

Määrittelyn haasteet siirryttäessä vesiputousmallista ketteriin menetelmiin

Piia Lehtonen



Tekijä tai tekijät Piia Lehtonen	Ryhmätunnus tai aloitusvuosi HETI10SIM
Raportin nimi Määrittelyn haasteet siirryttäessä vesiputousmallista ketteriin menetelmiin	Sivu- ja liitesivumäärä 46 + 7
Opettajat tai ohjaajat Niina Kinnunen	
<p>Tämän opinnäytetyön tavoitteena on selvittää, mitä etuja ja haasteita määrittelyssä ketterillä menetelmillä on verrattuna vesiputousmalliin. Lisäksi opinnäytetyössä analysoidaan, miten valittu ohjelmistokehitysmenetelmä vaikuttaa määrittelyyn ja sen dokumentointiin ja vastaako ketterässä projektissa tehty dokumentaatio liiketoiminnan vaatimuksia.</p> <p>Opinnäytetyössä tarkastellaan räätälöityjen järjestelmien vaatimusmäärittelyä ja toiminnallista määrittelyä asiakkaan näkökulmasta. Ketteristä menetelmistä opinnäytetyöhön on valittu the Rational Unified Process ja Scrum.</p> <p>Opinnäytetyössä kuvataan ensin tarkasteltavia menetelmiä ja niiden suosiota. Seuraavaksi kerrotaan määrittelyn dokumentoinnista ja asiakaskokemuksesta tarkasteltavissa menetelmissä. Opinnäytetyön empiirisessä osassa kerrotaan keväällä 2013 puolistrukturoituna haastatteluna tehdystä kvalitatiivisesta eli laadullisesta kyselytutkimuksesta.</p> <p>Tutkimuksen tulosten mukaan ketteriä menetelmiä kannattaa käyttää, mutta ei kaikissa projekteissa. Parhaiten ketterät menetelmät soveltuvat pieniin ja innovointia vaativiin projekteihin. Menetelmä ei tutkimuksen mukaan vaikuta määrittelyyn ja dokumentointiin. Tutkimuksessa nousi esille myös menetelmien soveltamisen vaikeus ja osaamisen puute käytettäessä ketteriä menetelmiä.</p>	
Asiasanat määrittely, ketterät menetelmät, dokumentointi, ohjelmistokehitys	

Degree Programme in Information Technology

<p>Authors Piia Lehtonen</p>	<p>Group or year of entry HETI10SIM</p>
<p>The title of thesis The challenges of the analysis when changing waterfall model to agile methods</p>	<p>Number of report pages and attachment pages 46 + 7</p>
<p>Advisor(s) Niina Kinnunen</p>	
<p>This thesis examines the advantages and disadvantages of agile methods compared to waterfall method in analysis. In addition, the thesis analyses how the current software development method influences the analysis and documentation, and whether the documentation responds to the requirements of business specialists in an agile project.</p> <p>The thesis examines both requirement specification and functional specification of tailored systems from the customer's perspective. The thesis covers Rational Unified Process and Scrum agile methods.</p> <p>First, the thesis describes current methods and their popularity. Next, the thesis tells about the documentation of analysis and customer experience in current methods. The empirical section of the thesis is based on the qualitative semi structured study conducted in spring 2013.</p> <p>The results of the study show that agile methods should be used but not for all projects. The agile methods are suitable for small and innovation demanding projects. The method does not affect analysis and documentation. According to the results of the study, adapting the agile methods is difficult and there is a lack of knowledge when using agile methods.</p>	
<p>Key words analysis, agile methods, documentation, software development</p>	

Sisällys

1 Johdanto	1
1.1 Työn tavoitteet	2
1.2 Rajaukset.....	2
2 Tarkasteltavat ohjelmistokehitysmenetelmät	3
2.1 Vesiputousmalli.....	3
2.2 Ketterät menetelmät	4
2.2.1 RUP.....	6
2.2.2 Scrum	7
2.3 Menetelmien käyttö ja suosio.....	10
3 Määrittelyn dokumentointi ja asiakaskokemus tarkasteltavissa menetelmissä	13
3.1 Dokumentointi vesiputousmallissa.....	15
3.2 Dokumentointi ketterissä menetelmissä	15
3.2.1 RUP.....	17
3.2.2 Scrum	18
3.3 Asiakaskokemus tarkasteltavissa menetelmissä	19
4 Tutkimuksen toteutus ja tulokset	21
4.1 Toteutus ja haastateltavien esittely.....	21
4.2 Dokumentointi ja asiakaskokemus vesiputousmallissa	27
4.3 Dokumentointi ja asiakaskokemus RUP:ssa	29
4.4 Dokumentointi ja asiakaskokemus Scrumissa	31
4.5 Ketterien menetelmien soveltuvuus vastaajan organisaatioon	36
5 Yhteenveto ja johtopäätökset	38
Lähteet.....	43
Liitteet	47
Liite 1. Opinnäytetyössä käytetyt termit ja lyhenteet	47
Liite 2. Haastattelukysymykset.....	52

1 Johdanto

Ovatko ketterät menetelmät ohjelmistotuotannossa ohimenevä muoti-ilmiö vai moderni ja tehokas työtapa? Onko niistä vesiputousmallin korvaajaksi myös lakisääteisessä tai vaativassa tietojenkäsittelyssä kuten esimerkiksi julkisella sektorilla, pankissa tai vakuutusyhtiössä?

Ketterien menetelmien suosio kasvaa jatkuvasti, ja ne ovat saavuttaneet jo jähmeämätkin organisaatiot. Esimerkiksi maailman suurin työnantaja, Yhdysvaltain puolustusministeriö, ja osa julkisen sektorin organisaatioista Suomessa ovat siirtyneet ketteriin menetelmiin (Lekman 2012; Pajuoja, Huhtala, Pursiainen & Knuutila 2011, 18; Trafi 2012). Uutuudenviehätyksen lisäksi ketteriä menetelmiä houkuttelevat käyttämään niiden selkeät hyödyt. Ketterät menetelmät vauhdittavat projektien etenemistä, näyttävät nopeasti työn tulokset sekä säästävät aikaa ja rahaa (Kolehmainen, 2013; Lekman 2013; Ojanperä 2012).

Määrittelyllä ja dokumentoinnilla on perinteisesti ollut tärkeä merkitys tehtäessä liiketoiminnan kannalta kriittisiä tai lainsäädäntöön perustuvia järjestelmiä. Hyvä dokumentaatio antaa kokonaiskuvan, kuvaa tarvittavat yksityiskohdat kuten laskentakaavat ja lainsäädännön vaatimukset, helpottaa kattavaa testausta sekä tukee ylläpitotyötä koko järjestelmän elinkaaren ajan. Toisaalta tarkka määrittely ja dokumentointi vievät paljon aikaa ja lisäävät kustannuksia. Haluan opinnäytetyössäni selvittää, helpottavatko ketterät menetelmät työlästä määrittelyvaihetta ja minkälaisia haasteita niiden käytössä voi määrittelylle olla. En usko ketterien menetelmien estävän hyvän dokumentaation tekemistä, mutta haluan tietää, mitä ketteryys käytännössä tarkoittaa esimerkiksi määrittelyssä räätälöityä järjestelmää vakuutusyhtiölle tai sähköistä palvelua valtion viraston asiakkaiden käyttöön. Minua kiinnostaa, miten dokumentoidaan ketterästi ja voisiko dokumentoinnissa päästä vesiputousmallia helpommalla. Lisäksi haluan selvittää, miten asiakas kokee ketterien menetelmien käytön.

Opinnäytetyön keskeiset termit ja lyhenteet kuvataan liitteessä (liite 1). Scrumin osalta opinnäytetyössä käytetään suomenkielisen Scrum-sanaston termejä (Suomenkielinen Scrum-sanasto 2012).

1.1 Työn tavoitteet

Opinnäytetyön tavoitteena on selvittää, mitä etuja ja haasteita määrittelyssä ketterillä menetelmillä on verrattuna vesiputousmalliin. Lisäksi opinnäytetyössä analysoidaan, miten valittu menetelmä vaikuttaa määrittelyyn ja sen dokumentointiin ja vastaako ketterässä projektissa tehty dokumentaatio liiketoiminnan vaatimuksia.

1.2 Rajaukset

Opinnäytetyössä tarkastellaan räätälöityjen järjestelmien määrittelyä ja dokumentointia asiakkaan näkökulmasta. Määrittelyllä tarkoitetaan sekä vaatimusmäärittelyä että toiminnallista määrittelyä eli sitä dokumentaatiota, jonka työstämisessä liiketoiminnan edustajat ovat tiiviisti mukana, jonka he hyväksyvät, johon testaus perustuu ja jota hyödynnetään järjestelmän ylläpidossa sen koko elinkaaren ajan. Dokumentaation osalta opinnäytetyön ulkopuolelle rajataan käyttäjien ohjeistus, suunnittelun dokumentaatio ja muu tekninen dokumentaatio, käyttöpalveluiden (esim. operointi) ohjeistukseksi tehty dokumentaatio sekä dokumenttien katselmointi.

