissa ohjelmistoprojekteissa koettuja ongelmia.

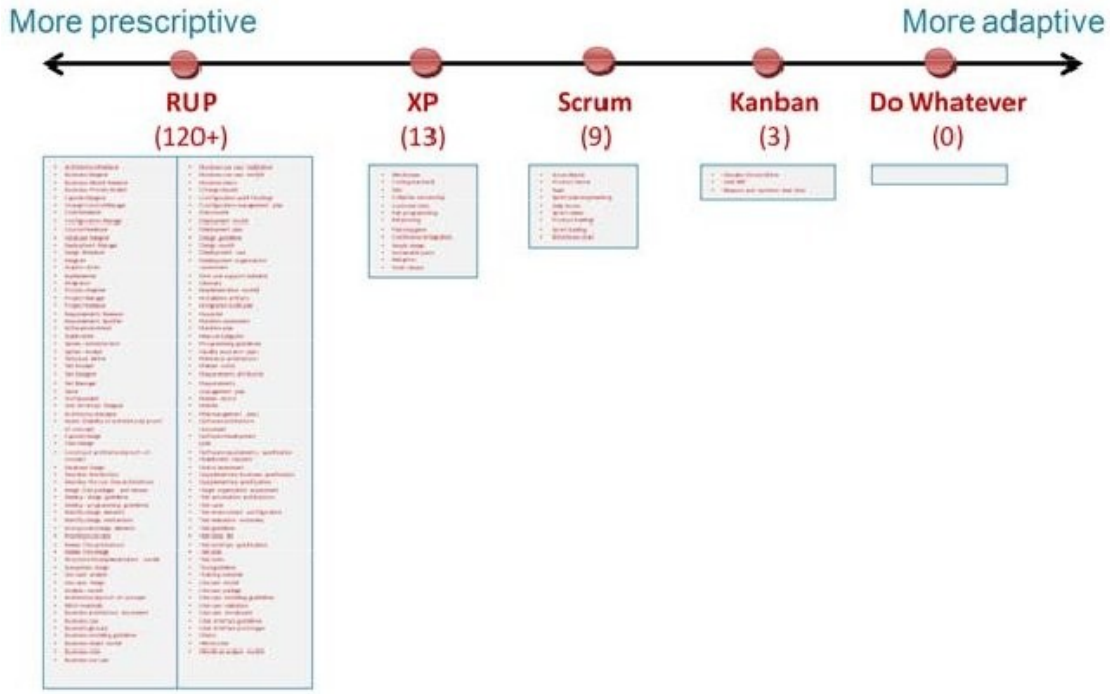
Ketterän ohjelmistokehityksen ytimessä on ajatus itseohjautuvista kehitysryhmistä, jotka reagoivat nopeasti muuttuviin vaatimuksiin. Ketterän ohjelmistokehityksen perustan valanut niin sanottu *Agile manifesto* julkaistiin lokakuussa 2001 ja siinä lueteltiin ketterän ohjelmistokehityksen neljä pääteesiä [14]:

1. Ihmisiä ja näiden välistä interaktiota arvostetaan enemmän kuin prosesseja tai työkaluja
2. Toimivaa ohjelmistoa arvostetaan enemmän kuin kattavaa dokumentaatiota
3. Asiakasyhteistyötä arvostetaan enemmän kuin sopimusneuvotteluja
4. Valmiutta muutoksiin arvostetaan enemmän kuin kattavaa suunnittelua ja suunnitelmassa pysymistä

On huomattava että itse *Agile manifesto* ei sisällä mitään varsinaista työtapaa. Tämän kattotermin alle on kuitenkin kerätty erilaisia ohjelmistokehitystekniikoita, joista tunnetuimpia ovat Kanban, Scrum, XP (eXtreme Programming) sekä RUP (Rationalized Unified Process). Näitä malleja usein verrataan keskenään sen perusteella, kuinka vahva niiden asettama kehys on, eli kuinka paljon ne määräävät eri prosesseja ja rooleja ohjelmistokehitysryhmän työlle.

Kuvassa viisi on verrattu muutamaa yleisintä ketterää ohjelmistokehitysmenetelmää niiden asettaman viitekehyksen rajoittavuuden mukaisesti. IBM:n kehittämä RUP -viitekehys pyrkii eniten ohjaamaan työn kulkua eri rooleineen, työprosesseineen sekä vaiheineen, joita on määritelty yli 120 kappaletta. Esimerkiksi projektin elinkaari

kuvataan RUP viitekehyksessä neljällä eri vaiheella: voimaantulovaihe, kehittelyvaihe, rakennusvaihe ja muutosvaihe. Toisaalta Kanbanin asettamassa kehyksessä on vain kolme rajoitetta: työnkulun visualisointi, käynnissä olevan työn määrän rajoitus sekä läpimenoajan seuranta ja optimointi. Ääripäiden välille jää isommissa tiimeissä useimmin käytetyt menetelmät Kanban, Scrum sekä XP. [17]



Kuva 5: Ketterät ohjelmistokehitysmenetelmät jaoteltuna niiden sitovuuden mukaan

Kaikkein vapain rajoitteista on kuvassa mainittu ”Do Whatever” metodiikka jossa kukin kehittäjä saa työskennellä niin kuin itse haluaa. Tämäkin työtapa toimii hyvin pienissä kehitystöissä, olettaen että kehittäjillä on jonkin verran kokemusta ja että he pystyvät tekemään itsenäisiä päätöksiä.

3.4 Lean-filosofia

Ketterään ohjelmistokehitykseen liitetään usein termi *lean*. Tämä termi on tullut ohjelmistokehitykseen kovan teollisuuden puolelta, Toyotan autotehtailta. Kyseessä on

enemmänkin johtamisfilosofia, eikä niinkään varsinainen työtapa, joka kertoisi suoraan miten asioita pitäisi tehdä.

Leanin ytimessä on kaikkien sellaisten tehtävien karsiminen jotka eivät suoraan vaikuta prosessin lopputulokseen. Leanin seitsemän prinsiippiä on käännetty ohjelmistomaailmaan seuraavasti: [18]

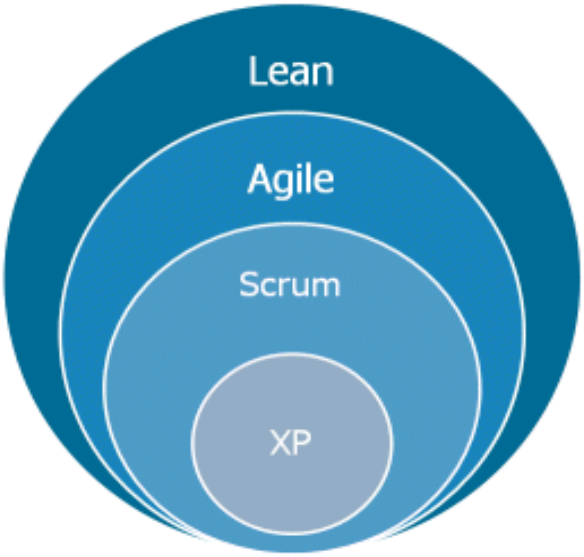
1. Turhan eliminointi
   Kaikki prosessit ja niiden osat, mitkä eivät tuota lisäarvoa asiakkaalle nähdään Lean -filosofiassa turhina. Näiksi voidaan määritellä esimerkiksi ylimääräinen byrokratia ohjelmistokehitysprosessissa, mainitun prosessin hitaus, koodi ja toiminnallisuus, jota ei käytetä tai kehitystä hidastavat epämääräiset määritykset.
2. Oppimisen korostaminen
   Vain jatkuvalla oppimisella mahdollistetaan prosessin kehitys ja estetään taantuminen. Oppimisprosessin ytimessä on mahdollisimman nopean palautteen saaminen, tuli se sitten asiakkailta tai esimerkiksi automaattisen kooditestauksen muodossa.
3. Päätösten teon siirtäminen mahdollisimman pitkälle
   Ohjelmistokehitys ei ole eksaktia tiedettä ja hyvin usein sitä ohjaavat monet epävarmuustekijät. Teknologiat muuttuvat nopeasti ja asiakkaan toiveet vielä nopeammin. Mitä myöhemmin päätös jostain tietystä asiasta tehdään sitä enemmän on sillä hetkellä tietoa ympäristöstä ja sitä todennäköisempää on, että päätös on ainakin oikeansuuntainen. Tämä mahdollistaa muutosten tekemisen ohjelmistoon hyvinkin myöhäisessä vaiheessa kehitysprosessia.
4. Tuotteen, tuloksen tuottaminen mahdollisimman aikaisin
   Jotta oppimisen mahdollistavan palautteen saaminen olisi mahdollista, on Lean -filosofian mukaan tuote pystyttävä julkaisemaan mahdollisimman aikaisin ja mahdollisimman nopeasti. Tämä periaate myös mahdollistaa hyvin nopean reagoinnin muuttuneisiin vaatimuksiin.
5. Päätösvallan siirtäminen kehitysryhmälle
   Nykyisissä ihmisjohtamiseen keskittyvissä asiantuntijaorganisaatiossa on

selvää, että esimies ei voi toimia ainoana päätöksentekijänä. Jos näin olisi, se johtaisi ryhmän luovuuden vähentymiseen sekä kehitysprosessin jäykistymiseen. Lean-filosofiassa esimies toimii enemmän valmentajana, jonka tehtäviin kuuluu kehitysryhmän työn sujuvuuden varmistaminen esimerkiksi kaikenlaisten esteiden poistajana.

6. Kehitetään järkeviä kokonaisuuksia
   Suuretkin ohjelmistot koostuvat osista, joiden pitäisi toimia saumattomasti yhteen. Ohjelmisto ja sen osat elävät ajassa ja niitä pitää aika-ajoin uudelleenkirjoittaa (refactor) vastaamaan uutta ympäristöä ja toiminnallisuutta.
7. Nähdään koko kuva
   Kun kehitetään uutta toiminnallisuutta tai parannetaan vanhaa, on tärkeää nähdä koko kuva. Jos jokin tarpeellinenkin muutos heikentää koko järjestelmää, täytyy tarkoin harkita voidaanko se toteuttaa. Jos ohjelmistokehitys koostuu useasta ryhmästä, ovatko niiden resurssit tasapainossa. Osaoptimoinnilla usein vain siirretään pullonkaulaa paikasta toiseen.

Lean ja Agile tarjoavat yhdessä viitekehyksen, johon Scrum ja muu käytännön toimintaan orientoituneet käsitemallit tukeutuvat.

Lean -ajattelutavan tärkein työkalu on arvovirtakuvaus (value stream mapping), jonka avulla kuvataan tuotannon prosessin nykytila sekä tavoiteltu tila. Arvovirtakuvauksella muodostetussa mallissa on mukana kaikki oleellinen tieto tuotantoprosessista: työn kulku (flow), työhön käytetty aika kussakin työn osa-alueessa, odottamiseen käytetty aika, eri osa-alueilla tehdyn työn kytkeytyminen toisiinsa, valmiin tuotoksen tuottamiseen kuluva aika. Arvovirtakuvauksen avulla löydetään usein ne pullonkaulat ja turhat toiminnot joita lähdetään karsimaan ja optimoimaan.



Kuva 6: Ketterien käsitemallien laajuus

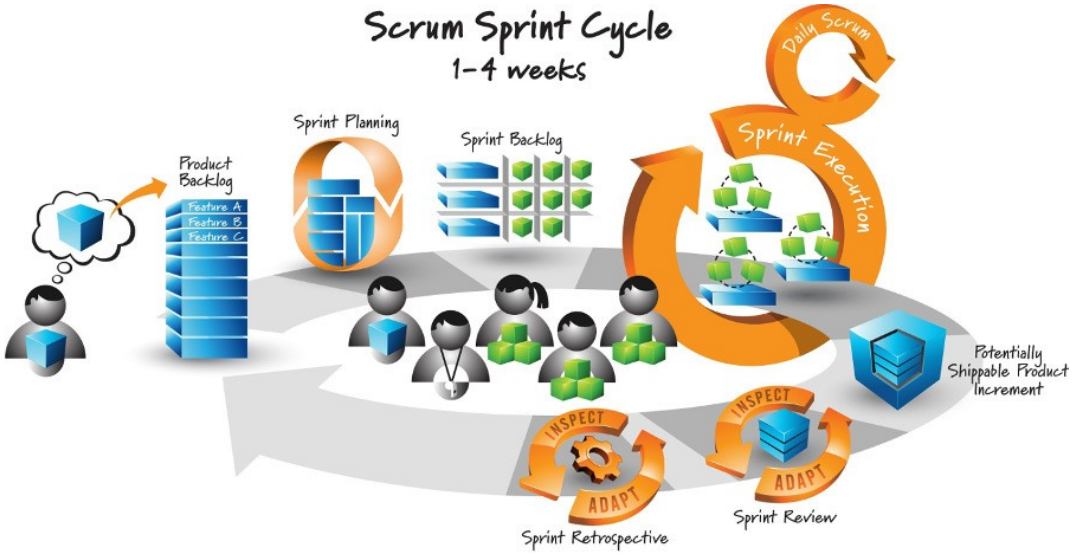
Kuvassa kuusi on esitelty eri ketterien käsitemallien laajuus ja suhde toisiinsa. On huomioitava että XP -ohjelmistokehitysmalli ei suinkaan ole vain osasetti Scrumista tai että Lean on kaikkien muiden supersetti. Kuvassa pyritään näyttämään, kuinka yksityiskohtainen kukin malli: Lean on selkeästi yleisimmällä tasolla, eikä se anna käytännön toimenpiteistä mitään ohjeita. Täten sitä pystytäänkin soveltamaan hyvin erilaisissa ympäristöissä aina teollisuuden valmistusprosesseista ohjelmistonkehitykseen. XP taas on skaalan toisesta päästä: se toteuttaa Lean- ja Agile -ajattelua ja antaa toimintamalleja käytännön toimintaan, mutta on hyvin ohjelmistokehityspainotteinen eikä sovellu siirrettäväksi muille toiminnan aloille.

Seuraavassa luvussa käsitellään tarkemmin kaikkein suosituinta ketterän ohjelmistokehityksen prosesseista, Scrumia.

4 SCRUM

Scrum on nykypäivänä yleisesti tunnettu ja käytetty työskentelytapojen viitekehys, josta löytyy paljon tietoja sekä verkosta että painettuna. Tässä kappaleessa pyrin esittelemään Scrumin pääperiaatteet lyhyesti ja selkeästi, keskittyen enemmän niihin osiin, jotka ovat oleellisia opinnäytetyön kannalta.