Opinnäytetyön ulkopuolelle rajataan lisäksi projektinhallinta, kustannukset, ostaminen, kilpailutus sekä toimittajien ja asiakkaiden väliset sopimukset. Opinnäytetyössä ei myöskään oteta kantaa siihen, kannattaako käyttää vesiputousmallia vai ketteriä menetelmiä, vaan ainoastaan analysoidaan vesiputousmallin ja valittujen ketterien menetelmien eroavaisuuksia ja käytössä huomioitavia seikkoja määrittelyn ja dokumentoinnin osalta.

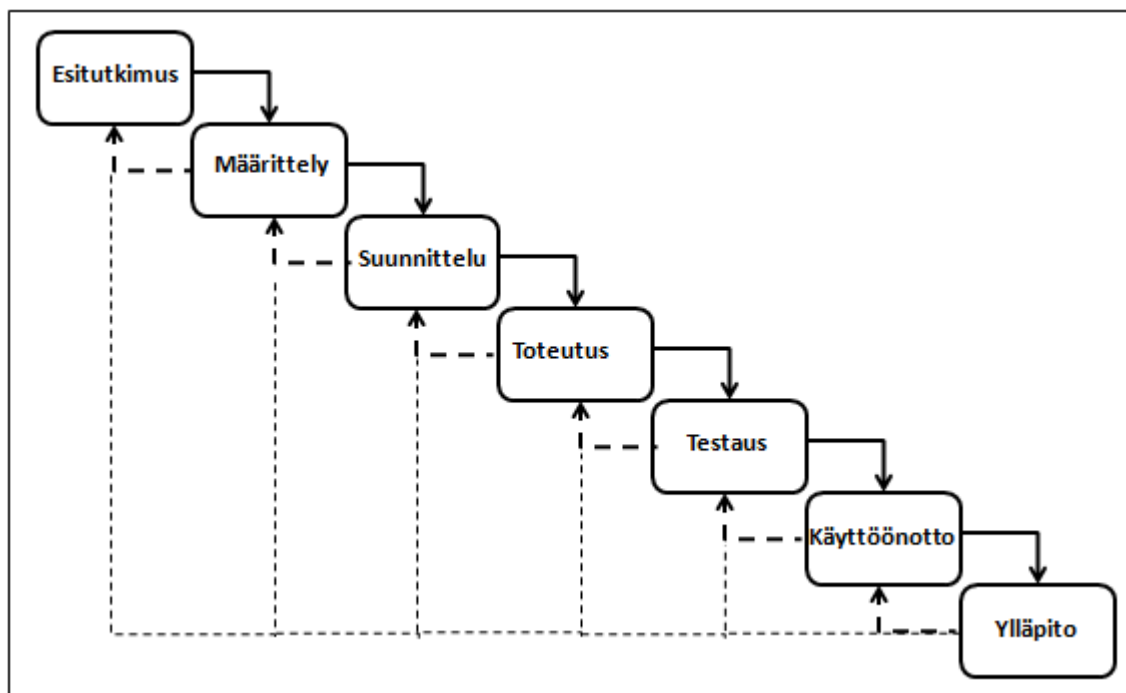
Opinnäytetyössä tehtävää tutkimusta varten haastateltavat valitaan asiakkaan puolelta julkisen sektorin ja vakuutusalan organisaatioista, koska näiden toimialojen järjestelmissä määrittelyllä ja dokumentoinnilla on perinteisesti ollut tärkeä rooli. Ketteristä menetelmistä opinnäytetyössä tarkastellaan Scrumia ja RUP:ia, koska ne ovat Suomessa yleisesti käytettyjä, keskenään erilaisia ja tuttuja opinnäytetyössä tehdyn tutkimuksen kohdeorganisaatioissa.

2 Tarkasteltavat ohjelmistokehitysmenetelmät

Opinnäytetyössä tarkastellaan ohjelmistokehitysmenetelmistä vesiputousmallia, RUP:a ja Scrumia. Näistä vesiputousmalli on tunnetuin ja käytetyin elinkaarimalli, ja RUP ja Scrum kuuluvat ketteriin menetelmiin (Abrahamsson, Salo, Ronkainen & Warsta 2002, 18; Pohjonen 2002, 40).

2.1 Vesiputousmalli

Vesiputousmalli on eteenpäin kulkeva prosessi, jossa tietojärjestelmän kehittämisen vaiheet seuraavat suoraviivaisesti toisiaan esitutkimuksesta aina ylläpitoon saakka (kuvio 1). Vesiputousmallin edeltävissä prosessivaiheissa tehtyjen virheiden korjaaminen on hankalaa, koska se tarkoittaa yleensä prosessissa peruuttamista ja edeltävien vaiheiden uusimista. Ongelmaa pahentavat tarkistuspisteiden ja dokumenttien kiinnittäminen vaiheiden rajapinnoille, jolloin edellisen vaiheen dokumentaatio toimii syötteenä seuraavalle vaiheelle. Vesiputousmallin huonona puolena on myös se, että asiakas saa tuloksia nähtäväkseen varsin myöhäisessä vaiheessa järjestelmäkehitystä. (Pohjonen 2002, 40.)



Kuvio 1. Vesiputousmalli (Pohjonen 2007, 40)

McConnellin (2002, 136) mukaan puhdas vesiputousmalli toimii kuitenkin hyvin projekteissa, joissa kehitettävän tuotteen määrittely on selkeä ja tekniset menetelmät hyvin ymmärrettyjä. Tällöin vesiputousmalli tarjoaa vakaan vaatimusmäärittelyn ja auttaa löytämään virheet mahdollisimman varhaisessa ja edullisessa vaiheessa projektia. Esimerkiksi tehtäessä hyvin määriteltyä päivitystä tuotannossa olevaan järjestelmään tai siirrettäessä ohjelmisto uudelle laitealustalle, vesiputousmalli voi olla sopiva valinta myös nopeaan kehittämiseen. (McConnell 2002, 136.)

Vesiputousmalli on saanut alkunsa Roycen vuonna 1970 kirjoittamasta, väärinymmärretyistä artikkelista. Iteroinnin puute ja seuraavan vaiheen aloittaminen vasta edellisen päätyttyä eivät olleet Roycen alkuperäisen ajatuksen mukaisia. Todellisuudessa Royce piti taaksepäin iterointia tärkeänä ja ajatteli, että järjestelmä kannattaa tehdä kahdesti, jos vastaavaa järjestelmää ei ole aikaisemmin tehty. Royce korosti, että kustannukset voivat kaksinkertaistua, jos ongelmat havaitaan vasta testausvaiheessa. Hyvin monen projektimallin, jopa Scrum-mallin, sisältä paljastuu Roycen alkuperäinen idea jossain muodossa. (Haikala & Mikkonen 2011, 37.)

2.2 Ketterät menetelmät

Ketterät menetelmät (agile) –termi syntyi vuonna 2001, kun joukko kevyempien menetelmien kehittäjiä ja kannattajia kokoontui Utahiin, Yhdysvaltoihin keskustelemaan käyttämistään menetelmistä. Kokouksen aikana sovittiin ketterien menetelmien periaatteista, perustettiin Agile Alliance-järjestö ja julkaistiin ketterien menetelmien ”peruskirja” Agile Manifesto. Agile Manifeston mukaan tärkeintä ohjelmistokehityksessä ovat tyytyväinen asiakas ja toimiva ohjelmisto. (Haikala & Mikkonen 2011, 43–45.)

Tänä päivänä Agile Alliance on kansainvälinen, voittoa tavoittelematon järjestö, joka edistää ketterien menetelmien käyttöä ja kehittämistä (Agile Alliance 2013a) ja Agile Manifesto on käännetty yli 40 kielelle (Agile Alliance 2013b). Suomessa ketterien ohjelmistokehitysmenetelmien käyttöä ja tunnettavuutta edistää Agile Finland ry (Agile Finland ry).

Agile Manifeston (2011) mukaan ketterän ohjelmistokehityksen 12 periaatetta ovat:

- Asiakas pysyy tyytyväisenä saadessaan projektin alusta alkaen säännöllisesti uusia ohjelmaversioita.
- Vaatimuksia saa muuttaa tarvittaessa, jopa ohjelmiston myöhäisessäkin kehitysvaiheessa.
- Uusia, toimivia ohjelmistoversioita julkaistaan säännöllisesti ja usein (julkaisujen väli parista viikosta pariin kuukauteen).
- Liiketoiminnan edustajat ja ohjelmistokehittäjät työskentelevät yhdessä päivittäin.
- Projektit rakennetaan motivoituneiden yksilöiden ympärille, heille annetaan tarvittava ympäristö ja tuki ja heihin luotetaan.
- Tehokkain kommunikointitapa (sekä kehitystiimin sisällä että muiden yhteistyökumppaneiden kanssa) on keskustella kasvokkain.
- Toimiva ohjelmisto on edistymisen tärkein mittari.
- Projektissa työskentelevien työtahti ja työkuorma pidetään tasaisena.
- Jatkuva huomion kiinnittäminen parhaaseen tekniseen laatuun ja hyvään suunniteluun parantaa ketteryyttä.
- Pyritään yksinkertaisuuteen ja minimoidaan vähemmän tärkeät työt.
- Itseohjautuvissa tiimeissä syntyvät parhaat arkkitehtuurit, vaatimukset ja suunnitelmat.
- Tiimin tulee kehittää jatkuvasti omaa tehokkuuttaan ja toimintaansa.

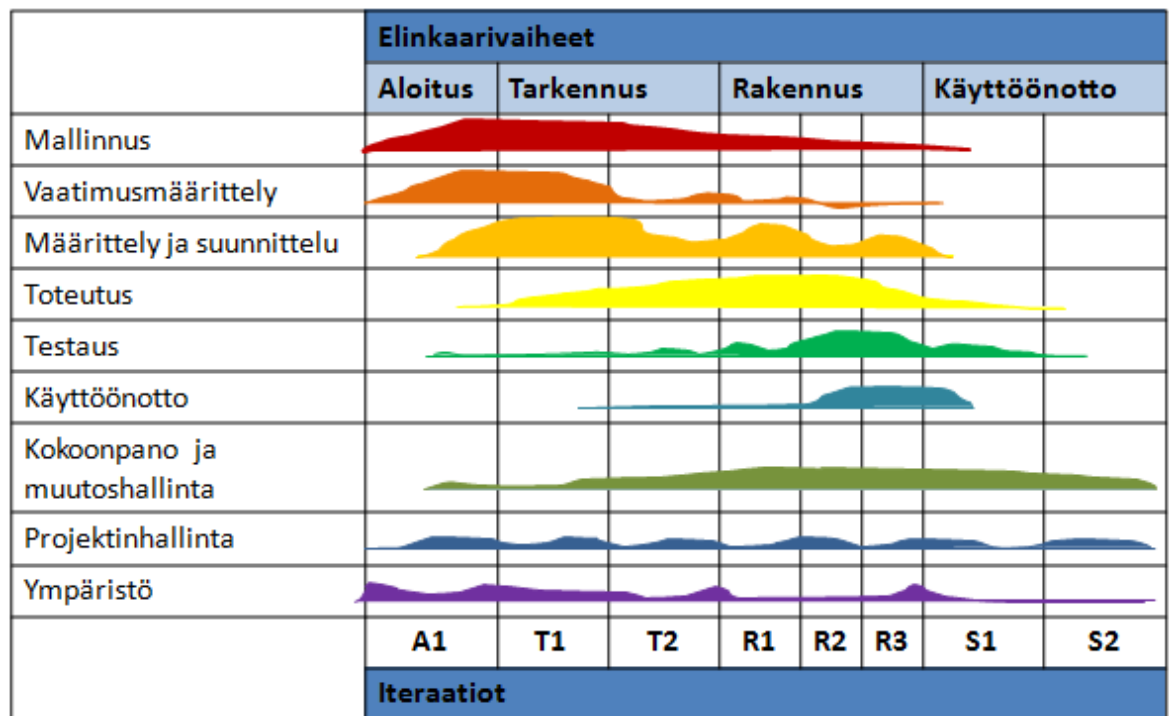
Haikala ja Mikkonen (2011, 45) pitävät Agile Manifeston periaatteita helposti hyväksyttävänä tietyin rajauksin. He arvioivat, että niiden lähtökohtana on pienehkö asiakasprojekti, ja periaatteita voi olla vaikea soveltaa esimerkiksi laajempiin tuotekehityshankkeisiin (Haikala & Mikkonen 2011, 45).

Opinnäytetyössä tarkastellaan ketteristä menetelmistä Scrumia ja RUP:a (the Rational Unified Process). Niiden lisäksi ketteriä menetelmiä ovat esimerkiksi Extreme Programming, Crystal family of methodologies, Feature Driven Development, Adaptive Software Development ja Dynamic Systems Development Method (Abrahamsson ym. 2002, 18).

2.2.1 RUP

RUP on 1990-luvulla Rational-nimisen yrityksen kehittämä ohjelmistokehityksen prosessikehikko. RUP:sta tuli osa IBM:n Rational Method Composer –tuotetta, kun IBM osti Rationalin. RUP on sisällöltään hyvin laaja ja se on tarkoitettu räätälöitäväksi tilanteen mukaan. (Haikala & Mikkonen 2011, 42.)

RUP kattaa useita modernin ohjelmistokehityksen parhaita käytäntöjä. Tärkeimmät niistä ovat: Kehitä iteratiivisesti, hallinnoi vaatimuksia, käytä komponenttipohjaista arkkitehtuuria, mallinna visuaalisesti, tee jatkuvaa laadunvarmistusta ja kontrolloi muutoksia. (Kruchten 2004, 18.)



Kuvio 2. RUP-vaiheistus (IBM 2007, 3)

RUP:n peruseriaatteita ovat käyttötapausohjattu lähestymistapa ja arkkitehtuurikeskeisyys. Käyttötapaukset kuvaavat mitä toiminnallisuutta järjestelmään tarvitaan ja ohjaavat ohjelmistokehitystyötä. Arkkitehtuuria kehitetään eri näkökulmia hyödyntäen varhaisesta prosessinvaiheesta alkaen ja tarkennetaan työn edistyessä. (Kruchten 2004, 26; Kruchten 2004, 29–30.) RUP tukee iteratiivista ja inkrementaalista työtappaa, jolloin

ohjelmisto rakentuu koko ajan laajentamalla. RUP koostuu neljästä elinkaarivaiheesta: Aloitus (Inception), Tarkennus (Elaboration), Rakennus (Construction) ja Käyttöönotto (Transition) (kuviot 2). Jokainen elinkaarivaihe sisältää yhden tai useamman iteraation, ja jokaisessa iteraatiossa toistuvat samat työvaiheet hiukan eri painotuksilla. Esimerkiksi aloitus- ja tarkennusvaiheissa tehdään enemmän liiketoiminnan mallinnusta ja määrittelyä, kun taas toteutus ja testaus painottuvat rakennusvaiheeseen. Vesiputousmalliin verrattuna yhdessä RUP:n iteraatiossa tehdään kaikkia vesiputousmallin vaiheita. (Haikala & Mikkonen 2011, 42–44; IBM 2007, 3.) Opinnäytetyössä tarkasteltavat vaatimusmäärittely ja toiminnallinen määrittely liittyvät ensisijaisesti RUP:n työvaiheisiin mallinnus, vaatimusmäärittely sekä määrittely ja suunnittelu (Kroll & Kruchten 2006, 288).

Käytännön tekemisen pohjana RUP:ssa on kolme keskeistä kokonaisuutta: roolit, tehtävät ja tuotokset (Kruchten 2004, 51). Rooli ei ole yksilö tai titteli, vaan se kuvaa osaamista, jota tarvitaan tehtävien suorittamiseen ja tuotosten tekemiseen. Esimerkiksi roolilla testaaaja voi olla tehtävä ”laadi testaussuunnitelma”, jonka tuotoksena syntyy testaussuunnitelma. Yksi ihminen voi olla samaan aikaan useassa roolissa ja jokainen tehtävä on nimetty tietylle roolille. (Kruchten 2004, 36–38.)

RUP:n etuna muihin menetelmiin verrattuna on, että se kattaa koko ohjelmistokehitysprosessin, myös ohjelmistokehityksen alkuosan vaiheet, ja sitä tukemaan on tarjolla erilaisia kaupallisia välineitä (Abrahamsson ym. 2002, 92). Yleisesti ottaen RUP:a ei pidetä erityisen ketteränä ohjelmistokehitysmenetelmänä, mutta sitä on mahdollista käyttää myös ketterällä tavalla. Ketterän RUP:n onnistuminen vaatii kuitenkin käyttäjältään taitavaa räätälöintiä. (Abrahamsson ym. 2002, 60–61.)