Scrum on työkalu ohjelmistojen ja palvelujen luomiseen noudattaen ketterän ohjelmistokehityksen periaatteita. Ensimmäisen kerran Scrumia alettiin käyttää ohjelmistokehityksen työkaluna vuonna 1993; ensimmäinen aihetta käsittelevä kirja julkaistiin vasta 2001. Scrumin ajattelumallin ytimessä on ketterän ohjelmistokehityksen arvojen noudattamisen lisäksi iteratiivisten ja aikarajoitettujen toimintojen hyödyntäminen toimintamallin kaikilla tasoilla. [15]



Kuva 7: Scrum yleisesti

Kuvassa seitsemän on kuvattu kaikki Scrumin määrittelemät komponentit, tuotokset ja roolit. Näitä käsitellään tarkemmin kappaleissa 4.1-4.3. Ylätason kuvastakin voidaan kuitenkin todeta, että työtavassa korostetaan toistuvuutta, iteraatioita. Scrum -prosessin kaikki kehityksen kannalta oleelliset roolit on tarkasti määritelty, jolloin

itse toiminta on mahdollisimman sujuvaa. Selkeästi määritellyt ja avoimet prosessit tuovat kehitykseen läpinäkyvyyttä ja parantavat kommunikaatiota kehitysryhmän sekä asiakkaiden ja sidosryhmien välillä. Jatkuvan oppimisen ja parantamisen korostaminen on Lean -filosofian mukaista toimintaa.

Scrumin sanastolle ei ole täysin vakiintuneita suomennoksia kaikilta osin, tässä työssä pyrin käyttämään Lekman Consultingin julkaisemia ehdotuksia [16].

Scrumin toimintatavan ytimessä on joukko rooleja, toimintoja sekä tuotoksia, jotka kuvataan seuraavissa kolmessa kappaleessa.

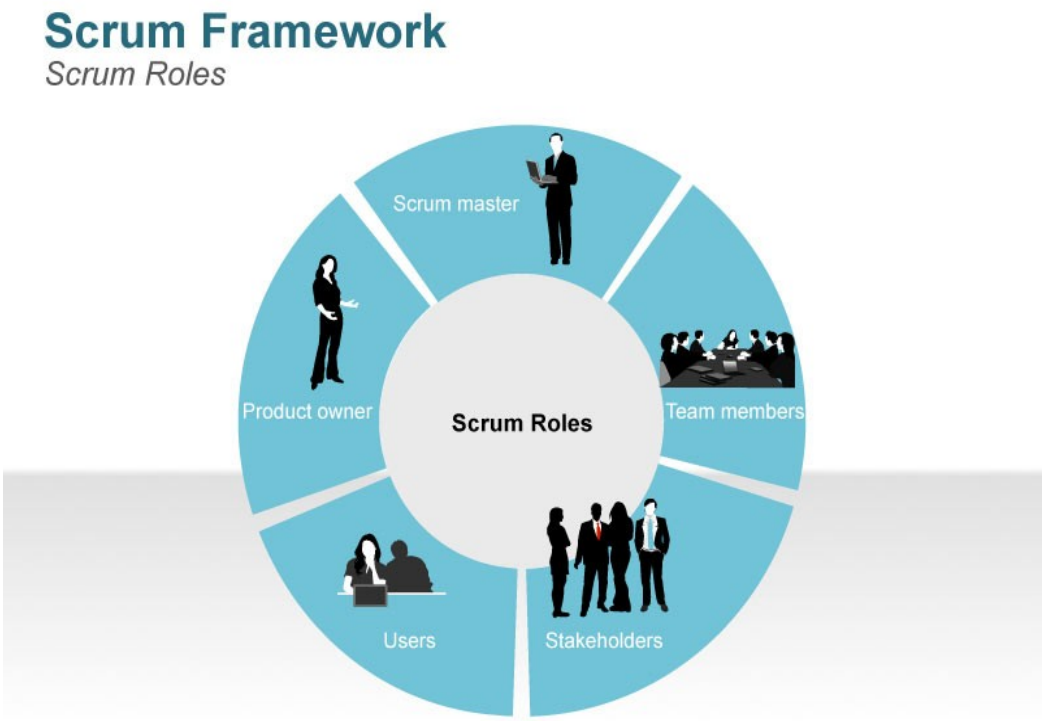
4.1 Roolit

Scrum -kehityksen ytimessä on tiimiajattelu. Scrumtiimi koostuu kolmenlaisista rooleista:

- scrummaster (Scrum master)
- tuoteomistaja (product owner)
- kehittäjä, kehitystiimi (development team)

Lisäksi Scrumissa tunnistetaan seuraavat roolit, jotka eivät suoraan kuulu scrumtiimiin:

- sidosryhmät (stakeholders)
- käyttäjät (users)



Kuva 8: Scrumin määrittelemät roolit

Kuvassa 8 on esitelty kaikki viisi Scrumin määrittelemää roolia. Scrummasterin (suomenkielistä vakiintunutta sanaa käsitteelle ei ole) tehtävänä on varmistaa, että kaikki scrumtiimin jäsenet ymmärtävät Scrumin toimintatavat ja toimivat niiden mukaan. Hän on ryhmän palveleva johtaja tai fasilitaattori, joka mahdollistaa kehitystiimin mahdollisimman jouhevan toiminnan poistamalla kaikki työskentelyä haittaavat esteet ja hidasteet. Scrummaster varmistaa, että prosessi toimii kuten pitää ja on vastuussa esimerkiksi palaverien pitämisestä sekä muista käytännön järjestelyistä. Scrummaster ei yleisesti ottaen ole kehitystiimin linjaesimies eikä hänellä ole suoraa määräysvaltaa tiimin jäseniin.

Tuoteomistaja on asiakkaan edustaja scrumtiimissä. Hänen tehtävänään on maksimoida kehitystiimin työn arvo kohdistamalla työ tuotteen kannalta tärkeimpiin osa-alueisiin. Hän määrittelee tuotteelle asetetut vaatimukset ja niiden prioriteetin; kehityskohteista muodostetaan tuotteen kehitysjono, jota tuoteomistaja hallinnoi. Tuoteomistaja myös auttaa kehitystiimiä vastaamalla heidän kysymyksiinsä

vaatimusten yleisestä luonteesta. Tuoteomistaja on vain yksi henkilö, vaikka hänellä voi olla työnsä apuna suurikin omien asiakkaiden ja yhteistyökumppanien verkosto.

Kehitystiimi on joukko ohjelmistokehittäjiä ja muita asiantuntijoita, jotka muuntavat tuoteomistajan vaatimuslistan valmiiksi ohjelmistoksi. Tiimissä on kaikkien tarvittavien osa-alueiden osaajia, joten se on täysin itsenäisesti toimiva yksikkö. Tiimi on itseohjautuva, eikä sillä ole suoraa linjaesimiestä. Tiimin työskentely tapahtuu sprinteissä, eli aikarajoitetuissa kehitysjaksoissa, joissa kehitystiimi tekee valmiiksi joukon esivalittuja työtehtäviä.

Sidosryhmät ovat tahoja, jotka jollain tavalla liittyvät scrumtiimin kehittämään tuotteeseen. He voivat olla esimerkiksi tuotteiden toimittajia (muita kuin varsinainen nimetty tuoteomistaja), organisaation muuta johtoa tai suoria asiakkaita. Tuotteen omistaja on sidosryhmien suora edustaja scrumtiimissä, mutta Scrumin mukaan sidosryhmät otetaan prosessiin mukaan myös sprintin katselmuksessa. Tästä lisää kohdassa 4.3.

Käyttäjät ovat sidosryhmien tavoin abstraktimpi rooli, eivätkä käyttäjät ole suoraan mukana scrumtiimin toiminnassa. Tuotteen omistaja osaltaan edustaa käyttäjiä ja kaikki tuotekehitysjonon ominaisuudet kirjoitetaan käyttäjäkertomuksen muotoon. Käyttäjät voivat sidosryhmien tavoin olla mukana jokaisen sprintin päättävässä katselmuksessa ja tässä yhteydessä antaa suoraa palautetta koko scrumtiimille.

4.2 Tuotokset

Scrumin tuotoksilla pyritään viestimään mahdollisimman avoimesti ja läpinäkyvästi, mitä tiimi tekee milloinkin. Tuotoksia voi seurata kuka tahansa organisaation jäsen; muokkausoikeus on kuitenkin rajattu tarkasti kullekin tuotokselle.

Scrumin tuotoksia on kolmenlaisia:

- tuotteen kehitysjono (product backlog)
- sprintin tehtävälista (sprint backlog)
- tuoteversio (increment)

Kuva 9: Scrumin tuotokset

Kuvassa yhdeksän on hahmoteltu Scrumin kolme tuotosta sekä niihin läheisesti liittyvät prosessit. Kaikki kehitystyö alkaa tuotteen kehitysjonosta. Se on tuoteomistajan hallinnoima lista kaikista tuotteelle vaadituista ja toivotuista ominaisuuksista. Tuoteomistaja voi lisätä listalle uusia ominaisuuksia, poistaa vanhoja tai priorisoida listaa halunsa mukaan. Kehitysjonon vaatimukset on kirjoitettu käyttäjäkertomuksiksi, joissa kuvataan käyttäjän ymmärtämällä kielellä toivottu toiminnallisuus. Tuotteen kehitysjonossa siis ei oteta kantaa toiminnallisuuden toteuttamisen vaatimiin teknisiin yksityiskohtiin.

Jokaista sprinttiä edeltävässä suunnittelutilaisuudessa kehitystiimi poimii tuoteomistajaa konsultoimalla tuotteen kehitysjonosta kehitettäväksi ne toiminnot, joilla on korkein prioriteetti. Näistä muodostuu sprintin tehtävälista. Tuoteomistaja ei siis suoraan voi määrätä, mitä ominaisuuksia kehitetään.

Sprintin tehtävälistassa on ne tuotteen kehitysjonosta poimitut tehtävät, jotka kehitystiimi arvioi pystyvänsä toteuttamaan yhden sprintin aikana. Kehitystiimi muuntaa tuotteen kehitysjonon yleisellä kielellä kirjoitetut vaatimukset teknisiksi vaatimuksiksi. Tässä prosessissa tuotteen omistajan pitää olla mukana auttamassa kehitystiimiä, mikäli he apua tarvitsevat. Tuoteomistaja ei pysty enää vaikuttamaan sprintin tehtävälistaan. Tehtävälistan työt pisteytetään niiden keston mukaisesti; pisteytysprosessiin osallistuvat kaikki kehitystiimin jäsenet.

Sprintin tehtävälistan tehtäviä aletaan toteuttaa sprintin aikana. Sprintin tuloksena saadaan tuoteversio, joka on sprintin aikana valmistuneiden tehtävälistassa olleiden töiden summa. Sprintin aikana mahdollisesti kesken jääneitä töitä ei siirretä tuotteeseen, vaan tuoteversion katsotaan olevan aina niin valmis, että se voidaan tuoteomistajan niin toivoessa julkaista asiakkaan käyttöön. Sprintin aikana kesken jääneet tehtävät voidaan siirtää automaattisesti seuraavaan sprinttiin, tai ne voidaan arvioida uudelleen seuraavassa valmistelupalaverissa ja tämän arvioinnin perusteella joko toteuttaa tai olla toteuttamatta. On tärkeää huomata että automaattisesti tekemättä jääneet työt eivät siirry sprintistä toiseen, vaikka useimmissa käytännön Scrum -toteutuksissa näin tuntuu tapahtuvan. Tästä lisää luvussa 6.5.

4.3 Tapahtumat

Scrum -prosessia ohjataan tapahtumilla tai toiminnoilla, joita on kuusi erilaista. Nämä toiminnot ovat

- tuotteen kehitysjonon työstö (product backlog refinement, release planning)
- sprintin suunnittelupalaveri (sprint planning)
- sprintti (sprint)
- päiväpalaveri (daily scrum)
- sprintin katselmointi (sprint review)
- sprintin arviointi eli retrospektiivi (sprint retrospective)

Kaikki yllä listatut toiminnot ovat aikarajoitettuja. Mikäli asiat eivät tule käsitellyksi tavoiteajan sisällä, siirretään käsittelemättä jääneet asiat seuraavaan kertaan ja pyritään jatkossa parantamaan toimintatapaa niin, ettei aikarajoja ylitetä.

Kuva 10: Scrumin tapahtumat

Kuvassa 10 on kuvattu Scrumin määrittelemät tapahtumat. Scrum on iteratiivinen toimintatapa, jossa määritellyt toiminnot toistuvat sykleissä. Kukin sykli alkaa tuotteen kehitysjonon muokkaamisella (kuvassa release planning). Tässä toiminnossa kehitysjonoa, jossa ovat käyttäjäkertomuksiksi kirjoitetut tuotevaatimukset, tarkastellaan kriittisesti ja se priorisoidaan. Vanhentuneet kehityskohteet poistetaan ja uudet vaatimukset lisätään listalle. Jokainen kehitysjonon toiminto pilkotaan sopivan kokoiseksi ja sen vaatima työmäärä arvioidaan. Tämän toiminnon päävastuuhenkilö on tuotteen omistaja, mutta kehitystiimin apua tarvitaan varsinkin työaika-arvioiden antamisessa.

Kun tuotteen kehitysjono on saatu muokattua, järjestetään erillinen tilaisuus, jossa valitaan seuraavan sprintin kehityskohteet. Tämä tilaisuus on nimeltään sprintin suunnittelupalaveri. Tuotteen kehitysjonossa saattaa olla paljonkin toivottavaa toiminnallisuutta, jota ei mitenkään saada kaikkea kerralla tehtyä. Täten koko

scrumtiimi valitsee yhteisymmärryksessä tuotteen kehitysjonosta sillä hetkellä kaikkein tärkeimmät tehtävät, jotka toteutetaan sprintin aikana. Jotta sprintin työmäärästä muodostuisi realistinen ja sellainen, jonka kehitystiimi pystyy tekemään, on työmääräarvioiden oltava harkiten tehtyjä. Sprintin suunnittelupalaverin onnistuminen riippuukin paljolti siitä, kuinka huolellista työtä on tuotteen kehitysjonon muokkauksessa tehty. Sprintin suunnittelupalaverilla on kaksi tärkeää lopputulosta:

- koko scrumtiimillä on käsitys siitä, mitä toiminnallisuuksia sprintin aikana toteutetaan
- kehitystiimillä on suunnitelma siitä, miten toiminnallisuudet aiotaan toteuttaa

Varsinkin pidemmissä sprinteissä jälkimmäisen tuloksen merkitys kasvaa, sillä usein sprinttiin valituilla kehityskohteilla on sisäisiä riippuvuussuhteita, jotka asettavat rajoituksia tehtävien suoritusjärjestykselle.