2.2.2 Scrum

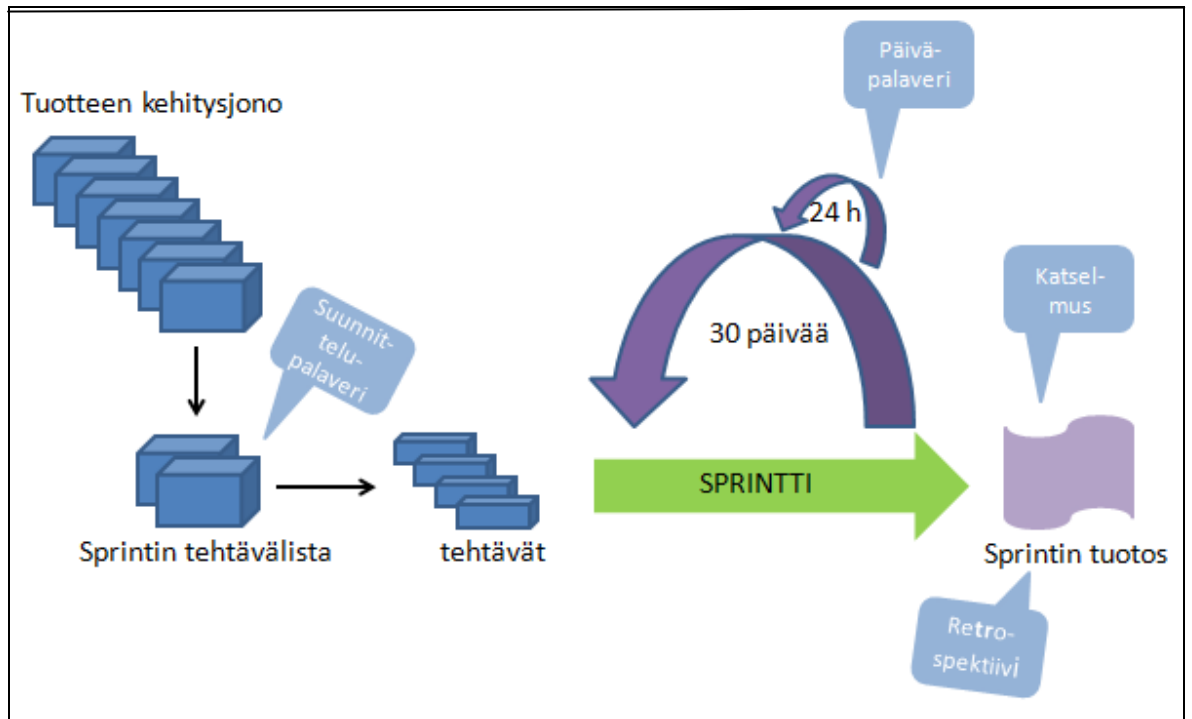
Scrum on pitkälti Jeff Sutherlandin ja Ken Schwaberin 1990-luvun alkupuolella luoma ohjelmistokehitysmenetelmä, joka pohjautuu Japanissa edellisellä vuosikymmenellä kehitettyihin periaatteisiin. Scrum on siis saanut alkunsa jo kauan ennen kuin ketteryydestä tuli muotia. (Haikala & Mikkonen 2011, 46–47.) Yleispätevänä viitekehysenä

Scrum sopii periaatteessa minkä tahansa kompleksisen työn ohjaamiseen, jossa tavoitteet voidaan kiinnittää vähintään viikoksi (Lekman 2011, 35).

Scrum hyödyntää iteratiivis-inkrementaalista lähestymistapaa, ja sen peruselementit ovat Scrumtiimi rooleineen, säännöt, tapahtumat ja tuotokset. Lisäksi tärkeitä ovat Scrumin kolme tukijalkaa: läpinäkyvyys, tarkastelu ja sopeuttaminen. Läpinäkyvyys konkretisoituu niin, että prosessin kannalta oleelliset tekijät – kuten sanasto ja ”valmiin” määritelmä – sovitaan yhdessä, jotta kaikilla on niistä yhteinen näkemys. Scrumin tuotoksia ja työn edistymistä tarkastellaan säännöllisin väliajoin, jolloin haitalliset poikkeamat havaitaan. Sopeuttaminen tarkoittaa prosessin tai käytettävien materiaalien säätämistä silloin, kun tarkastelija havaitsee jonkun prosessin osan olevan hyväksyttävien raja-arvojen ulkopuolella. Tarkastelua ja sopeuttamista voidaan tehdä sprintin suunnittelupalaverissa, päiväpalaverissa, sprinttikatselmuksessa ja sprintin retrospektiivissä. (Schwaber & Sutherland 2011, 3–4.)

Scrumissa on kolme eri roolia: tuoteomistaja, scrummaster ja kehitystiimi. Tuoteomistaja vastaa työn tuloksesta ja arvosta sekä tuotteen kehitysjonon hallinnasta. Scrummaster vastaa siitä, että kaikki ymmärtävät ja käyttävät oikein Scrumia sekä auttaa tarvittaessa tuoteomistajaa, kehitystiimiä ja koko organisaatiota Scrumin soveltamisessa. Kehitystiimi koostuu asiantuntijoista, jotka kehittävät tuotetta kehitysjonon sisällön mukaisesti ja kehitysjonon osoittamassa järjestyksessä. Tiimit päättävät itse – ilman ulkoista ohjausta – miten parhaiten tekevät työnsä. Tiimit sisältävät kaiken tarvittavan osaamisen, eikä niillä ole riippuvuuksia tiimin ulkopuolisiin henkilöihin. (Schwaber & Sutherland 2011, 4–6.)

Tuotteen kehitysjono on järjestetty lista kaikista tuotteen vaatimuksista, ominaisuuksista, toiminnoista, korjauksista ja parannuksista. Jokainen kehitysjonon kohta sisältää vähintään järjestyksen, kuvauksen ja työmääräarvion. Kehitysjono muuttuu sitä mukaa, kun tuote kehittyy ja työ etenee. Sprintin tehtävälista koostuu kyseiseen sprinttiin valituista tuotteen kehitysjonon kohdista ja suunnitelmasta niiden toteuttamiseksi. (Schwaber & Sutherland 2011, 11–13.)



Kuvio 3. Scrum-prosessi (Haikala & Mikkonen 2011, 48; Schwaber & Sutherland 2011, 7–11).

Scrumin ydin on enintään kuukauden mittainen sprintti, jonka aikana tuotetaan julkaisukelpoinen versio tuotteesta (kuvio 3). Jokainen sprintti sisältää kehitystyön lisäksi sprintin suunnittelupalaverin sprintin alussa, päiväpalaverit sekä sprinttikatselmuksen ja sprintin retrospektiivin sprintin lopussa. Seuraava sprintti alkaa heti edellisen päätyttyä. Scrumin sääntöjen mukaan on oleellista, että sprintin aikana ei saa tehdä muutoksia sprintin tehtäviin tai kehitystiimin koostumukseen eikä laatutavoitteita lasketa. Tuotemistaja voi keskeyttää sprintin, jos sen tavoite muuttuu tarpeettomaksi tai toteuttaminen ei ole enää järkevää. Sprintin suunnittelupalaverissa suunnitellaan yhdessä sprintin aikana tehtävä työ, sprinttikatselmuksessa tarkastellaan tehtyä tuoteversiota ja sprintin retrospektiivissä analysoidaan sprintin onnistumista ja työskentelyä. Enintään 15 minuuttia kestävä päiväpalaveri pidetään jokaisena työpäivänä ja sen aikana jokainen kehitystiimin jäsen kertoo vuorollaan mitä on tehnyt edellisen päiväpalaverin jälkeen, mitä aikoo tehdä seuraavaksi ja onko hänen etenemiselleen ongelmia. (Schwaber & Sutherland 2011, 7–11.)

Scrumin etuna on sen yksinkertaisuus: Scrumin periaatteet on helppo selittää ja oppia nopeasti. Scrum ei ota kantaa kaikkiin ohjelmiston elinkaareen liittyviin tehtäviin, käy-

tettyihin kehitysmenetelmiin tai työkaluihin, ja kärjistettynä sanoen Scrum on projektin toteutusvaiheeseen tarkoitettu tapa organisoida projektin iteraatiot. (Haikala & Mikkonen 2011, 47.) Scrumia voidaan kuitenkin varsin hyvin käyttää yhdessä jonkun muun ohjelmistokehitysmenetelmän, esimerkiksi Extreme Programmingin (XP), kanssa tai vaikka RUP:n rakennusvaiheessa (Haikala & Mikkonen 2011, 53; Lekman 2011, 35).

2.3 Menetelmien käyttö ja suosio

Tietojärjestelmien kehittämisessä käytettävien ohjelmistokehitysmenetelmien historia on suhteellisen lyhyt, vaikka se sisältää useita erilaisia ja varsin kokeellisiakin menetelmiä. 40-vuotisen historiansa aikana ohjelmistokehitysmenetelmät ovat yhdistyneet, hajaantuneet ja lainailleet toisiltaan jopa kokonaisia osia. (Paananen 2005, 350.)

Ohjelmistokehitysmenetelmää käytetään harvoin aivan oppikirjan mukaisena. Usein yritys ottaa kokeiltavakseen jonkun tunnetun ja suositun menetelmän ja ajan mittaan muokkaa siitä omaan organisaatioonsa soveltuvan version. Ketteriä menetelmiä käyttävissä yrityksissä käytössä on usein yrityksen omiin tarpeisiin sovellettu versio Scrumista, jostain muusta ketterästä menetelmästä tai yhdistelmä useasta ketterästä menetelmästä. Moni yritys sekoittaa omaan sovellettuun versioonsa sekä ketteriä että ei-ketteriä menetelmiä. (Haikala & Mikkonen 2011, 52; Pohjonen 2002, 71; West & Grant 2010, 9.)