Sprintti on Scrumin toimintatavan ytimessä ja myös varmasti tunnetuin Scrumin toiminnallisuuksista. Sprinttiin osallistuu pääasiallisesti sekä kehitystiimi että scrummaster, mutta myös tuoteomistaja voi olla mukana. Tuoteomistaja ei kuitenkaan voi sprintin aikana ohjata kehitystiimin toimintaa. Sprintin aikana tehtävät työt on listattu Scrum -laudalle, joka on näkyvissä kaikille ja joka voi olla oma ohjelmistonsa tai esimerkiksi vain postit -lappuja ilmoitustaululla. Sprintin normaali kesto on yhdestä neljään viikkoon ja uusi sprintti alkaa heti vanhan loputtua. Sprintin tuloksena on tuoteversio, joka sisältää kaikki sprintin aikana tehdyt muutokset tuotteeseen. Tuoteversio on aina niin valmis, että se voidaan tarpeen vaatiessa julkaista käyttäjille.

Päiväpalaveri, eli daily scrum, pidetään sprintin jokaisen päivän aluksi. Päiväpalaverin maksimipituus on 15 minuuttia ja se pidetään joka päivä samaan aikaan samassa paikassa. Yleisenä tapana on pitää palaveri seisten, jolloin pystytään paremmin pysymään aikarajoituksen sisällä. Päiväpalaverissa kukin kehitystiimin jäsen vuorollaan kertoo kolme asiaa: mitä tein eilen, mitä teen tänään ja mikä mahdollisesti estää tai hidastaa omaa työtäni. Päiväpalaveri on kehitystiimin sisäinen

tiedonjakotapahtuma, mutta se on myös scrummasterille tilaisuus kuulla, miten sprintti etenee ja mitä sellaisia ongelmia tiimillä on jotka hänen pitää mahdollisesti ratkaista.

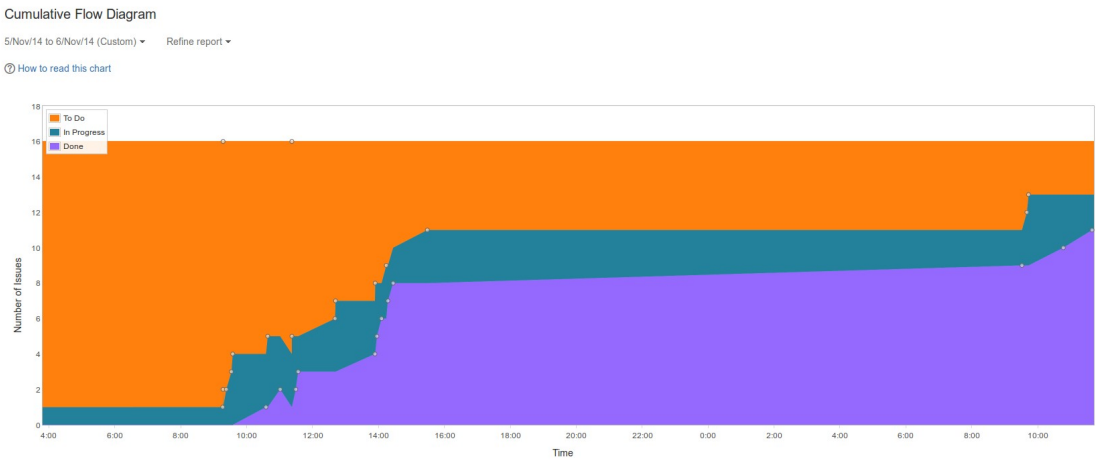
Sprintin katselmointi tapahtuu sprintin päättymisen jälkeen. Katselmoinnissa koko kehitystiimi scrummasterin johdolla esittelee käyttäjille, sidosryhmille sekä tuoteomistajalle sprintin tuloksen eli uuden tuoteversion. Katselmuksen tavoitteena on saada palautetta tuoteversiosta, ja keskustella seuraavan sprintin tavoitteista. Sprintin katselmoinnin kesto on ajallisesti rajoitettu, niin että se korreloi sprintin keston kanssa. Esimerkiksi viikon mittaisen sprintin katselmus kestää maksimissaan tunnin.

Viimeisenä toimintona yhdessä Scrumin syklissä on sprintin arviointi eli retrospektiivi. Tähän tilaisuuteen osallistuu vain kehitystiimi sekä scrummaster. Sprintin arvioinnissa kehitystiimin jäsenet arvioivat avoimesti ja rehellisesti mennyttä sprinttiä ja arvioinnin aikana pyritään löytämään kehityskohteita seuraavaa sprinttiä varten. Arvioinnin kohteena on tiimin toiminta yleisesti, käytetyt työkalut, olosuhteet jne. Retrospektiivillä pyritään jatkuvaan oman toiminnan parantamiseen ja se on erittäin tärkeä osa Scrum -prosessia.

Itsearvioinnin perustana on kolmijakoinen kehityskohteiden etsintä: minkä tekemistä jatketaan, minkä tekeminen lopetetaan ja mitä uutta kokeillaan. Arvioinnin helpottamiseksi on kehitetty myös muita arviointitapoja, esimerkiksi starfish -malli, jossa aiempaan malliin lisätään kaksi kategoriaa ”vähemmän jotain” ja ”enemmän jotain” sekä purjevenemalli jossa pyritään löytämään positiivisia asioita jotka antavat lisää vauhtia (purjeveneessä tuuli) sekä negatiivisia asioita, jotka jarruttavat tiimin toimintaa (ankkuri).

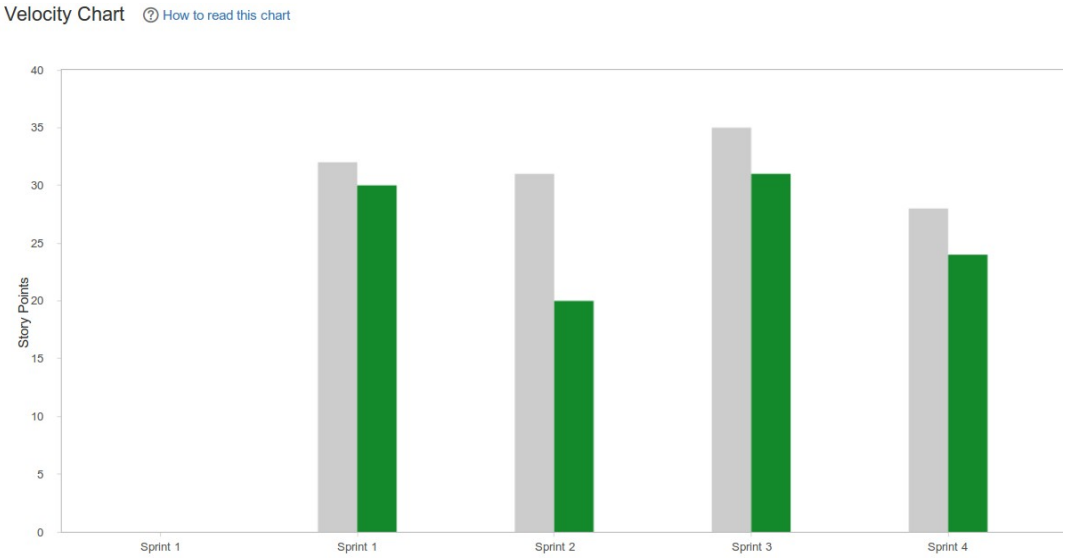
4.4 Seuranta ja mittarit

Sprintin aikana kehitysryhmän suoriutumista annetuista tehtävistä seurataan sprintin edistymiskäyrän avulla (sprint burndown chart). Edistymiskäyrä on jatkuvasti päivittyvä kuva, jossa jäljellä olevaa työmäärää pisteinä verrataan jäljellä olevaan aikaan. Kuvassa 11 on esitetty Jira -työkalun automaattisesti generoima kumulatiivinen edistyskäyrä. Kuvassa aika juoksee vaaka-akselilla, ja pystyakselilla on tehtävien töiden lukumäärä pisteissä. Oranssi väri merkitsee aloittamattomia töitä, sininen väri töitä, joita juuri kyseisellä ajanhetkellä tehtiin ja violetti väri valmistuneita töitä.



Kuva 11: Kumulatiivinen edistymiskäyrä

Toinen mittari on kehitysryhmän vauhti (velocity). Vauhdilla tarkoitetaan yksinkertaisesti sitä työmäärää, minkä kehitysryhmä pystyy tekemään yhden standardimittaisen sprintin aikana. Ensimmäistä sprinttiä valmisteltaessa ei vauhdista voida tehdä kuin arvioita; hyvin nopeasti kuitenkin vauhdista saadaan empiiristä tietoa ja vauhtilukua voidaan käyttää hyväksi sprinttejä suunniteltaessa. Kehitystiimin vauhti on tyypillisesti hyvin stabiili, joskin henkilöiden taitojen kehittyessä myös vauhti kasvaa.



Kuva 12: Vauhtikaavio

Kuvassa 12 on esitelty vauhtikaavio neljän sprintin jälkeen. Vihreät palkit esittävät tehtyjä töitä, ja harmaat palkit suunniteltuja töitä. Palkkien korkeus korreloi työmäärän eli pisteiden lukumäärän kanssa.

5 CASE: SCRUM YLLÄPITOYMPÄRISTÖSSÄ

Alunperin Scrum on kehitetty ohjelmisto*kehityksen* työkaluksi. Tässä opinnäytetyössä sitä kuitenkin sovelletaan ohjelmistojen ylläpitoympäristöön, joka näennäisestä vastaavuudestaan huolimatta on kuitenkin hyvin erilainen ympäristö ohjelmistokehitykseen verrattuna. Selkeistä eroavaisuuksista huolimatta Scrum -prosessia voidaan käyttää molempien alojen työkaluna, kunhan nämä eroavaisuudet otetaan huomioon ja prosessia räätälöidään kuhunkin työhön ja ryhmään sopivaksi.

Tässä luvussa käydään läpi opinnäytetyön aikana toteutetun käyttöönottoprojektin vaiheet yksityiskohtaisesti. Kappaleessa 5.1 kerrataan motivaatio Scrum -projektin toteuttamisen taustalla ja kappaleessa 5.2 käydään läpi ne Scrumin piirteet, jotka jätettiin käyttöönottoprojektissa pois sekä ne, jotka otettiin käyttöön. Lisäksi käydään läpi tarkemmin Scrumin tavoite sekä aikataulu. Kappaleessa 5.3 kuvataan työskentely-

ympäristöä, jossa Scrum otettiin käyttöön, ja kappaleet 5.4-5.8 käsittelevät käytännön Scrum-työtä. Työtavasta kerätyt tutkimustulokset analysoidaan luvussa 6.

5.1 Tavoite ja motivaatio

Ilmatieteen laitoksen operatiivisessa säätuotannossa käytetään, ylläpidetään ja kehitetään kolmannen osapuolen kehittämää ajojenhallintaohjelmistoa. Tämä ohjelmisto on koko tuotantojärjestelmän ytimessä, sillä sen avulla hallitaan monimutkaisia ajoketjuja keskinäisine riippuvuuksineen ja ajastuksineen. Kyseinen ohjelmisto on ympärivuorokautisessa käytössä ja valvonnassa.

Ohjelmisto on vanha, ominaisuuksiltaan rajoitettu ja sen kehitys on lopetettu. Sille on kuitenkin kehitetty korvaaja, joka korjaa monia tunnettuja ongelmia. Tämä uusi versio ei ole täysin taaksepäin yhteensopiva vanhan ohjelmiston kanssa, joten siirtyminen vanhemmasta ohjelmasta uuteen vaatii järjestelmällistä työtä. Uuden ajojenhallintaohjelmiston käyttöönottoon sovellettiin Scrum -metodiikkaa. Samalla kun tuotantorutiinien käynnistäminen ja valvonta siirrettiin ajohallintajärjestelmästä toiseen, siirrettiin samalla myös kaikki toiminnallisuus (ohjelmat, konfigurointitiedostot) vanhoilta tuotantojärjestelmiltä uusille.

Ilmatieteen laitoksen operatiivisesta tuotannosta vastaavan ryhmän toimintakenttä on varsin laaja. Itse tuotannon ylläpidon ja kehityksen lisäksi ryhmän jäsenet ovat aktiivisesti mukana kansainvälisessä toiminnassa ja matkapäiviä kertyy vuosittain jopa 100. Kukin asiantuntija on myös tyypillisesti vastuussa useasta erityyppisestä järjestelmästä, tällä järjestelyllä pyritään varmistamaan järjestelmien vastuuhenkilöiden saatavuus kaikissa tilanteissa. Ongelmaksi tämä muodostuu silloin, kun samanaikaisia tehtäviä on enemmän kuin yksi henkilö pystyy järkevästi toteuttamaan. Ongelma on siis työn pirstaleisuus: työaika jakautuu usean eri työtehtävän kesken, jolloin varsinkin suuret, aikaa vievät projektit saattavat venyä ikuisuuksiin. Lisäksi ylläpidolliset tehtävät ovat aina prioriteetiltaan kehitystöitä korkeammalla, mikä hankaloittaa töiden aikatauluttamista.

Uudella työtavalla pyrittiin ratkaisemaan kaksi asiaa:

- uuden järjestelmän käyttöönotto halutaan tehdä järkevässä aikataulussa ja niin, että se saadaan valmiiksi eikä työ jää roikkumaan tekemättömien töiden listalle
- käyttöönotto halutaan tehdä mahdollisimman tehokkaasti, niin että optimaalisesti työn valmistumisaika on sama kuin työhön käytetty aika; tavoitteena minimoida odotteluun kuluvaa hukka-aikaa (Lean -ajattelussa *muda*)

5.2 Scrum ylläpidon työkaluna

Scrum on pääasiallisesti tarkoitettu sovelluskehityksen työkaluksi. Ylläpitoympäristö eroaa kehitysympäristöstä muutamalla eri tavalla:

- Ylläpitoympäristössä kehitystyötä tehdään, kun siihen on aikaa. Ongelmatilanteiden ratkonta, ennaltaehkäisevät ohjelmistopäivitykset tai laitteiden asennukset ja poistot ovat usein prioriteetiltaan korkeammalla kuin muu ohjelmistojen kehitystyö. Tästä johtuen yhtäjaksoisen, pelkästään kehitykseen varatun ajan löytäminen on vaikeaa ellei mahdotonta.
- Tyypillisesti ylläpitohenkilöstö on vastuussa monesta eri järjestelmästä. Esimerkiksi tietokantaylläpitäjä saattaa olla vastuussa sekä tietokantaohjelmiston ylläpidosta että palvelinten ylläpidosta, varmistuksista ja tietokantasuunnittelusta. Tämä edelleen vaikeuttaa yhtenäisen, yhdelle projektille varatun ajan löytämistä, sillä ylläpitäjän aika on jaettu usean eri toiminnon kesken.
- Ylläpidossa ei usein ole varsinaista asiakasta. Moni toiminta on itseohjautuvaa ja itselähtöistä. Usein kehitetään ohjelmia itselle omaa työtä helpottamaan.

Näistä ylläpitotyön haasteista johtuen Scrum -työtapaa sovellettiin raskaalla kädellä sopimaan työryhmän toimintamalliin. Samalla kuitenkin pyrittiin pitämään kiinni

Scrumin kaikkein olennaisimmista periaatteista, kuten aikarajoitetuista iteraatioista, toimimisesta ryhmänä sekä itsearvioinnista.

Tämän opinnäytetyön aikana Scrumia sovellettiin ylläpitoympäristöön poistamalla menetelmästä joitain elementtejä, jotka eivät sopineet ryhmän ja työkohteen ominaisuuksille. Nämä poistetut ominaisuudet ovat

1. **Tuoteomistaja**
   Scrumia käytettiin uuden toiminnanohjausjärjestelmän käyttöönotossa. Tämä ohjelmisto on ryhmän itsensä käytössä, eikä järjestelmää käytä ryhmän ja tuotannon valvonnan lisäksi kukaan muu. Täten ryhmä on siis käytännössä itse oma tuoteomistajansa.

2. **Tuotteen kehitysjono**
   Erillistä tuotteen kehitysjonoa ei ollut, vaan työt sprintteihin valittiin suoraviivaisemmalla prosessilla scrumtiimin kesken. Tämä osoittautui hyväksi tavaksi, sillä kehitystiimillä oli kattava näkemys tehtävien töiden järjestyksestä sekä aikatauluttamisesta.

3. **Pitkät sprintit**
   Ylläpitotyön luonteesta johtuen pitkäkestoista, yhtenäistä aikaa sprinteille on vaikeaa löytää. Täten sprinttien kesto rajattiin kahteen päivään ja todettiin, että pidetään mieluummin useampi lyhyt kuin muutama pitkä sprintti. Lisäksi tavoitteena oli uuden työtavan asteittainen käyttöönotto; liian radikaalit muutokset työtapoihin saattaisivat herättää vastarintaa ja siten haitata tai jopa estää kokonaan uuden työtavan käyttöönoton.

4. **Sprintin tulosten esittely**
   Koska tuoteomistajaa, käyttäjiä tai sidosryhmiä ei ollut, ei sprintin tuloksia myöskään erikseen esitelty kenellekään. Tuloksia kuitenkin käsiteltiin sisäisesti ryhmän kesken arviointipalaverissa.

5. **Päiväpalaveri**
   Kaikki sprintit järjestettiin niin, että osallistujat istuivat samassa tilassa. Koska sprinttien kesto oli varsin lyhyt ja koska kommunikointi ryhmässä sujui hyvin, ei erilliselle päiväpalaverille nähty tarvetta. Joinain sprintteinä pidettiin lyhyt

päiväpalaverimainen kierros, jossa kukin osallistuja kertoi, mitä tekee seuraavaksi, mutta tällä ei todettu olevan suurtakaan merkitystä tai hyötyä ryhmän toiminnan kannalta.

Scrumin teoreettisesta viitekehyksestä poimittiin mukaan seuraavat ominaisuudet:

1. **Kehitystiimi**
   Kehitystiimin muodosti viisi operatiivisen tuotannon asiantuntijaa. Ryhmän efektiivinen kokoonpano vaihteli eri sprinttien välillä kolmesta viiteen henkilöön (pois lukien scrummaster). Sprintissä oli aina mukana kaikkien välittömästi tarvittavien osa-alueiden asiantuntijat: ohjelmistokehittäjät, ajohallintajärjestelmän ylläpitäjät, järjestelmien ylläpitäjät. Yleisesti ottaen henkilöiden osaamisalueissa on niin paljon päällekkäisyyksiä, että yhden avainhenkilön poissaolo pystyttiin sisäisesti paikkamaan. Sprintin aikana tarvittaessa tukeuduttiin myös laitoksen muihin teknisiin asiantuntijoihin.
2. **Scrummaster**
   Scrummasterina ja ryhmän fasilitaattorina toimi tutkija itse. Tämä oli tosin vastoin Scrumin suosituksia, joissa todetaan, että linjaesimiehen ei pitäisi toimia scrummasterina mahdollisten eturistiriitojen vuoksi.
3. **Sprintit**
   Sprinttejä järjestettiin opinnäytetyön aikana neljä kappaletta, kukin kestoltaan kaksi päivää. Fyysisesti sprintin aikana järjestäydyttiin yhteen yhteiseen tilaan, joka oli muodollisesti eristyksissä muusta työyhteisöstä. Näin saatiin varmistettua työrauha ja maksimaalinen keskittyminen sprintin tavoitteisiin.
4. **Sprintin suunnittelupalaveri**
   Sprintin suunnittelupalaveri pidettiin ennen jokaista sprinttiä. Suunnittelupalaverin aikana päätettiin sprintin töistä, eli käytännössä niistä tuotantorutiineista, jotka siirretään uuteen järjestelmään. Työt päätettiin täysin ryhmän toimesta, koska varsinaista tuotejonoa tai edes tuoteomistajaa ei ollut. Työt myös pisteytettiin niiden vaatiman työmäärän perusteella asteikolla 1-4

pistettä, jossa 1 vastaa puolen päivän työtä ja 4 kahden päivän eli koko sprintin mittaista työtä.

5. **Sprintin tehtävälista**
   Suunnittelupalaverissa päätetyt työt kirjattiin Jira -työkaluun, jota käytettiin töiden backlogina sekä sprintin avustavana työkaluna. Jirasta saatiin automaattisesti generoitua burndown ja velocity -tunnusluvut.

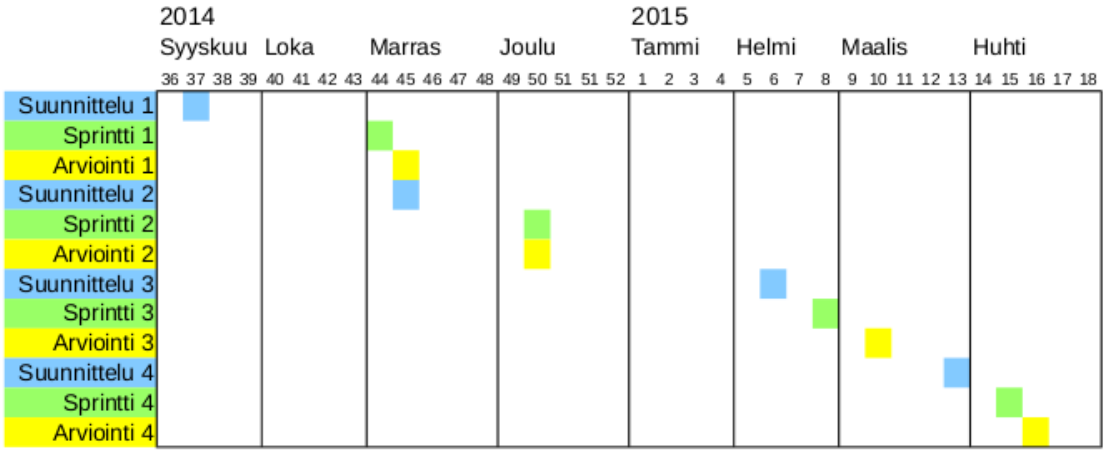
6. **Sprintin arviointi**
   Jokaisen sprintin jälkeen pidettiin arviointikokous, jossa tarkasteltiin mennyttä sprinttiä kriittisesti. Arvioinnin apuna käytettiin Scrumin normaalia kolmijakoista mallia (alla olevan listan kolme ensimmäistä kohtaa), sekä välillä *starfish*- menetelmää jossa perusmallia laajennetaan kahdella uudella kategorialla (listan kaksi alinta kohtaa): [15]

   - asiat, joiden tekeminen lopetetaan (stop doing)
   - asiat, joiden tekemistä jatketaan (keep doing)
   - asiat, joita ruvetaan tekemään (start doing)
   - asiat, joita pitää olla lisää (more of)
   - asiat, joita pitää olla vähemmän (less of)

5.3 Aikataulu

Käyttöönottotyö tehtiin muun työn ohessa eikä sille ollut ulkoista rahoitusta. Tästä johtuen aikataulu oli hyvin joustava ja korkeamman prioriteetin työt menivät usein käyttöönottotyön edelle. Tässä näkyi kuitenkin osittain Scrumin soveltuvuus myös ylläpitotyöhön: kun työtä ei aktiivisesti tehty, oli se kuitenkin taka-alalla koko ajan odottamassa ja siihen palattiin säännöllisesti. Itse työnteko ei kärsinyt pidemmistäkään tauoista sprinttien välillä.

Sprintit aloitettiin loppuvuodesta 2014, ja tämän opinnäytetyön aikana niitä ehdittiin pitää neljä kappaletta. Sprinttien sekä niihin läheisesti liittyvien suunnittelu- ja arviointipalaverien aikataulutus on esitetty kuvassa 13.



Kuva 13: Sprinttien aikataulu

Kokonaisuudessaan opinnäytetyön aikana pidettiin siis neljä sprinttiä, jotka ajoittuivat keskimäärin noin 4-6 viikon päähän toisistaan. Suunnittelupalaveri pidettiin alussa paljon ennen itse sprintin alkamista, mutta myöhemmin todettiin, että jos suunnittelu tehdään lähempänä itse sprinttiä, pysyvät suunnitellut asiat paremmin mielessä. Sprintin arviointi pyrittiin aina pitämään hyvin pian sprintin päättymisen jälkeen.

5.4 Sprintti 1: 5-6. marraskuuta 2014

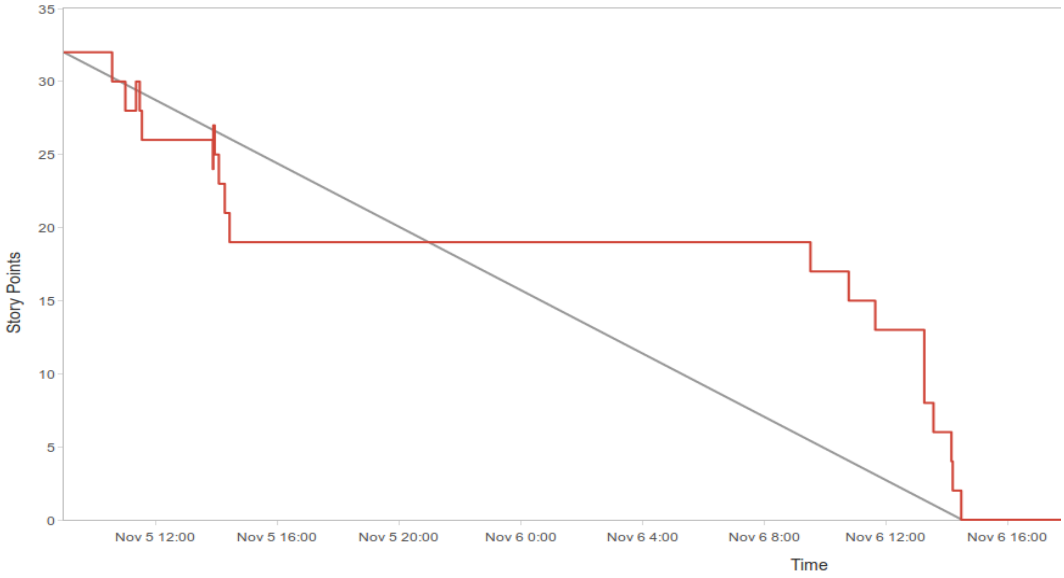
Ensimmäisen sprintin suunnittelu aloitettiin syyskuussa 2014. Suunnittelupalaverissa listattiin ne kehityskohteet, jotka todettiin olevan mahdollista toteuttaa ensimmäisen kaksipäiväisen sprintin aikana. Töiden valitseminen tapahtui hyvin pragmaattisesti: tuotantojärjestelmän nykytila heijastettiin projektorille ja sieltä valittiin ryhmän kesken sellaiset komponentit, jotka todettiin olevat kohtuullisen vaivattomia siirtää. Ensimmäistä sprinttiä varten valittiin tarkoituksella tehtäviä vaikeusasteikon alapäästä. Tämän lähestymistavan huono puoli on se, että koska työn kokonaislaajuutta kaikkine riippuvuuksineen ei edes yritetty mallintaa, oli koko käyttöönottoprojektin

edistymisen arviointi haastavaa. Toisaalta tämä valintatapa oli käytännönkin kannalta ainoa vaihtoehto, sillä tuotantojärjestelmä elää koko ajan jollain tasolla eikä sitä voida pitää staattisena kokonaisuutena. Koska sprinttejä järjestettiin noin kuukauden tai kahden välein, on selvää, että järjestelmän tilaa tarkasteltaessa pitää katsoa aina nykytilaa. Lopullisia tehtäviä valikoitui sprinttiin 16 kappaletta, joille kaikille annettiin alkuarvauksena työmääräarvio 2 pistettä. Täten kokonaispistemäärä sprintille ja samalla kehitystiimin arvioitu velocity oli 32 pistettä. Koko käyttöönottoprojektin arvioiduksi kokonaistyömääräksi saatiin 120 työtä, joten ensimmäinen sprintti sisälsi siis reilut kymmenen prosenttia koko työmäärästä. On huomioitava, että tämä on vain karkeahko arvio, sillä yksittäiset työtehtävät eivät vaikeudeltaan tai kestoltaan ole yhteismitallisia.

Aikataulusyistä johtuen itse sprintti tapahtui noin kaksi kuukautta suunnittelupalaverin jälkeen, marraskuun alussa 2014. Sprintissä oli mukana neljä ylläpitäjää sekä scrummaster. Fyysisesti sprintti järjestettiin erillisessä, normaalista poikkeavassa tilassa, jotta työhön voitaisiin keskittyä mahdollisimman hyvin ja jotta keskeytyksien määrä minimoitaisiin. Koska kaikki sprintin osallistujat istuivat samassa tilassa, ja koska kommunikointi sprintin aikana oli toimivaa, ei erillistä päiväpalaveria pidetty.

Sprintti kesti suunnitellut kaksi päivää ja jälkikäteen todetusti työmäärä oli sopiva sprintin kestoon nähden, sillä vain yksi työ jäi toteuttamatta ja sekin vain osittain. Tämä työ siirrettiin sellaisenaan seuraavaan sprinttiin. Työaika-arviot olivat liian korkeat työmäärään nähden, sillä pisteiden kokonaismäärä oli 32 ja ryhmän teoreettinen maksiminopeus oli 16 pistettä. Tämä oli kuitenkin odotettua, sillä kaikille töille annettiin alkuarvaus kaksi pistettä. Myöhemmissä sprinteissä työt arvioitiin realistisemmin.