Kaiken kaikkiaan ohjelmistotuotannolle on tyypillistä, että mikään toimintatapa ei automaattisesti ole oikea, vaan yleensä täytyy soveltaa tilanteen mukaan. Uusia ideoita ei kannata tyrmätä kokeilematta, vaikka tietynlainen skeptisyys ja terveen järjen käyttö onkin suotavaa. Uusista suuntauksista voi poimia myös toimivia piirteitä, vaikka kokonaista menetelmää ei ottaisikaan käyttöön. (Haikala & Mikkonen 2011, 28.)

Ketteriä periaatteita, käytäntöjä ja menetelmiä on valtava määrä, ja organisaation täytyy itse pystyä valitsemaan mitä se niistä käyttää. Valinnat on tärkeää osata tehdä organisaation omasta näkökulmasta, organisaation toimintaympäristö ja ketteryyden kehitysaste huomioiden. Epäonnistumisen riski on suuri, jos organisaatio yrittää toteuttaa liian radikaaleja muutoksia tai edistyneitä menetelmiä. (Sytyke ry & Agile Finland ry 2013, 39.)

Opinnäytetyössä tarkasteltavista menetelmistä vesiputousmalli on ehdottomasti pitkäikäisin, koska sitä on käytetty jo 1970-luvun alkupuolelta saakka (Haikala & Mikkonen 2011, 37). Sekä RUP että Scrum ovat olleet käytössä 1990-luvulta alkaen, joten kumpaakaan niistä ei voida pitää kovin uutena menetelmänä (Haikala & Mikkonen 2011, 42; Schwaber & Sutherland 2011, 3).

Jokaista tarkasteltavista menetelmistä käytetään edelleen aktiivisesti. Kansainvälisen Forrester Research Inc. tutkimuslaitoksen vuonna 2010 julkaiseman raportin mukaan 35 % yrityksistä käyttää ketteriä menetelmiä, 31 % ei käytä mitään menetelmää, 21 % käyttää iteratiivisia menetelmiä ja 13 % käyttää vesiputousmallia. Raportin perusteella ketteristä menetelmistä suosituin on Scrum, jota käyttää 11 % yrityksistä. Iteratiivisista menetelmistä RUP:n osuus on ainoastaan 3 %. (West & Grant 2010, 2.)

VersionOnen vuoden 2013 alussa julkaiseman vuosittaisen Agile development survey – tutkimuksen mukaan ketteriä menetelmiä käyttää peräti 84 % yrityksistä. Ketteriä menetelmiä ei kuitenkaan käytetä kaikissa projekteissa: Tutkimukseen vastanneista yrityksistä puolet käyttää ketteriä menetelmiä kuitenkin useammassa kuin joka toisessa projektissa. (VersionOne 2013, 3.) Myös VersionOnen tutkimuksen mukaan ylivoimaisesti suosituin ketteristä menetelmistä on Scrum. Scrumia ja sen variaatioita käyttää 72 % tutkimukseen vastanneista yrityksistä. VersionOnen tutkimuksessa ei kysytty vesiputousmallin tai RUP:n käytöstä, mutta Agile Unified Process –menetelmää ilmoitti käyttävänsä prosentti vastanneista. (VersionOne 2013, 5.)

Myös Suomessa ketterien menetelmien suosio kasvaa jatkuvasti. Vuonna 2012 julkaistun Oulun yliopistossa tehdyn ”Survey on Agile and Lean Usage in Finnish Software Industry” – tutkimuksen mukaan ketteriä menetelmiä tai Lean-menetelmää käyttää 58 % kyselyyn vastanneista. Organisaatioista, joissa ei vielä käytetty ketteriä menetelmiä tai Lean-menetelmää, 16 % suunnitteli aloittavansa niiden käytön seuraavan kahden vuoden kuluessa. Ketteristä menetelmistä ylivoimaisesti suosituin myös Suomessa on Scrum, jota käytti 83 % vastanneista. Tässäkään tutkimuksessa ei kysytty RUP:n käytöstä. Kysely tehtiin Tietotekniikan liiton jäsenille ja siihen vastasi 408 henkilöä 200 eri organisaatiosta. (Rodriguez, Markkula, Oivo & Turula 2012, 142–143; Rodriguez ym. 2012, 146.)

Suomessa ketterät menetelmät ovat saaneet jalansijaa jo julkisella sektorillakin. Maanmittauslaitos, Maa- ja metsätalousministeriön tietopalvelukeskus Tike ja useat muut virastot käyttävät tai suunnittelevat siirtymistä ketterään hanke- ja kehitysmalliin. (Lekman 2012.) Liikenteen turvallisuusvirasto Trafi tiedotti marraskuussa 2012, että sen uusi järjestelmäkokonaisuus tehdään ketteriä menetelmiä käyttämällä. Kyseessä on yksi ensimmäisistä valtionhallinnon puhtaasti ketteriin menetelmiin perustuvista kilpailutuksista. (Trafi 2012.) Teknologian ja innovaatioiden kehittämiskeskus Tekesillä on jo pidempi kokemus ketterien menetelmien käyttämisestä Tekesin tietojärjestelmien ylläpidossa ja jatkokehityksessä yhteistyössä Solitan kanssa (Pajuoja ym. 2011, 18).

3 Määrittelyn dokumentointi ja asiakaskokemus tarkasteltavissa menetelmissä

Haikalan ja Mikkosen (2011, 30) mukaan määrittelyllä tarkoitetaan asiakasvaatimusten ja niistä johdettujen toimintojen dokumentointia asiakasnäkökulmasta. Pohjonen (2002, 31) jakaa määrittelyn vaatimusmäärittelyyn ja määrittelyyn eli järjestelmänalyysiin. Vaatimusmäärittelyä voi edeltää esitutkimus, jossa selvitetään edellytykset tietojärjestelmän kehittämiseksi (Pohjonen 2002, 27).

Vaatimusmäärittelyssä kuvataan eri sidosryhmien järjestelmälle asettamat toiminnalliset ja ei-toiminnalliset vaatimukset. Toiminnalliset vaatimukset määrittelevät mitä järjestelmän pitäisi tehdä ja ei-toiminnalliset vaatimukset kuvaavat erilaisia järjestelmän reunaehtoja kuten esimerkiksi käytettävyys- tai kapasiteettivaatimuksia. Vaatimuksissa huomioidaan myös rakennettavaan järjestelmään liittyvät erilaiset ulkoiset tekijät kuten lainsäädäntö, standardit ja asetukset. Dokumentoitujen vaatimusten tulisi olla keskenään ristiriidattomia sekä helposti luettavissa, tulkittavissa ja ymmärrettävissä. Vaatimusmäärittelystä pitäisi löytyä kuvaus kehittämishankkeen toimeksiannosta, mahdolliset lisäselvitykset, yleiskuvaukset järjestelmän nykytilasta ja tavoitetilasta sekä kaikki priorisoidut ja numeroidut toiminnalliset vaatimukset, ei-toiminnalliset vaatimukset ja rajoitteet. (Pohjonen 2002, 28–31.)

Määrittelyvaiheessa selvitetään mitä järjestelmän pitäisi tehdä. Määrittely tehdään analysoimalla vaatimusmäärittelyn vaatimukset ja johtamalla niistä järjestelmän toiminnallinen määrittely. Määrittelystä pitäisi löytyä yleiskuvaukset järjestelmän tarkoituksesta ja toiminnasta, kuvaukset järjestelmän ympäristöstä, käyttäjistä, rajoitteista, toiminnoista, tiedoista, tietokannoista, rajapinnoista, järjestelmään ja sen käyttöön liittyvistä riippuvuuksista ja oletuksista sekä määrittelyt järjestelmän suorituskyvystä, virhetilanteista toipumisesta, käytettävyydestä ja turvallisuudesta. (Pohjonen 2002, 31–32.)

Määrittelydokumentaatiota hyödynnetään suunnitteluvaiheessa kuvattaessa miten järjestelmä toteutetaan sekä testauksen suunnittelussa ja käyttöohjeistuksen laatimisessa (Pohjonen 2002, 32). Kattava ja jatkuva määrittelydokumentaatio auttaa myös järjestelmän elinkaaren pisimmässä yksittäisessä vaiheessa, ylläpidossa, ja voi auttaa säästä-

mään huomattavasti ylläpitokustannuksissa (Pohjonen 2002, 37–38). Ylläpitovaihetta ei kannata väheksyä, koska jopa kaksi kolmasosaa tietojärjestelmien kehittämisestä on tuotannossa olevien järjestelmien ylläpitoa: korjauksia, muutoksia ja jatkokehitystä, ja ainoastaan yksi kolmasosa uusien järjestelmien kehittämistä (Pohjonen 2002, 18).