Kuvassa 14 on esitetty ensimmäisen sprintin edistymiskäyrä Jira -työkalusta otettuna. Pystyakselilla on jäljellä olevien pisteiden kokonaismäärä, vaaka-akselilla aika. Käyrä laskee sitä mukaa, kun töitä valmistuu. Ylöspäin käyrässä siirrytään silloin, kun kesken sprintin työmäärä kasvaa, esimerkiksi jos uusia töitä on lisätty tehtävälistalle tai jos jonkin työn työmääräarviota on kasvatettu.



Kuva 14: Ensimmäisen sprintin edistymiskäyrä

Sprintin jälkeen pidettiin arviointipalaveri, jossa etsittiin kehittämiskohteita seuraavaa sprinttiä ajatellen. Apuna käytettiin kolmijakoista menetelmää ja löydettyjä kehityskohteita olivat:

- jatketaan eristyksissä työskentelyä ja ryhmähengen ylläpitoa (keep doing)
- lopetetaan kokouksissa käyminen kesken sprintin sekä sähköpostin lukeminen (stop doing)
- kokeillaan isommalla porukalla ja työhön perehdyttämistä työparimenetelmällä (start doing)

5.5 Sprintti 2: 17-18. joulukuuta 2014

Toisen sprintin suunnittelu töiden osalta tehtiin marraskuun puolessavälissä, melko pian edellisen sprintin arviointipalaverin jälkeen. Työtapa oli sama kuin aikaisemmin, eli siirrettävät kokonaisuudet valittiin tuotannon nykytilaa tutkien. Tällä kertaa

valituksi tuli monimutkaisempia töitä kuin ensimmäisessä sprintissä ja jokaiselle työlle annettiin erikseen työmääräarvio asteikolla 1-4. Lopullisesti töitä sprinttiin kertyi 14 kappaletta, joista yksi periytyi edellisestä sprintistä. Työaika-arvioiden kokonaissumma oli 31 pistettä.

Sprintti järjestettiin 17-18. joulukuuta ja osallistujia oli viisi plus scrummaster. Vaikka sprinttiin valmistautumiseen suunnittelupalaverissa keskityttiin aikaisempaa tarkemmin, törmättiin nopeasti siihen, että osaa töistä ei voida toteuttaa sprintin aikana, sillä riippuvuuksia vanhaan järjestelmään on liian paljon. Lisäksi myöhemmin sprintin aikana todettiin, että työmääräarviot eivät pitäneet paikkaansa vaan olivat liian optimistisia. Vaikka pisteissä mitattuna työmäärä oli lähes sama kuin ensimmäisessä sprintissä, ja sprintissä oli mukana enemmän kehittäjiä, ei kaikkia työtehtäviä saatu tehtyä valmiiksi sprintin aikana. Lisäksi osa tehtävistä piti pilkkoa, niin että vain sillä hetkellä mahdolliset osat tehtiin. Loppujen lopuksi valmiiksi saatiin 14 kokonaisuutta, joista muodostui yhteensä 26 pistettä. Yksi tehtävä jäi kesken ja kaksi poistettiin sprintistä mahdottomina (tässä vaiheessa). Kokonaisuudessaan tehtäviä käsiteltiin siis 17 kappaletta, mikä on enemmän kuin alkuperäinen 14, koska töitä pilkottiin pienemmiksi osiksi sprintin aikana.

Kuva 15: Toisen sprintin edistymiskäyrä

Kuvassa 15 on esitetty toisen sprintin edistymiskäyrä. Käyrästä voidaan todeta että ensimmäisen päivän kehitystahti oli ensimmäistä sprinttiä rauhallisempi, johtuen vaikeammista töistä ja toisaalta siitä, että aikaa kului töiden uudelleenarviointiin heikomman valmistelun vuoksi.

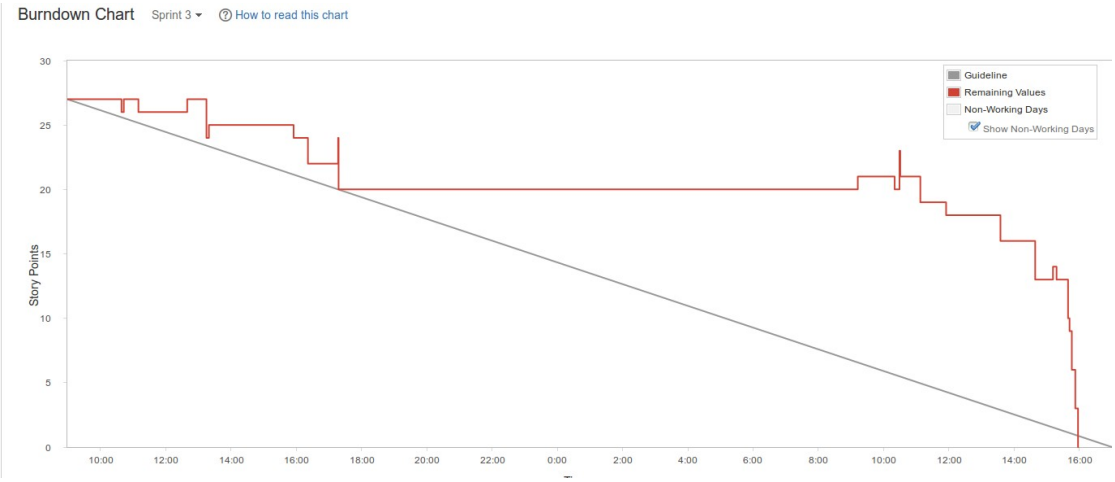
Sprintin arviointi pidettiin seuraavana päivänä ja kehityskohteita löydettiin seuraavasti:

- jatketaan työtavan toteuttamista
- kokeillaan paikan päällä ruokailua
- parannetaan sprinttiin valmistautumista ja tarkempien työmääräarvioiden antamista

5.6 Sprintti 3: 18-19. helmikuuta 2015

Toisen ja kolmannen sprintin välillä oli pidempi tauko, johtuen lomista ja vuodenaluskiireistä. Kolmannen sprintin suunnittelu tehtiin helmikuun puolivälissä ja itse sprintti pidettiin seuraavalla viikolla. Suunnittelussa pyrittiin edellisen sprintin arvioinnin mukaisesti parantamaan työaika-arvioita, sekä selvittämään töiden mahdollinen riippuvuus toisistaan tai ulkopuolisista tekijöistä tarkemmin kuin aikaisemmin. Töitä kertyi 13 kappaletta, yhteensä 26 pistettä, aikaa käytettiin kaksi päivää kuten aikaisemminkin. Työmääräarvioiden pisteluku tarkentui jokaisessa sprintissä kohti ryhmän teoreettista maksimia arvioinnin tarkkuuden kehittyessä.

Sprinttiin osallistui scrummaster sekä viisi kehittäjää. Sprintin aikana todettiin, että riippuvuusanalyyseissa oli onnistuttu aiempaa paremmin, sillä töitä jouduttiin pilkkomaan ja muokkaamaan vähemmän kuin aikaisemmassa sprintissä. Töistä saatiin tehtyä kahta lukuun ottamatta kaikki, pisteissä laskettuna yhteensä 25.



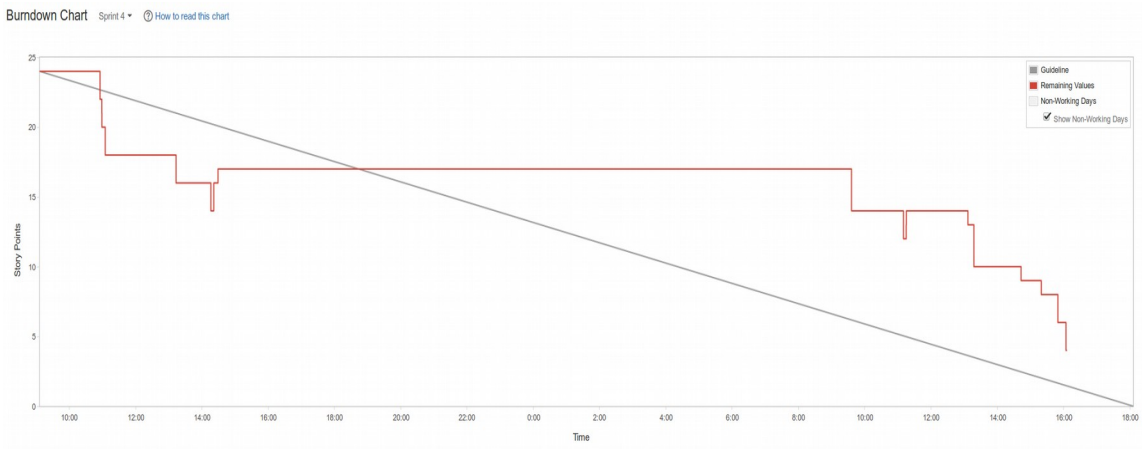
Kuva 16: Kolmannen sprintin edistymiskäyrä

Kuvassa 16 on esitetty kolmannen sprintin edistymiskäyrä. Käyrästä näkyy, että työt saatiin tehtyä sprintin loppuun mennessä, mutta toisaalta käyrän nousuja näkyy jonkin verran, mikä tarkoittaa joko töiden lisäystä tai työmäärien uudelleenarviointia.

Sprintin arviointipalaveri pidettiin maaliskuun alussa. Kehityskohteiksi löydettiin lähinnä fyysiseen ympäristöön liittyviä parannuksia, mm. tilan viihtyvyys ja vaihtelevuus. Palautteen antamisen stimulointiin käytetty kolmimalli ei ole toiminut kovinkaan hyvin ensimmäisen retrospektiivin jälkeen ja sen käytöstä luovuttiinkin kolmannen sprintin jälkeen. Ylipäätään toiminta sprintin aikana oli vaivatonta, eikä suuria ongelmia työskentelytapaa kohtaan tai eri yksilöiden välisiä ristiriitoja esiintynyt. Retrospektiivi pidettiin melko lyhyenä, sillä keinotekoisesti pidennetty retrospektiivi herätti ryhmässä ärtymystä, koska asioiden todettiin toimivan hyvin. Konsensus koko retrospektiivin luonnetta kohtaan olikin, että se on hyvä järjestää, mutta jos kaikki on mennyt moitteetta, niin palautetta ei pysty eikä kannata väkisin yrittää kerätä.

5.7 Sprintti 4: 8-9. huhtikuuta 2015

Neljännen sprintin suunnittelutilaisuus pidettiin poikkeuksellisesti vain päivää ennen varsinaisen sprintin alkua. Tämä koettiin positiiviseksi muutokseksi, sillä sprinttien alkaessa oli tieto tehtävistä töistä vielä hyvin muistissa. Sprinttiin valittiin töitä 13 kappaletta josta kertyi 24 pistettä. Pistemääräarvioilla tämä oli alhaisin työmäärä kaikista sprinteistä. Sprinttiin osallistui scrummaster sekä viisi ylläpitäjää. Kuvassa 17 on esitetty neljännen sprintin edistymiskäyrä.



Kuva 17: Neljännen sprintin edistymiskäyrä

Sprinttiin valmistautuminen onnistui erinomaisesti eikä töitä jouduttu siirtämään tai äkillisiä ongelmia ratkomaan puutteellisen valmistelun vuoksi. Ehkä tämänkin vuoksi valitut työt valmistuivat etuajassa, jonka vuoksi mukaan poimittiin kesken sprintin lisää töitä tuotejonosta eli käytännössä tuotantojärjestelmästä. Lopuksi yksi suurempi kokonaisuus jäi kesken ja se siirrettiin seuraavaan sprinttiin. Neljäs sprintti oli viimeinen, joka mahtui opinnäytetyön aikakehykseen.

Sprintin retrospektiivi pidettiin seuraavalla viikolla. Retrospektiivi pidettiin varsin vapaamuotoisena, sillä aiemmissa tilaisuuksissa ei kehittävää palautetta juurikaan enää löytynyt. Retrospektiivin aikana todettiin, että aikaisemmasta työn kokonaiskestoarviosta poiketen käyttöönottoprojektissa oli neljän sprintin aikana

edetty noin puoleenväliin: tarkempi analyysi tuotantojärjestelmästä osoitti, että lähes kolmannes alkuperäisestä työmäärästä voitiin erinäisistä syistä laskea käyttöönottoprojektin ulkopuolelle. Arviointitilaisuudessa kehityskohteiksi nostettiin

- erillisten näyttöjen käyttäminen sprinteissä; kannettavien tietokoneiden näyttöjen rajallinen koko haittasi kehitystä joissain tilanteissa
- työn parempi koordinointi kehittäjien välillä silloin, kun kaksi henkilöä työskentelee saman kokonaisuuden parissa

Näiden sprinttien pohjalta kerättiin tuloksia Scrumin toimivuudesta yleisesti sekä ryhmän toimintaa sprinttien aikana. Tuloksia tarkastellaan tarkemmin seuraavassa luvussa.

6 TULOSTEN TARKASTELU

Tässä opinnäytetyössä sovellettiin Scrum -viitekehystä melko raskaasti, jotta se sopisi tarkastelussa olevan organisaatioyksikön tarpeisiin mahdollisimman hyvin. Scrum kannustaa räätälöimään prosessia omien tarpeiden mukaisesti. Metodiikasta poistettiin joitain ydinkäsitteitä turhina, joten voidaan jopa sanoa, että käytetty keinovalikoima oli vain Scrumin kaltainen.

Scrum -toteutusta lähestyttiin kahdelta kannalta: alan kirjallisuudesta käytiin läpi yleisempiä johtamiseen ja ryhmädynamiikkaan liittyviä teorioita, ja niitä sovellettiin ryhmän toimintaan. Toisaalta itse käytännön työtä toteutettiin tarpeeksi monta kertaa, jotta siitä saatiin käytännön kokemuksia. Kokemuksia kerättiin työhön osallistuneilta ylläpitäjiltä haastattelujen avulla.

Tässä luvussa tarkastellaan eri tavoin saatuja tuloksia Scrumin soveltuvuudesta tälle kyseiselle työryhmälle ja työkohteelle. Kappaleessa 6.1 analysoidaan, minkälaisessa ympäristössä Scrumia alettiin ottaa käyttöön ja esitetään oma arvio Scrumin

soveltumisesta tälle kyseiselle työryhmälle sekä tiimin toiminnan että sen johtamisen kannalta. Kappaleessa 6.2 analysoidaan Scrum -prosessin johtamiselle asettamia haasteita luvun kaksi esittelemissä viitekehyksissä. Kappaleessa 6.3 esitellään teemahaastatteluista esiin nousseet asiat, ja analysoidaan Scrumin toimivuutta ylläpitäjien eli käytännön työn tekijöiden kannalta. Kappaleessa 6.4 arvioidaan tutkimuksen tieteellistä validiteettia ja viimein kappaleessa 6.5 esitellään muutama vertaistutkimus ja referoidaan niiden johtopäätökset. Tämän tutkimuksen yleisemmälle tasolle vedetyt johtopäätökset kuvataan luvussa 7.

6.1 Scrumin soveltuvuus ryhmälle

Scrum -työhön osallistunut organisaatioyksikkö on vastuussa tuotantojärjestelmien ylläpidosta ja kehityksestä, ja se koostuu eri alojen asiantuntijoista. Asiantuntijaorganisaatioille tyypilliseen tapaan kunkin osa-alueen vastuuhenkilöllä on suuri vapaus ja vastuu kehittää järjestelmiä itse haluamallaan tavalla. Työnjohdon vastuu on enemmän hallinnollisella puolella: riittävään osaamisen varmistaminen, loma-aikojen varahenkilöjärjestelmien turvaaminen sekä yleinen henkilöjohtaminen. Tästä organisoitumistavasta johtuen ryhmän työtapa on hyvin joustava ja välitön. Pakollinen byrokratia on pyritty pitämään minimissä ja esimerkiksi muutoksien tekeminen tuotantoon asiakkaan toiveiden mukaisesti voidaan tehdä helposti ja nopeasti. Joustavan työtavan hinta on sen ajoittainen kaoottisuus: koska formaaliset prosessit on pyritty pitämään minimaalisina, muutosten tekemisen vastuu jää usein yhden henkilön harteille.

Ryhmän asiantuntijoille on annettu hyvin suuri vapaus suunnitella oma työmetodiikkansa. Työympäristön vahvuuksina voidaan nähdä seuraavat seikat:

- ryhmän jäsenet ovat hyvin motivoituneita; vaihtuvuus ryhmässä on hyvin vähäistä
- ryhmässä vallitsee hyvä yhteishenki
- työnantajan näkökulmasta työt hoituvat hyvin kustannustehokkaasti

- ryhmä vastaa erinomaisesti viime hetkelläkin tapahtuviin muutoksiin
- ryhmä kykenee innovoimaan ja kehittämään uusia ratkaisuja jatkuvasti kehittyvässä ympäristössä

Yleisesti työskentelyä kohtaan ei kohdisteta suuria muodollisia vaatimuksia tai prosessin mukaisia toimintatapoja, mikä mahdollistaa hyvin nopean reagoinnin muutoksiin. Tästä johtuvat myös työtavan varjopuolet:

- työn jakautuminen on epätasaista, sillä töiden suunnittelu ei tapahdu keskitetysti
- työn priorisointi voi olla mielivaltaista ilman selkeää kriteeristöä; usein työpanos jakautuu liian moneen kohteeseen joiden prioriteetti vaihtelee jatkuvasti
- yhteistyöhön ryhmän jäsenten välillä ei ole organisatorista tukea vaan se perustuu hyviin henkilökohtaisiin suhteisiin; yhteistyö on täten hyvin henkilö- ja työtehtäväriippuvaista
- muodollisten työtapojen puute saattaa johtaa tuotannossa testaamiseen, mikä näkyy laadun heikkenemisenä tai käyttöpoikkeamana asiakkaalle
- työtapa on hyvin kuluttava johtuen selkeän linjan puutteesta ja jatkuvasta valmiudessa olemisesta

Scrum -mallin mukaiseen kehitystyöhön osallistui opinnäytetyön aikana ryhmän jäsenistä noin puolet. Normaalisti ryhmän jäsenet eivät juurikaan työskentele ryhmissä, vaan yksin tai työparimaisesti. Koko ryhmästä koostettu Scrum -työhön osallistunut ryhmän osajoukko kuitenkin poikkeaa tästä yleisestä muodosta, sillä sprinttien aikana kaikki työskentelivät yhdessä saavuttaakseen sprintille asetetun tavoitteen. Vaikka kehitystiimin jäsenet tunsivat toisensa hyvin ennen sprinttien alkamista, oli itse tilanne kuitenkin uusi ja varsin erilainen normaaliin työskentelytapaan nähden. Tuckerin mallia soveltaen ensimmäisen sprintin alussa ryhmä toimi selvästi forming -tilassa: työtavat eivät olleet tuttuja ja vakiintuneita ja uusi intiimimpi työtilanne aiheutti yleistä varovaisuutta henkilöiden välisissä

suhteissa. Hyvin nopeasti, jo ensimmäisen sprintin loppuvaiheilla, ryhmädynamiikka muuttui ja toimintataso vaihtui suoraan performing -tilaan. Tällöin jokainen ryhmän jäsen löysi oman paikkansa ryhmän hierarkiassa ja ryhmän yleinen henki oli vapautunut. Keskustelu oli monitahoista ja tapahtui suoraan ryhmän jäsenten välillä eikä ohjautunut jonkin tietyn henkilön kautta. Vaikutus näkyi myös sprinttien ulkopuolella, jolloin henkilöiden väliset suhteet olivat välittömämmät kuin ennen Scrum -työskentelyn aloittamista.

Sprinttien aikana myös vahvistui kunkin ryhmän jäsenen rooli ryhmän sisällä Belbinin yhdeksänjakoisen mallin mukaisesti. Henkilöiden roolit sprinteissä ja niiden ulkopuolella olivat varsin samanlaisia, mutta sprinttien intensiivisyys voimisti rooleja ja toi niitä enemmän esille kuin normaali työympäristö. Jos ryhmässä esiintyvät roolit jaetaan Belbinin kolmen kategorian – ajattelijat, tekijät sekä ihmisosaajat – mukaisesti, voidaan todeta, että sekä tekijöitä että ihmisosaajan rooleja omaavia henkilöitä oli lähes yhtä paljon. Ajattelija -roolin edustajia ryhmästä löytyi sprinttien aikana hiukan vähemmän kuin kahden muun kategorian edustajia. Sprinttejä ajatellen tämä on erittäin hyvä kokoonpano, sillä toteutettujen sprinttien luonne oli varsin suorittava: hyvän esityön ansiosta sprinteissä pystyttiin tekemään työtä varsin suoraviivaisesti eikä suurempia avoimia kysymyksiä tai rajanvetoja tarvinnut tehdä. Koska sprinttien aikana ryhmä kokoontui yhteen fyysiseen tilaan, oli tunnelma välillä hyvin hektinen ja intensiivinen, jolloin ihmisosaajien ryhmä pystyi tasapainottamaan sisäistä tilaa ja varmistamaan, ettei energiataso noussut liian korkealle. Ajattelijoista sprinteissä tarvittiin erityisesti analysoija -roolin omaavia henkilöitä, jotka varmistivat osaltaan, ettei kovassa paineessa ja kiireessä sorruttu vääriin ratkaisuihin.

6.2 Scrum johtamisen kannalta

Scrum -prosessin mukaan scrumtiimin johtaja on scrummaster. Scrummasterilla ei kuitenkaan ole suoraa käskyvaltaa kehitystiimin jäseniin, vaan hän toimii oikeastaan Scrum -prosessin ohjaajana. Tässä opinnäytetyössä Scrumia sovellettiin, niin että scrummaster oli samalla myös kehitystiimin linjaesimies. Yhdistelmä ei ole sinällään

kielletty Scrumin säännöstön mukaan, mutta se edellyttää, ettei scrummaster pyri ohjaamaan kehitystiimiä hyväksikäyttäen esimiesasemaansa. Tästä johtuen linjaesimies-scrummaster joutuu keskittymään normaalia enemmän siihen, ettei sekoita näitä kahta roolia keskenään.

Hersey jakoi johtamismallissaan johtamisen tyylit neljään kategoriaan riippuen johdettavan ammatillisesta kehitystasosta. Kehitystasot olivat järjestyksessään vähemmästä enemmän kehittyneeseen saneleva, myyvä, osallistuva sekä delegoiva. Scrummasterin pitää scrumtiimiä johtaessaan huomioida, että päätösvalta ammatillisista asioissa on pitkälti luovutettu tiimille. Täten Scrum -tiimiä johdettaessa saneleva johtamistyyli on luonnollisesti poissuljettu. Myöskin delegoiva tyyli sopii huonosti Scrum -prosessiin, sillä scrummasterin edellytetään aktiivisesti toimivan aktiivisesti ryhmän hyväksi estämällä ja poistamalla mahdollisia työhön tai henkilösuhteisiin liittyviä ongelmia. Toimivimmaksi tyyliksi tämän opinnäytetyön sprinteissä paljastui osallistuva tyyli, jossa johtamistyyli on ihmiskeskeinen. Tämä ei ollut suuri yllätys ottaen huomioon aikaisemmin mainitun scrummasterin toimenkuvan. Kuitenkin, mikäli kehitystiimi on kehitystasonsa alkupäässä, Tuckerin norming tai storming -tiloissa, on scrummasterinkin mahdollisesti ohjattava tiimiä vahvemmin siirtymällä hetkellisesti myyvään tilaan, jossa sekä tehtävä- että ihmiskeskeisyys ovat korkeat. Tästä moodista pitäisi kuitenkin pyrkiä eroon heti, kun kehitystiimi on valmis päättämään asioista itse.

Yleisesti ottaen Scrumin sprinttien ympärille keskittyneessä työtavassa oli linjaesimiehen ja organisaation kannalta selkeitä positiivisia sekä negatiivisia piirteitä. Positiivisina piirteinä nousivat esiin työn tehokkuus, suunnitelmallisuuden lisääntyminen sekä ryhmän tietotaidon ja osaamisen kehitys ja jakautuminen. Työn tehokkuus näkyi varsinkin siinä, että ne työt, jotka sprintteihin määriteltiin, saatiin pääasiallisesti tehtyä. Kokonaisuudessaan se käyttöönottotyö, mitä Scrumin avulla lähdettiin tekemään, edistyi hyvin sprinttien aikana.

Yksi tärkeimmistä positiivisista piirteistä organisaation kannalta oli kuitenkin ryhmän sisäisen tietotaidon kehittyminen sekä jakautuminen. Kehitystiimin jäseninä oli sekä

kokeneempia ylläpitäjiä että vasta aloittaneita henkilöitä ja sprintit osoittautuivat erinomaisiksi paikoiksi tiedon jakamiseen. Tiedon jakamisen mahdollisti sprintin välitön tunnelma, jossa uskallettiin kysyä apua ja sitä myös tarjottiin auliisti. Vastaavaa oppimistilannetta ei organisaation normaalissa arkityötilanteessa synny.

Työn suunnitelmallisuuden lisääntyminen näkyi etenkin sprinttien suunnittelupalavereissa. Ensimmäisissä palavereissa työmäärien arviointi oli karkeaa ja työmääräarvioita edustavia numeroita annettiin ilman syvällisempää analyysia. Myöhemmissä suunnittelupalavereissa työt arvioitiin ja pisteytettiin tarkemmin ja sprinteissä saatettiin todeta arvioinnin onnistuneen. Sprintin jälkeen pidettävä arviointitilaisuus osoittautui aluksi hankalaksi, sillä oman toiminnan systemaattisesta arvioinnista ei ollut kokemusta. Lisäksi arviointi tehtiin ryhmässä ja arvioitavana oli koko ryhmä, jolloin ainakin aluksi näkyi selkeätä varovaisuutta arvioiden antamisessa, sillä kukaan jäsen ei tahtonut ainakaan liian voimakkaasti arvioida ryhmän ja täten muiden ylläpitäjien toimintaa. Myöhemmin arviointitilaisuudet pidettiin tarkoituksellisesti aika kevyinä, sillä prosessin todettiin toimivan niin hyvin, että palautteen puristaminen ryhmästä oli ajanhukkaa.

Scrum -työtavassa tavattuja negatiivisia tai negatiivisella tavalla vaikuttavia seikkoja olivat sprinttien intensiivisyys sekä riski siitä, että samassa organisatorisessa ryhmässä työskentelevät henkilöt, jotka eivät osallistuneet sprinttiin, saattoivat tuntea itsensä ulkopuoliseksi. Sprinttien intensiivisyys, joka syntyi siitä, että kaikki scrumtiimin jäsenet kokoontuivat samaan tilaan ja työskentelivät samaa päämäärää kohden, oli pääasiassa se ajava voima, joka mahdollisti kaikki positiiviset muutokset. Intensiivisyys on kuitenkin hyvin kuluttavaa sekä scrummasterille että kehitystiimille. Opinnäytetyön aikana sprinttien kesto rajattiin kahteen päivään ja niitä pidettiin kerran kuussa tai harvemmin. Sprinttien kestoa tuskin olisi voinut kasvattaa, frekvenssiä olisi ehkä voinut tiuhentaa, jos olisi ollut tarve. Jos Scrumia haluttaisiin käyttää pääasiallisena työtapana, täytyisi sprintit järjestää niin, että kukin henkilö työskentelisi omalla työpisteellään. Tällöin intensiteetti laskisi mutta samalla tuloksetkin jäisivät alhaisimmiksi.

Toinen mahdollisesti negatiivisesti vaikuttava seikka Scrum -työtavan käyttöönotossa on tilanne, jossa laajemman ryhmän sisälle muodostuu toinen vahvempi ryhmä. Selvästikään Scrum ei sovi kaikille eikä ole realistista olettaa, että kaikki siihen osallistuisivat. Tällöin on vaarana, että ne henkilöt, jotka eivät osallistu Scrumiin tuntevat itsensä ulkopuoliseksi. Vieraantumisilmiötä voi ehkäistä parhaiten puhumalla Scrumista ja sprinteistä avoimesti ja kutsua ihmisiä paikalle katsomaan, mitä sprintti käytännössä tarkoittaa.

6.3 Kehittäjäkokemukset

Scrumiin osallistuneilta ylläpitäjiltä (eli Scrum-lingossa kehittäjiltä) kerättiin tietoa teemahaastatteluin, joita tehtiin kolme kappaletta, kaksi henkilökohtaista sekä yksi ryhmähaastattelu. Henkilökohtaisiin haastatteluihin valikoitiin työkokemuksiltaan vastakkaiset henkilöt, joista toinen oli vasta aloittanut substanssin mukaiset työt ja toisella oli yli 10 vuoden kokemus. Näillä ääripäillä pyrittiin hakemaan kokemuksia työuransa eri vaiheessa olevilta henkilöiltä. Ryhmähaastattelussa oli mukana koko Scrum -kehityksessä ollut ryhmä. Kellään haastateltavista ei ollut aikaisempaa kokemusta Scrumista. Kaikkiin haastatteluihin käytettiin aikaa noin puoli tuntia per haastattelu. Haastattelut tehtiin suomeksi tai englanniksi, vieraskielisistä haastatteluista poimitut lainaukset olen kääntänyt suomeksi.