Pahimmillaan dokumentoinnin laiminlyönti järjestelmää rakennettaessa ja ylläpidettäessä saattaa johtaa siihen, että järjestelmän ymmärtäminen on mahdotonta ilman sen alkuperäisiä kehittäjiä ja ylläpitäjiä. Jos heitä ei enää ole käytettävissä, ei järjestelmää enää voida uusia, vaikka järjestelmä olisi organisaation toiminnan kannalta keskeinen ja tullut elinkaarensa päähän. (Pohjonen 2002, 19.)

Dokumentointi on usein tietojärjestelmien kehittämisen heikoimpia lenkkejä, koska kiireessä siitä on helppo tinkiä. Tarvittavan dokumentaation määrä ja laatu pitäisi kuitenkin ratkaista aina kehitystyön koon ja monimutkaisuuden perusteella. Määrä ei korvaa laatua ja toisaalta pienessä kehitysprojektissa voidaan pärjätä vähäiselläkin dokumentaatiolla. (Paananen 2005, 355–356.)

Jokaisella rakennettavalla järjestelmällä tulisi olla tietty perusdokumentaatio. Omien dokumenttimallien määrittely helpottaa dokumentointia ja yhdenmukaistaa organisaation dokumentteja. Jokaiselle tietojärjestelmän kehityksessä syntyvälle dokumentille määrätään perusrunko, jota noudatetaan niin pitkälle kun mahdollista, sekä lyhyet soveltamisohjeet. (Pohjonen 2002, 79–80.)

Tärkein määrittelytapa on aina luonnollinen kieli. Määrittely pitäisi olla aina ymmärrettävä ja yksiselitteinen, vaikka käytännössä määrittelyn aukkoja ja epätasällisyyksiä korjataankin projektin edetessä. Myös erilaiset ohjelmistosta piirrettävät kaaviot tehostavat kommunikointia eri osapuolten kesken. Määrittelydokumenteja kirjoittaessa pitäisi huomioida, että sama dokumentti voi myös asiakkaan puolella kiinnostaa hyvin erilaisia henkilöitä kuten esimerkiksi ohjelman pääkäyttäjää, tulevia käyttäjiä ja tietohallintopäällikköä. Jos dokumentissa edetään yleiskuvauksesta kohti yksityiskohtia, voi jokainen lukija yleiskuvauksen perusteella päätellä oman osuutensa ja keskittyä sen läpikäyntiin. Yhden asian kirjaaminen yhteen paikkaan helpottaa toteutusta ja muutosten tekemistä dokumentaatioon. Kaikkeaa mahdollista dokumentaatiota ei kannata ylläpitää ja sen

vuoksi on tärkeää erotella projektin aikaiset ja ylläpidettävät osuudet. Tätä voidaan helpottaa esimerkiksi käyttämällä liitteitä. (Haikala & Mikkonen 2011, 69–71.)

Hyvän vaatimusmäärittelyn kirjoittaminen ei ole helppoa. Yksi vaatimusmäärittelyn suurimpia haasteita on sen rooli liiketoiminnan ja teknologian yhteisenä kielenä. Vaatimukset täytyy kuvata niin, että liiketoiminnasta vastaavat ymmärtävät ja osaavat priorisoida niitä. Vaatimukset eivät kuitenkaan saa jäädä liian liiketoimintakeskeisiksi, jotta myös kehittäjät ymmärtävät ja osaavat toteuttaa vaatimukset. (Teikari 2012, 16–17).

3.1 Dokumentointi vesiputousmallissa

Dokumentaatio on dokumenttiohjautuvassa vesiputousmallissa tärkeässä roolissa. Tärkeimmät vesiputousmallin vaiheesta toiseen kulkevat tuotokset ovat dokumentteja ja ne kuvaavat myös työn edistymistä. (McConnell 2002, 136.)

McConnellin (2002, 139) mukaan vesiputousmallin käyttö nopean kehittämisen projekteissa voi dokumentaation näkökulmasta olla haastavaa, koska vesiputousmalli voi vaatia liikaa dokumentaatiota ja dokumentaation ylläpitoon voi kulua suhteettoman paljon aikaa. Suurin osa ongelmista johtuu kuitenkin vesiputousmallin erillisistä ja peräkkäisistä vaiheista. Pahimmat heikkoudet voidaan korjata muuntamalla mallia niin, että dokumentaatiopainotusta vähennetään, muutetaan vaiheet osittain päällekkäisiksi ja helpotetaan taaksepäinmenoa. (McConnell 2002, 143.)

3.2 Dokumentointi ketterissä menetelmissä

Ketterien menetelmien käyttäminen ei millään tavalla estä dokumentaation tekemistä ja on puhdas väärinymmärrys, että ketterissä menetelmissä ei dokumentoitaisi ollenkaan. Ketterissä menetelmissä ymmärretään hyvin, että dokumentit tukevat kommunikointia ja yhteistyötä, auttavat tiedonsiirrossa ja järjestelmien jatkokehityksessä, säilyttävät historiallista tietoa sekä täyttävät lainsäädännön vaatimukset. Ketterien menetelmien ideologian mukaan pääasia ei kuitenkaan ole dokumentoida, vaan ymmärtää. On tärkeää, että kehittäjät ymmärtävät mitä asiakas haluaa. Ja aivan yhtä tärkeää on, että asiakas, testaajat, ylläpitäjät ja tulevat käyttäjät ymmärtävät mitä kehittäjät ovat rakentamassa. Pelkkä dokumentaatio ei kuitenkaan aina riitä, jotta eri osapuolet ymmärtävät toisiaan.

Myös Agile Manifesto pitää toimivaa tuotetta tärkeämpänä kuin kattavaa dokumentaatiota. Hyviä ideoita kannattaa testata käytännössä mahdollisimman varhaisessa vaiheessa, jotta saadaan varmuus niiden toteutuskelpoisuudesta. (Highsmith 2010, 287.)

Teikarikin (2013, 17–18) korostaa, että loppukäyttäjien tarpeet eivät yleensä selviä riittävän hyvin kysymällä, ja ohjelmiston kokeileminen on ainoa keino varmistaa, että ohjelmisto todella palvelee käyttäjien tarpeita. Kokeilemisen taustalla on tosiasia, että emme kuitenkaan osaa ennustaa ja opimme sitä mukaa kun teemme. Toistuvalla kokeilemisella saadaan usein palautetta, ja korjausliikkeitä voidaan tehdä nopeasti, helposti ja edullisesti. (Teikari 2012, 20.)

Paras ymmärrys syntyy sopivasta yhdistelmästä dokumentaatiota ja ihmisten välistä vuorovaikutusta. On tärkeää ymmärtää, että dokumentaatio ei koskaan voi korvata aitoa vuorovaikutusta ja kommunikaatiota. Esimerkiksi hiljaisen tiedon siirtämisessä kasvokkain tapahtuvan keskustelun on havaittu olevan dokumentaation lähettämiseen verrattuna huomattavasti tehokkaampi muoto siirtää tietoa. (Highsmith 2010, 288–289.)

Dokumentaatio on tärkeää mitoittaa projektin, projektiryhmän koon, toimialan ja organisaation mukaan. Dokumentaatio, joka on suurelle tiimille korvaamaton, voi olla ajan haaskausta hyvin pienelle tiimille. (Highsmith 2010, 290.) Asian toinen puoli on kuitenkin, että kun dokumentaation määrä kasvaa kehitystiimin koon kasvaessa, ketteryys vähenee (Abrahamsson ym. 2002, 95–96).

Dokumentoinnissa kannattaa suosia yleiskuvauksia ja välttää kaikkien yksityiskohtien tarkkaa kuvaamista. On tärkeää myös tiedostaa, mitä dokumentaatiota ylläpidetään projektin jälkeen ja mitä ei. (Highsmith 2010, 290.)

Määrittelyssä vain tarkemmin tunnetut ominaisuudet kannattaa määrittellä tarkemmin ja muut aluksi vain karkeammalla tasolla. Ketterien menetelmien periaatteiden mukaisesti määrittelyt tarkentuvat jatkuvasti, ja kehitystyön pohjaksi tehty määrittely on vasta ensimmäinen arvaus. Ominaisuus on määritelty riittävän tarkasti silloin, kun liiketoiminnan edustajat osaavat priorisoida sen suhteessa muihin ominaisuuksiin, ja kehittäjät ymmärtävät mitä ovat toteuttamassa. (Teikari 2012, 29; Teikari 2012, 32.)

Kannattaa miettiä kriittisesti mitä dokumentaatiota juuri tässä tilanteessa tarvitaan, ja millä menetelmillä ja välineillä dokumentaatiota tehdään. Dokumentaatio on myös työväline, joka auttaa ajatusten jäsentelyssä ja keskustelussa. Dokumentaatiota tehdessään on hyvä huomioida lukijakunta ja miettiä toimiiko sille paremmin visuaalinen kuvaus, kuten esimerkiksi graafit ja kuvat, proosa vai niiden yhdistelmä. (Sytyke ry & Agile Finland ry 2013, 89.)