Haastattelussa keskityttiin kahteen teemaan: henkilöiden subjektiiviset kokemukset itse Scrum -työtavasta ja sen eri osa-alueista sekä ryhmän toiminnasta sprinttien aikana. Teemahaastattelurunko on esitelty liitteessä A. Haastattelut nauhoitettiin, jonka jälkeen ne myöhemmin kirjoitettiin puhtaaksi analysointia varten. Analysoinnissa haastateltavien ajatuksia pyrittiin kategorisoimaan ensinnäkin kahden yleisen edellä mainitun kategorian mukaisesti. Tämän kategorioinnin jälkeen ne vielä ryhmiteltiin seuraaviin alakategorioihin: Scrum yleisesti, suunnittelupalaverit, töiden valinta sprinteissä, retrospektiivit, muutosten tekeminen [prosessiin], parannuksia, jatko, aikataulutus, kommunikaatio, työskentely ryhmänä, ryhmän johtaminen, ryhmän koostumus ja roolit sekä oma rooli. Nämä alakategoriat johtuivat osittain ylemmän

tason kategorioista, mutta osittain ne muodostuivat luonnollisesti haastattelutekstin pohjalta. Alakategorioiden ajatukset ja tuntemukset olen pyrkinyt tiivistämään yhteen tai kahteen virkkeeseen, ja jos mahdollista esitän myös suoria lainauksia haastatteluista. Lainaukset on erotettu omiksi kappaleikseen ja kirjoitettu kursiivilla.

Scrum yleisesti

Kaikki haastateltavat toivat Scrumista esille pääasiassa positiivisia seikkoja. Scrumin hyvinä puolina nähtiin erityisesti sprintit, joiden aikana oli mahdollista tehdä työtä keskeytyksettä, ilman häiriötekijöitä. Tämä oli mahdollista, koska sprinttien aikana koko kehitysryhmä kokoontui yhteen fyysiseen tilaan, joka poikkesi normaalista työympäristöstä.

Mielestäni se [sprintti] on hyvä keino lopettaa normaali työ hetkeksi ja siirtyä toiseen huoneeseen ja tehdä vain tätä työtä. Mielestäni se toimii erittäin hyvin.

Scrumin vahvuudeksi todettiin myös sen hyvin selkeä tavoitteen asettelu. Vaikka ryhmällä ei entuudestaan ollut kokemusta Scrumista, opittiin sen työtavat hyvin nopeasti.

Silloin kun olemassa tommonen tiukka substanssitavoite [..], ja puitteet sille, niin siinä tulee se sisäinen [..] oma fiilis että hoidetaan tää homma.

Sprinttien suunnittelupalaverit koettiin myös hyödyllisinä, vaikka hyödyllisyyden asteessa oli nähtävissä enemmän variaatiota. Henkilölle, joka ei omaa pitkää kokemusta työstä on selkeästi nähtävissä enemmän hyötyä siitä, että suunnittelupalaverissa kartoitetaan huolellisesti tehtävät työt, kuvataan ne riittävällä tarkkuudella riippuvuuksineen ja varsinkin annetaan niille realistinen työaika-arvio. Kokeneempi työntekijä joka tietää nämä asiat kokemuksensa perusteella saattaa nähdä perusteellisen työaika-arvioinnin enemmän ajan hukkana.

Mä en näe niillä pisteillä niin suurta merkitystä, ymmärrän periaatteen että yritetään arvottaa niitä ja tietenkin jollain tavalla niitä töitä pitää arvioida. Mutta ehkä ei niihin pisteisiin kuitenkaan loppujen lopuksi kannata hirveästi tuijottaa.

Haastateltavat toivat myös esille suunnittelutilaisuuden demokraattisen luonteen ja sen voimaannuttavan vaikutuksen ryhmään:

Mun mielestä meillä oli [hyvä ryhmähenki] koska me tehtiin se ennakkotyö eli töiden valinta ryhmässä, niin se jo toi ryhmälle semmoisen käsityksen ja yhtenäisyyden tunteen et mitä meidän pitää tehdä.

Sprinttien jälkeiset retrospektiivit, eli oman toiminnan arviointipalaverit koettiin selkeästi kaikkein vähiten hyödyllisiksi. Tämä näkyi jo käytännössäkin palautetta ja arviointeja kerätessä, sillä kehitysryhmän kokemus sprintistä oli pääsääntöisesti niin hyvä, ettei mitään merkittäviä kehityskohteita noussut esille.

Se [retrospektiivi] oli hyödyllistä ekan kerran jälkeen, mutta viime kerralla siinä oli hyvin paljo samaa ja mun mielestä se ei tuonut enää mitään uutta lisäarvoa.

Ryhmän toiminta

Haastateltavilta kysyttiin myös heidän mielipiteensä ja tuntemuksensa toimimisesta ryhmänä. Luvuissa 2 ja 4 käsiteltiin syvällisesti sekä ryhmädynamiikkaa, johtamista että Scrumin rooleja ja näiden teorioiden pohjalta on helppo vetää johtopäätös, että kehitysryhmän on oltava sisäisesti hyvin toimiva ja tarpeeksi kypsä, jotta Scrum -prosessi voi ylipäätään toimia. Kappaleessa 6.1 totesin oman arvioni olevan, että ryhmä on hyvin yhtenäinen, sen jäsenten väliset sosiaaliset suhteet ovat avoimet ja

välittömät eikä ryhmän toimintaa suoraan hidasta tai häiritse mikään sisäinen tekijä. Tämä kuva vahvistui myös haastattelujen kautta.

Ihan selkeesti mun mielestä kaikilla on yhteinen tavoite ja yritetään tukea toisia sen saavuttamisessa.

Hyvän ryhmähengen ja ylipäätään ryhmän efektiivisen toiminnan perusta on toimiva kommunikaatio, kuten luvussa 2.4 asiaa käsiteltiin. Monikielisessä ryhmässä kommunikoinnin merkitys nousee entisestään, sillä ryhmän vähemmistökielet jäävät helposti paitsioon, ellei ryhmä itse kommunikoi omalähtöisesti monella kielellä. Kommunikointi muulla kuin omalla äidinkielellä on aina vaikeampaa ja kynnys kommunikoinnin aloittamiseen on korkeampi. Korkeampi kynnys usein merkitsee, että jotain jää kokonaan sanomatta, koska henkilö ei joko osaa muotoilla ajatustaan sanoiksi tai ei yksinkertaisesti viitsi, koska ajatus ei käänny helposti vieraalle kielelle. Tästä huolimatta haastateltavat toivat selkeästi esille, että kommunikointi sprinttien aikana oli toimivaa ja välitöntä.

Ei mun mielestä kommunikoinnissa ole ongelmia, samassa huoneessa [kun ollaan niin] äkkiähän siinä kysyy joltain tai joku toinen kysyy.

On huomattavaa, että tässäkin kommunikaation edellytys on sama fyysinen tila: jos sprintit olisi järjestetty niin, että jokainen tiimin jäsen istuisi eri huoneessa ja kommunikaatio tapahtuisi teknisillä välineillä (sähköposti, pikaviestimet), ei viestintä olisi niin tehokasta, vaikka ryhmän koostumus olisi täysin sama.

On aina parempi jos voi kysyä suoraan vierustoverilta apua, ja hän voi suoraan katsoa näytöltä koodiasi. Kommunikointi [pikaviestimen] avulla olisi rajallista eikä onnistuisi.

Ryhmän sisäisistä rooleista haastateltavilla ei ollut selkeätä käsitystä. Yksi selitys tähän saattaisi olla se, että vahvoja rooleja ei ehtinyt kehittyä, koska varsinaisten sprinttien kesto on ollut kokonaisuudessaan aika lyhyt, kahdeksan päivää. Toinen mahdollinen selitys on se, että haastateltavat eivät itse vain tunnista rooleja, koska ovat itse niin syvällä ryhmässä etteivät pysty etäännyttämään itseään siitä tai sitten koska eivät yksinkertaisesti ole kiinnittäneet asiaan huomiota. Jälkimmäinen oletus saa vahvistusta myös haastatteluista:

En mä osaa ehkä tunnistaa niitä mutta varmasti [..] muodostuu tämmöiset tietyt roolit, ihan varmasti semmoista tapahtuu.

Mielenkiintoista on, että haastateltavat eivät myöskään tunnista vahvasti omaa rooliaan ryhmässä; ryhmä on niin tasavertainen ettei sen jäsenillä ole tarvetta korostaa omaa rooliaan.

Ryhmän johtamisesta haastateltavat olivat yksimielisesti samaa mieltä, että Scrumin käyttämä ”servant leader” -tyyli, eli tapa jossa scrumtiimin johtaja scrummaster toimii ainoastaan ryhmän avustajana ja fasilitaattorina toimii sekä teoriassa että käytännössä. Tämä ei ole kovin yllättävä tulos, sillä kehitystiimin jäsenet ovat kaikki asiantuntijoita ja tottuneet itsenäiseen työhön sekä päätösten tekemiseen; täten toimintatapa, jossa asiantuntijat tekivät päätöksiä sitä mukaan kuin niitä oli tarve tehdä ja päätökset tehtiin ryhmäkeskustelun jälkeen, jossa kaikilla oli mahdollisuus kommentoida asiaa, oli tasavertaisin ja toimivin.

Mun mielestä me oltiin varsin itsenäisiä kuitenkin, on tärkeää ryhmän hengen kannalta et on sama suunta, tietää mihin ryhmä menee, se sitten luo samalla konsensuksen [ryhmän] sisälle.

Toisaalta haastatteluissa tunnistettiin myös se, että joskus ryhmä saattaa tarvita vahvemman johtajan ja joskus johtajan on myös laitettava oma lusikkansa soppaan.

Tämä sinänsä viaton olettamus sisältää kuitenkin valtavan odotuspaineen scrummasterille: yhtäällä hänen pitää olla vahva ihmisosaaja, joka osaa antaa työtiimille tilaa ja on enemmänkin juoksevien asioiden hoitaja, toisaalta hänen pitää tarvittaessa pystyä tekemään substanssiasioihin liittyviä päätöksiä, silloin kun ryhmä ei siihen kykene ja vieläpä osallistua substanssityön tekemiseen scrumtiimin osana. On selvää, että sellaisia johtajia, jotka kykenevät ja ovat halukkaita tarvittaessa suoriutumaan näistä johtamisen spektrin ääripäistä, on melko harvassa.

Jos se [ryhmä] sitten ei toimi niin sitten tietenkin pitää [johtajan] tehdä päätöksiä.

Seuraavassa kappaleessa käsitellään tutkimuksessa kerättyjen tietojen ja niiden analysoinnin luotettavuutta ja yleistettävyyttä.

6.4 Tulosten luotettavuus

Tässä tutkimuksessa esitetyt tulokset eivät yksityiskohtaisuudessaan ole kovinkaan helposti yleistettävissä. Laadullisen tutkimuksen luotettavuutta mitataan usein reliabiliteetilla ja validiteetilla. Reliabiliteetilla, eli pysyvyydellä, mitataan tutkimuksen ulkoista luotettavuutta, sitä kuinka tiukkaan tutkimus on sidottu johonkin tiettyyn kontekstiin tai viitekehykseen, esimerkiksi aikaan tai kulttuuriin. Validiteetilla mitataan tutkimuksen sisäistä luotettavuutta, esimerkiksi onko tutkittu oikeata asiaa ja vastaavatko tutkimuksessa tehdyt johtopäätökset sitä todellisuutta, josta ne on saatu. [19]

Tämän tutkimuksen reliabiliteetti on melko alhainen: tutkimus kohdistuu yhteen organisaatioon ja tutkimusjoukko on varsin pieni. Lisäksi tutkimuksen kohteena ollutta työtapaa räätälöitiin voimakkaasti organisaation tarpeisiin, mikä vähentää tutkimuksen työtavasta saatujen tulosten yleistettävyyttä ja vertailun mahdollisuutta muihin vastaaviin tutkimuksiin. Alhainen reliabiliteetti on hyvin yleistä laadullisissa tutkimuksissa, joissa validiteetti saakin yleisesti ottaen enemmän huomiota.

Validiteetin määritelmä voidaan jakaa useaan osaan, joista eniten käytetyt ovat sisäinen validiteetti ja sisältövaliditeetti. Sisäisellä validiteetilla mitataan kuinka saadut tulokset vastaavat tutkimuksen todellisuutta. Sisäistä validiteettia voidaan tarkastella esimerkiksi ajan, mittaustapahtuman tai mitattavien kohteiden näkökulmasta. Esimerkiksi jos tutkimus on kestänyt pitkään, onko tutkittava kohde muuttunut oleellisesti tutkimuksen aikana. Tässä tutkimuksessa sisäinen validiteetti on korkea, sillä tutkimus on tehty lyhyessä ajassa, ja vaikka ryhmän tutkimusjoukko on absoluuttiselta kooltaan pieni, edustaa se kuitenkin ryhmän enemmistöä.

Sisältövaliditeetti kuvaa kuinka hyvin tutkimusaineiston analysointimenetelmä vastaa aineiston luonnetta. Laadullisessakin tutkimuksessa on selvittävä, kuinka tutkija on analysoinut aineistoa, mitä kriteerejä tässä on käytetty ja miten tutkija on päätynyt tutkimuksessa esitettyyn johtopäätökseen.

Työn ensimmäisessä vaiheessa, jossa tietoa kerättiin johtamisesta ja ryhmän toiminnasta, tulokset on kerätty pääasiallisesti henkilökohtaisen kokemuksen kautta, joka laskee validiteettia. Validiteettia nostaa se, että ryhmän toimintaa ja johtamista analysoitiin usean eri teoreettisen, toisistaan riippumattoman viitekehyksen perusteella. Teemahaastatteluilla kerätyn tiedon validiteetti on lähtökohtaisesti korkea, sillä sekä teemahaastattelurunko että analysointikeinot on selostettu tutkimuksessa ja ne korreloivat myös tutkimuskysymyksen asettelun kanssa.