Ketterissä menetelmissä suositaan epämuodollista ja interaktiivista dokumentointia kuten esimerkiksi digikuvien, fläppitaulun tai wiki-sivujen käyttöä (Highsmith 2010, 290). Myös Internet-pohjaiset dokumentoinnin apuvälineet, esimerkiksi Google Docs tai Confluence, ovat suosittuja (Haikala & Mikkonen 2011, 195).

3.2.1 RUP

Ohjelmistokehityksessä käytettävien välineiden lisäksi IBM tarjoaa RUP:n käyttäjälle kattavan ohjeistuksen, laajan valikoiman mallipohjia, konsepteja ja tarkistuslistoja. Laajasta kokoelmasta voi valikoida haluamansa osuudet organisaation ja projektin tarpeiden mukaan. Organisaatio voi myös räätälöidä IBM:n tarjoaman materiaalin pohjalta omat soveltamisohjeensa ja dokumentaationsa. (IBM 2007, 2–3.)

RUP:n mukaan ohjelmistokehityksessä voi syntyä erilaisia ja erimuotoisia tuotoksia kuten malleja (esimerkiksi käyttötapausmalli), mallin elementtejä (esimerkiksi käyttötapaus), dokumentteja (esimerkiksi sovellusarkkitehtuuridokumentti) ja teknisempiä tuotoksia kuten esimerkiksi lähdekoodi. RUP suosittelee, että tuotosten kirjaamisessa ja ylläpitämisessä käytettäisiin sopivia välineitä paperidokumenttien tuottamisen sijaan. Välineiden käyttö tehostaa työtä, tieto on aina ajan tasaista ja välineistä on mahdollista tarvittaessa generoida dokumentteja. (Kruchten 2004, 40–42.)

Vaatimusmäärittelyn ja toiminnallisen määrittelyn kuvaamiseen RUP tarjoaa lukuisia tuotoksia: esimerkiksi kohdeorganisaation kuvaus, liiketoiminnan visio-dokumentti, sanasto, käyttötapausmalli, käyttötapaus, käyttäjäroolit, liiketoimintasäännöt, vaatimusten kuvaukset, käyttöliittymän prototyyppi ja navigaatiokartta (Kruchten 2004, 277–

278). Sen lisäksi että organisaatio on päättänyt mitä RUP:n osuuksia ja tuotoksia käyttää, täytyy jokaisen projektin päättää omien tarpeidensa pohjalta mitä tuotoksia se tuottaa ja missä muodossa (Kroll & Kruchten 2003, 184). Yleinen virhe otettaessa RUP:aa käyttöön on, että valitaan RUP:n laajasta valikoimasta liian paljon tuotoksia ja tehtäviä. Tällöin työskentely hidastuu, joustavuus kärsii, kustannukset kasvavat ja projektin jäsenet käyttävät liian paljon energiaa erilaisten tuotosten hiomiseen sen sijaan, että rakentaisivat oikeaa järjestelmää. Jokaiseen projektiin kannattaa ottaa mukaan vain sellaiset osuudet, joista on aidosti hyötyä. Toisaalta liian vähäinen tuotosten määrä, niiden huono laatu tai sopimaton dokumentointitapa voivat johtaa ongelmiin tiedonkulussa. (Kroll & Kruchten 2003, 244–245.)

Siirtyminen vesiputousmallista RUP:n iteratiiviseen työtapaan ei myöskään aina käy mutkattomasti, vaan se vaatii muutoksia kaikkien projektin jäsenten työ- ja ajattelutapaan. Tyypillisiä vaatimusmäärittelyyn ja toiminnalliseen määrittelyyn liittyviä virheitä ovat kokonaisuuden piilottaminen hajottamalla käyttötapausmalli liian suureksi määräksi pieniä käyttötapauksia, juuttuminen yksityiskohtiin, sidosryhmien hyväksynnän puuttuminen vaatimuksista ja teknisten yksityiskohtien sisällyttäminen vaatimuksiin. (Kroll & Kruchten 2003, 261–262.)

3.2.2 Scrum

Yhteisen tavoitteen kirkastamiseksi ja kommunikoinnin helpottamiseksi sidosryhmien, johdon ja Scrumtiimin välillä voidaan tehdä visio ja tiekartta. Visioon kirjataan lyhyesti liiketoiminnan näkökulmasta korkeantason näkemys siitä, minkälainen kehitettävän järjestelmän pitäisi olla tai mitä projektissa pitäisi tehdä. Visiossa kuvataan myös miksi järjestelmä kannattaa tehdä ja mitä lisäarvoa se tuottaa liiketoiminnalle ja asiakkaille. Vision pohjalta laadittu tiekartta sen sijaan hahmottelee karkealla tasolla minkälaisen julkaisujen tai virstanpylväiden kautta tavoitteeseen päästään. (Woodward, Surdek & Ganis 2010, 49–50.)

Scrumissa tuotteen kehitysjono sisältää kaiken mitä ohjelmistossa tarvitaan, ja on ainoa lähde ohjelmistoon tehtäville vaatimuksille ja muutoksille (Schwaber & Sutherland 2011, 11). Viime kädessä kehitysjono on kuitenkin vain tehtävien lista, jolle voi laittaa

sellaisia tehtäviä, jotka kyseisessä tilanteessa ovat tarpeen. Kehitysjonot voivat olla muodoltaan hyvin monenlaisia, ja ne voivat sisältää erikokoisia – pieniä ja suuria – tarinoita tai riittävän pieniä alkioita, joita pystytään nopeasti toimittamaan. Kehitysjonon alkiot voivat olla mitä vaan: käyttötapausten skenaarion askelia, käyttöliittymän ruutuja tai yksityiskohtaisen määrittelyn kappaleita. (Sytyke ry & Agile Finland ry 2013, 66.)

Tuotteen kehitysjonon lisäksi Scrum ei ota kantaa miten ja missä muodossa järjestelmän vaatimuksia voisi kuvata. Moni Scrumtiimi kuitenkin käyttää käyttäjätarinoita vaatimusten kuvaamisessa, koska ne auttavat ymmärtämään toiminnallisuutta ja sen tuottamaa hyötyä käyttäjälle. Käyttäjätarinassa kerrotaan usein kuka tekee, mitä tekee, miksi tekee ja miten toiminnallisuus varmistetaan. Käyttäjätarinaaan lisätään lyhyet muistiinpanot kun kehitystiimi käy sen keskustellen läpi tuoteomistajan kanssa. (Woodward ym. 2010, 3.)

Toiminnallisuutta voidaan kuvata myös käyttötapauksilla tai käyttäjätarinoiden ja käyttötapausten yhdistelmillä kehitettäessä vaativia, kuten esimerkiksi lääketieteellisiä tai tarkkaa jäljitettävyyttä edellyttäviä, järjestelmiä (Woodward ym. 2010, 52). Kirjoitettua dokumentaatiota tarvitaan enemmän myös silloin, jos kehitystiimin tapaamisissa käytetään kieltä, joka ei ole kaikkien tiimin jäsenten äidinkieli, tai kehitystiimi työskentelee eri paikoissa tai eri aikavyöhykkeillä. Tällöin dokumentaatio tehostaa kommunikointia ja ymmärtämistä. (Woodward ym. 2010, 123.)

3.3 Asiakaskokemus tarkasteltavissa menetelmissä

Ohjelmistokehityksessä kaikkein tärkeintä on tehdä asiakkaiden ja loppukäyttäjien tarpeiden mukaisia tuotteita. Edullinen hinta tai nopeasti tehty tuote ei auta, jos tuote on vääränlainen. Usein asiakassuuntautuneet organisaatiot ovat päässeet eroon myös monista kehittämiseen liittyvistä ongelmista. (McConnell 2002, 234.)

Vesiputousmalli tarjoaa vakaan vaatimusmäärittelyn, mutta konkreettisia tuloksia on nähtävissä vasta hyvin myöhäisessä vaiheessa ohjelmistokehitystä. Projektin alussa on kuitenkin vaikea määrittellä kaikkia vaatimuksia kokonaisuudessaan ja usein vasta lopul-

lisen tuotteen kokeileminen ja näkeminen auttavat huomaamaan kaikki tarpeet. (McConnell 2002, 136–138.)