6.5 Vertaistutkimuksia

Täysin vastaavia töitä ei alan kirjallisuudesta löydy. Markus Ahonen tutki diplomityössään Scrumin soveltuvuutta vesiputousmallin korvaajana sovelluskehitysprojekteissa. Työssä tutkittiin kvalitatiivisilla keinoilla, kuinka yritys vaihtoi vesiputousmallin Scrum -malliin eräässä ohjelmistokehitysprojektissa. [20; 1-2, 39-42, 52-53] Tulokset olivat kahtiajakoiset:

- viestintä tilaajan (kyseinen yritys) ja tuottajan välillä parani
- tilaaja sai enemmän ja halvemmalla (tilaajan oma näkökulma)
- aikataulu venyi kolminkertaiseksi alkuperäisestä
- tilaaja ei sitoutunut tarpeeksi Scrum -prosessiin, mikä johti näkemyseroihin ja kinasteluun vastuualueista
- Scrum -menetelmän ylläpito osoittautui raskaaksi ja siitä lipsuttiin
- sprintteihin valmistautuminen ei ollut tarpeeksi huolellista, mikä näkyi kehittäjäryhmän tuottavuuden laskuna

Panu Vuori tutki omassa opinnäytetyössään Scrumin soveltuvuutta ohjelmistoprojektien hallintaan. Työssä tutkittiin kvantitatiivisesti Scrumin vaikutusta aikaisemmin ohjelmistoprojektien kehityksessä koettuihin ongelmiin läpinäkyvyydessä, kommunikaatiossa asiakkaan kanssa sekä vastuuhenkilöiden työajan hallinnassa. [21; 54-62] Työssä tehtiin seuraavanlaisia havaintoja:

- projektin sisäinen kommunikaatio parani
- projektin etenemisen seuraaminen helpottui, mukaan lukien työaika-arvioiden tekeminen
- Scrum on työtavaltaan hyvin erilainen kuin aiempi työtapa, tästä johtuen muutosvastarinnan välttämiseksi Scrumia kevennettiin mm. palavereja vähentämällä
- käytössä olleen Excel -pohjaisen työkalun ylläpitämiseen meni useita työtunteja viikossa ja oli täten liian työläs scrummasterille
- työtavan käyttöönotto ei vaikuttanut kustannuksiin mitenkään
- kehitetyn ohjelmiston käyttöönottovaiheessa Scrum prosessin koettiin olevan liian hidas äkillisten ongelmien ratkaisemiseksi ja prosessi ohitettiin

Nichamon Chantachaimongkol ja Puangpetch Sincharoenpanich Mälardalenin yliopistosta tutkivat kvalitatiivisin keinoin, mitkä elementit ovat oleellisia onnistuneen

Scrum -projektin kannalta. Tutkimusta varten tietoa kerättiin olemassa olevasta kirjallisuudesta sekä haastattelemalla yrityksiä, joissa on Scrum käytössä. Tutkimuksen johtopäätöksenä Chantachaimongkol ja Sincharoenpanich löysivät kolme erillistä tekijää (osa-aluetta), jotka kaikki pitää olla kunnossa jotta Scrum -projekti voi onnistua: [22; 39-48]

- organisaatiotekijä, eli johdon tuki Scrum -prosessille, asiakkaiden sitoutuminen prosessiin, työympäristö sekä käytetyt työkalut
- ihmistekijä; tutkijoiden mukaan sekä henkilöstön osaaminen ja kokemus että kehitystiimin sisäinen kommunikaatio ovat avaintekijöitä, jotka mahdollistavat onnistuneen projektin
- tekninen tekijä; tutkimuksen mukaan projektilla, jonka vaatimukset on määritelty mahdollisimman tarkasti ja jonka kehitys- ja testausprosessit ovat kaikkien tiedossa on parhaat mahdollisuudet onnistua

Yleisesti kirjallisuudesta on pääteltävissä että Scrumin käyttöönotto ei suoraan ratkaise kaikkia organisaation ohjelmistokehityksen ongelmia. Jokainen organisaatio ja kehitysprojekti on omanlaisensa ja vaatii räätälöintiä, jotta Scrumin viitekehitys saadaan sovitettua mahdollisimman hyvin työryhmälle. Nähtävissä on myös että jotkut Scrumin ominaisuudet kuten sprintit huomataan heti hyvin toimiviksi, mutta toiset kuten esimerkiksi palaverikäytännöt vaativat enemmän harjoittelua ja suunnittelua.

Uusimpana tulokkaana ohjelmistokehitysmallien perheeseen on tullut Scrum-ban, joka nimensä mukaisesti yhdistää Scrumin ja Kanbanin parhaat puolet. Scrum-banissa on poistettu Scrumin sprintit ja otettu tilalle Kanbanissa oleellinen työvaiheiden visualisointi sekä käynnistä olevan työn määrän rajoittaminen. Scrum-bania markkinoidaan paljolti ylläpitokäyttöön, sekä muihin sellaisiin ympäristöihin, joissa kriittisiä työtehtäviä tulee yllättäen ja niihin pitää reagoida nopeasti. [23]

Tässä opinnäytetyössä ei juurikaan menty Scrum-banin viitoittamalle tielle, vaan Scrumista luotiin omanlainen versio. Mielestäni tämä oli oikea ratkaisu, sillä koko

kokeilun parhaaksi asiaksi arvioitiin sprinttien ympärille rakentuneen työtavan tehokkuus. Myös uuden tietojärjestelmän käyttöönotto, jota sprinteissä tehtiin, oli otollinen työtavalle, koska sillä oli selkeä alku ja loppu. Normaalissa ylläpitotyössä Scrum-ban voisi hyvin toimia tutkimusryhmänkin käytössä.

7 JOHTOPÄÄTÖKSET

Tässä kappaleessa esitetään edellisessä kappaleessa esitettyjen tulosten perusteella vedetyt johtopäätökset Scrumin soveltumisesta ohjelmistojen ylläpitotyöhön, esimerkiksi käyttöönottoprojekteihin, kuten tässä työssä tehtiin.

Vertaistutkimusten tulokset ja tutkimuksen aikana saadut tulokset ovat hyvin samansuuntaisia. Scrum-tiimi vaatii onnistuakseen johdolta selkeän tuen ja toimintavapauden. Toisaalta tiimin pitää olla itsessään koherentti ja jonkin verran kokenut jotta ryhmältä voidaan odottaa hyviä tuloksia. Scrum -prosessi toimi sovelletulla tavallaan hyvin ja se sai lähes pelkästään positiivisia arvioita.

Scrumin soveltuvuutta ohjelmistojen ylläpitotyöhön voidaan arvioida työn kahden osa-alueen kautta: itse työtavan käytännön elementtien soveltuvuuden kautta, sekä työryhmän kypsyyden ja osaamisen kautta.

Yleisesti ottaen voidaan todeta että Scrum oikein sovellettuna sopii erinomaisen hyvin ohjelmistojen ylläpitotyön työkaluksi. Tässä työssä työtapaa käytettiin järjestelmän käyttöönottoprojektissa, jossa tuotannon kannalta keskeisiä osioita siirrettiin Scrumin avulla vanhasta järjestelmästä uuteen. Scrumin soveltuvuudesta ylläpitoprojekteihin poimin seuraavia huomioita:

- Scrumin avulla työmääräarviot muodostuivat osuviksi ja koko käyttöönottoprojektin edistymistä oli helppo seurata

- työtavan demokraattisen luonteen vuoksi tieto tuotannon substanssityöstä jakautuu tasaisesti koko ryhmän yli
- työ on tehokasta, sillä sprinttien aikana pystytään keskittymään vain suunniteltuun työhön eikä keskeytyksiä tule
- kehittäjät osallistuvat sprintin suunnitteluun ja työmäärien arviointiin, mikä sitouttaa ryhmän yhteiseen tavoitteeseen ja vahvistaa ryhmähenkeä
- toteutetussa muodossaan sprinttejä ei voitu järjestää kovin usein, sillä ne olivat melko kuluttavia, lisäksi ylläpitotyö ei salli että ylläpitäjät ovat kovinkaan usein dedikoitu johonkin tiettyyn työtehtävään (ja täten pois normaalista arkityöstään)
- jotkin Scrumin kuvaamat rakenteet ovat merkityksettömiä ylläpitotyössä (käyttäjät, tuoteomistaja, käyttäjätarinat, sprintin katselmointi, tuotejonon siivous)
- Scrumin kuvaama työtapa ei sovellu kaikille ryhmille, sillä se vaatii työntekijöiltä avoimuutta, rohkeutta ja kykyä toimia ryhmässä

Scrumilla on myöskin rajoituksensa. Normaalin, jokapäiväisen ylläpitotyön avuksi se ei sovi, sillä ylläpidossa pitää reagoida nopeasti jos jokin järjestelmä vikaantuu ja tällaisissa tilanteissa kaikki muut työt jäävät hetkeksi sivuun. Tämä ei luonnollisestikaan sovi yhteen Scrumin suunnittelupalaverien ja sprinttien kanssa. Scrumin vahvuus ylläpidossa onkin erilaisten, selkeästi rajattujen työtehtävien suorittamisessa. Onnistunut Scrum -projekti vaatii että työn kohde pitää ennalta tarkoin määritellä, ja projektin pitäisi pystyä täyttämään seuraavat ehdot:

- työryhmään tulee saada kaikkien tarvittavien osa-alueiden osaajat jolloin työnteko on sujuvaa
- työ voidaan jakaa toisistaan riippumattomiin kokonaisuuksiin joita voidaan tehdä rinnakkain
- työllä on selkeä alku ja loppu; kohde ei saa siis olla yleinen ylläpitotyö (tällaiseen esim. kanban tai scrum-ban saattaisi soveltua paremmin)

- työ pitää pystyä suunnittelemaan etukäteen; esimerkiksi ylläpidon äkillisiä hälytystehtäviä ei voida tehdä Scrumin avulla
- itse työ pitää olla selkeää ja kohtuullisen suoraviivaista; sprintit ovat tunnelmaltaan hyvin intensiivisiä eikä niiden ilmapiiri sovellu hyvin tutkiskelevaan ja mietiskelevään työtapaan vaan kannustaa enemmänkin suorittavaan työhön

Tutkimusryhmä ehti suorittaa neljä sprinttiä opinnäytetyön aikana. Arviolta käyttöönottoprojektin valmistuminen vaatisi vielä toiset neljä sprinttiä, jotka on tarkoitus aloittaa kesätauon jälkeen. Työtapa on yleisesti saanut ryhmän hyväksynnän ja tulevien Scrummattavien projektien hahmotteleminen on jo alkanut.

LÄHTEET

1 Tuckman, Bruce W. *Developmental Sequence in Small Groups.* Psychological Bulletin, Vol63(6), kesäkuu 1965, s. 384-399.

2 Tuckman, Bruce W ja Jensen, Mary Ann. *Stages of Small-Group Development Revisited.* Group & Organization Studies 2, joulukuu 1977, s. 419-427.

3 Belbin, Meredith R. *Management Teams: Why They Succeed or Fail., 3rd edition* Elsevier Ltd, 2010.

4 Hersey, Paul et al. *Organisaatiokäyttäytymisen perusteet.* Weilin+Göös, 1983.

5 Hersey, Paul et al. *Great Ideas Revisited: Revisiting the life-cycle theory of leadership.* Training and Development, Tammikuu 1996, s. 42-47.

6 Somech, Anit. *The Effects of Leadership Style and Team Process on Performance and Innovation in Functionally Heterogeneous Teams.* Journal of Management, 32-2006, s. 132-157.

7 Pentland, Alex. Harvard Business Review. *The New Science of Building Teams.* Saatavissa: https://hbr.org/2012/04/the-new-science-of-building-great-teams. Viitattu 30.1.2015.

8 Celkee Oy, TTL ry, Ohjelmistoyrittäjät Ry. 24.5.2013. *Tietojärjestelmien hankinta Suomessa 2013*. Saatavissa: http://www.ttlry.fi/sites/ttl.ttlry.mearra.com/files/Tietoj%C3%A4rjestelmien%20hankinta%20Suomessa%202013.pdf. Viitattu 7.10.2014.

9 The Standish Group. 1995. *CHAOS.* Saatavissa: https://net.educause.edu/ir/library/pdf/NCP08083B.pdf. Viitattu 7.10.2014.

10 Swartout, Paul. *Continous Delivery and DevOps: A Quickstart Guide.* Packt Publishing, 2012.

11 Cusumano, Michael A. 15.8.1995. *Beyond the Waterfall: Software Development at Microsoft*. Saatavissa: https://dspace.mit.edu/bitstream/handle/1721.1/2593/SWP-3844-33836288.pdf?sequence=1. Viitattu 7.10.2014.

12 Royce, Winston W. 1970. *Managing the Development of Large Software Systems*. Saatavissa: http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf. Viitattu 3.10.2014.

13 Schwaber, Ken. *SCRUM development process.* Saatavissa: http://navegapolis.net/files/Scrum_Development_Process.pdf. Viitattu 4.10.2014.

14 Beck, Kent et al. 2001. *Manifesto for Agile Software Development*. Saatavissa: http://agilemanifesto.org/. Viitattu 21.9.2014.

15 Rubin, Kenneth S. *Essential Scrum.* Addison Wesley, 2014.

16 Lekman consulting. *Suomenkielinen scrum-sanasto.* Saatavissa: https://scrumwell.files.wordpress.com/2012/01/suomenkielinen-scrum-sanasto-2012-v1-2.pdf. Viitattu 9.2.2015.

17 Kniberg, Henrik ja Skarin, Matias. *Kanban and Scrum: Making the best of both.* InfoQ, 2010.

18 Poppendieck, Mary et al. *Implementing Lean Software Development: From Concept to Cash.* Addison-Wesley Professional, 2006.

19 Virtuaali AMK. *Tutkimuksen validiteetti*. Saatavissa: http://www2.amk.fi/digma.fi/www.amk.fi/opintojaksot/0709019/1193463890749/1193464185783/1194413809750/1194415367669.html. Viitattu 27.2.2015.

20 Ahonen, Markus. *Tapaustutkimus: Soveltuuko Scrum vesiputousmallin korvaajaksi yrityksen sovelluskehitysprojekteihin?* Saatavissa: http://lib.tkk.fi/Dipl/2010/urn100203.pdf. Viitattu: 23.10.2014.

21 Vuori, Panu. 2014. *Ohjelmistoprojektinhallinan kehittäminen Scrum-menetelmällä*. http://www.theseus.fi/bitstream/handle/10024/79170/VUORI_PANU.pdf?sequence=1. Viitattu 19.1.2015.

22 Chantachaimongkol, Nichamon ja Sincharoenpanich, Puangpetch. *Critical factors for implementing the Scrum software development methodology.* Saatavissa: http://www.diva-portal.org/smash/get/diva2:605307/FULLTEXT01.pdf. Viitattu: 6.4.2015.

23 Ladas. Corey. *Scrum-ban.* Saatavissa: http://leansoftwareengineering.com/ksse/scrum-ban/. Viitattu: 6.3.2015.

Liite 1: Teemahaastattelujen aiherunko

Scrum yleisesti

- Aikaisempi kokemus ohjelmistokehitysprosesseista (ei ryhmähaastattelu)
- Opastus, ohjeistus
- Sprintin suunnittelupalaverit
    - töiden valinta, prosessin selkeys
    - työmääräarviointi
- Retrospektiivit
    - oman työn arviointimenetelmät
    - toimenpiteiden toteutus
- Omat vaikutusmahdollisuudet prosessiin
- Mitä muuttaisit, jatkaisitko toimintatapaa
- Aikataulutus, sprinttien syklitys

Ryhmän toiminta

- Ryhmänä työskentely
    - työnjako, kommunikointi
    - toiminta ryhmänä, itseohjautuvuus
- Ryhmän koostumus
    - muodostuminen, roolitus
    - alttius konflikteihin/konsensukseen
    - oma rooli ryhmässä (ei ryhmähaastattelu)
- Ryhmän johtaminen, kokemuksia