Ketterissä menetelmissä asiakas on keskeisessä roolissa, mikä korostuu myös Agile Manifestossa. Agile Manifesto pyrkii selkeästi pureutumaan vesiputousmallin ongelmiin ja parantamaan asiakaskokemusta toimittamalla säännöllisiä ohjelmaversioita, antamalla mahdollisuuden muuttaa vaatimuksia tarvittaessa myös myöhäisessä kehitysvaiheessa ja toisaalta myös vaatimalla asiakkaan ja kehittäjien tiivistä yhteistyötä. (Agile Manifesto 2001.) Ketterässä ohjelmistokehitysprojektissa asiakas ei ole pelkkä tilaaja, vaan tärkeä osa innovaatiotyötä ja tuotekehittelyä. Perinteiseen tilaaja-toteuttaja –malliin verrattuna asiakas osallistuu ketterässä projektissa tuotteen kehittämiseen useammalla tasolla. (Teikari 2012, 49.)

RUP:ssa jatkuvan palautteen saamisen asiakkaalta varmistavat myös käyttötapaohjattu lähestymistapa ja iteratiivinen kehitys. Käyttötapaohjaukset helpottavat asiakkaan ja kehittäjien välistä kommunikointia ja muistuttavat asiakkaan näkökulmasta koko ohjelmistokehityksen ajan. (Kroll & Kruchten 2003, 31.) Scrumissa tuoteomistajan rooli on keskeinen myös asiakaskokemuksen kannalta (Schwaber & Sutherland 2011, 4–5).

4 Tutkimuksen toteutus ja tulokset

Tutkimuksen tavoitteena oli selvittää, mitä etuja ja haasteita räätälöityjen järjestelmien vaatimusmäärittelyssä, toiminnallisessa määrittelyssä ja niiden dokumentoinnissa ketterillä menetelmillä on verrattuna vesiputousmalliin. Lisäksi tutkimuksessa kartoitettiin miten valittu menetelmä vaikuttaa määrittelyyn ja sen dokumentointiin, vastaako ketterässä projektissa tehty dokumentaatio liiketoiminnan vaatimuksia ja soveltuvatko ketterät menetelmät vastaajien organisaatioihin. Tutkimuksessa tarkasteltaviksi menetelmiksi oli valittu vesiputousmalli, RUP ja Scrum.

4.1 Toteutus ja haastateltavien esittely

Tutkimus oli luonteeltaan kvalitatiivinen kyselytutkimus ja se suoritettiin puolistrukturoituna haastatteluna (Hirsjärvi & Hurme 2009, 47; Hirsjärvi, Remes & Sajavaara 2009, 164). Tutkimusmenetelmän valintaan vaikutti eniten sen soveltuvuus tähän nimenomaiseen opinnäytetyöhön. Alun perin kyselytutkimus oli tarkoitus tehdä sähköisessä muodossa, mutta haastattelun uskottiin antavan laajempia ja syvällisempiä vastauksia. Kyselytutkimuksessa vastaukset voivat jäädä pinnallisiksi, vastaajat voivat ymmärtää kysymykset eri tavalla kuin niiden laatija on tarkoittanut, eivätkä vastaajat aina jaksaa kirjoittaa laajoja vapaita vastauksia. Haastatteleamalla on mahdollista saada laajempia vastauksia ja pyytää tarkennuksia vastauksiin. (Hirsjärvi & Hurme 2009, 35–37; Hirsjärvi ym. 2009, 205–206.)

Haastateltavien valinta tehtiin rajaamalla ensin kohderyhmään soveltuvat organisaatiot ja sen jälkeen etsimällä organisaatioista sopivia henkilöitä. Haastateltavat organisaatiot valittiin julkiselta sektorilta ja vakuutusosalta, koska näiden toimialojen järjestelmissä määrittelyllä ja dokumentoinnilla on perinteisesti ollut tärkeä rooli, jotta ohjelmat toimivat täysin lain kirjaimen ja toimialan ohjeistuksen mukaan ja ohjelmien ylläpito on helppoa. Em. toimialoilla järjestelmien elinkaari on useimmiten pitkä ja niihin tehdään paljon ylläpitotyötä, esimerkiksi lainmuutosten takia.

Haastateltavien lukumääräksi täsmentyi kolme, koska sitä useamman henkilön haastattelu olisi ollut liian työlästä ja aikaa vievää, ja toisaalta kolme asiantuntevaa henkilöä

riittää kattamaan opinnäytetyön vaatimukset. Haastateltavat valittiin kolmesta eri organisaatioista, jotta tutkimukseen saataisiin erilaisia näkemyksiä kuitenkin keskenään samantyyppisistä organisaatioista. Haastateltaviksi valittiin henkilöitä, joilla oli kokemusta räätelöityjen järjestelmien vaatimusmäärittelystä ja toiminnallisesta määrittelystä sekä ainakin osasta tarkasteltavia menetelmiä. Käytin haastateltavien valinnassa omia kontaktejani. Yhden haastatelluista tunsin ennen haastattelua hyvin, toisen olin tavannut muutaman kerran ja kolmas oli minulle ennestään vieras. Jokaisen organisaation liiketoiminnasta minulla oli kokemusta joko työni kautta tai asiakkaan näkökulmasta.

Haastatteluista ja niiden tarkasta ajankohdasta sovittiin etukäteen. Haastateltavia oli pyydetty varaamaan haastattelulle aikaa kaksi tuntia. Haastateltaville oli haastattelusta sovittaessa kerrottu lyhyesti opinnäytetyöstä ja sen rajauksista sekä haastattelun aihepiiristä. Kysymyslomake lähetettiin kaikille haastateltaville etukäteen tutustuttaviksi 9.4.2013 (liite 2). Kysymyksiin ei ollut annettu valmiita vastausvaihtoehtoja, vaan niihin odotettiin vastattavan omin sanoin. Haastateltavia ohjeistettiin ennen haastattelua, ettei heidän tarvitse valmistautua haastatteluun millään tavalla, eikä heidän ole välttämätöntä vastata kaikkiin kysymyksiin.

Haastattelut tehtiin yksilöhaastatteluina 11.4.2013, 17.4.2013 ja 18.4.2013. Kaksi haastattelua tehtiin haastateltavien työpaikalla kokoushuoneessa ja yksi rauhallisessa, tarkoitukseen soveltuvassa kahvilassa. Haastattelutilanteessa haastateltavilta kysyttiin etukäteen lähetyn kysymyslomakkeen kysymykset. Taustatietojen osalta kysyttiin myös tarkentavia kysymyksiä, jotta saatiin kattavampi kuva haastateltavasta ja organisaatiosta. Muutoin kysymykset noudattivat muotonsa ja järjestyksensä osalta etukäteen lähetettyä kysymyslomaketta, eikä uusia kysymyksiä esitetty. Joidenkin vastausten osalta kysyttiin tarkennuksia ymmärtämisen varmistamiseksi ja perusteluja. Varattu aika riitti haastatteluun jokaisen haastateltavan osalta. Keskimäärin yksi haastattelu kesti 1,5 tuntia.

Haastatteluista tehtiin muistiinpanot haastattelutilanteessa kirjaamalla vastaukset käsin paperille. Tallennusmenetelmä valittiin, koska se on luotettava, eikä sen käyttäminen ollut liian työlästä. Valintaan vaikutti myös, ettei tutkimuksen luonteen takia ole tarpeen saada tallennettua vastauksia sanatarkasti eikä esimerkiksi äänenpainoilla tai naurahduksilla ole merkitystä. (Hirsjärvi & Hurme 2009, 139–140; Hirsjärvi ym. 2009, 222.)

Jokainen haastattelu purettiin käsinkirjoitetuista muistiinpanoista sähköiseen muotoon samana iltana kun haastattelu oli tehty, jotta haastattelun sisältö oli vielä mahdollisimman tuoreena muistissa. Jokaiselle haastateltavalle tarjottiin mahdollisuutta tarkistaa sähköiseen muotoon puretut muistiinpanot. Kaksi haastateltavaa halusi tarkistuksen tehdä ja heille lähetettiin muistiinpanot haastattelua seuraavana päivänä tarkistettavaksi. Toinen lisäsi muistiinpanoihin muutaman lauseen ja toinen hyväksyi muistiinpanot sellaisenaan.

Haastateltavista käytetään nimiä vastaaja A, vastaaja B ja vastaaja C, koska haastateltaville oli luvattu, ettei heidän nimiään tai organisaation tarkkaa nimeä julkaista opinnäytetyössä. Yksi haastateltava erikseen toivoi, ettei hänen edustamaansa organisaatiota pystyisi päättelemään opinnäytetyöstä. Tämän takia tutkimustulosten yhteydessä ei ole yksilöity kuka tarkalleen ottaen vastasi mitään.

Haastateltavat työskentelevät eri organisaatioissa, mutta organisaatioiden liiketoiminta on tietojärjestelmien näkökulmasta hyvin samantyyppistä. Kaikilla organisaatioista on räätälöityjä järjestelmiä, sähköisiä palveluja ja valmisohjelmistoja.

Vastaaja A työskentelee kehittämisspällikkönä ja tiiminvetäjänä julkisen sektorin organisaatiossa, jonka toiminta perustuu lainsäädäntöön. Vastaaja A on työskennellyt organisaatiossa noin yhdeksän vuotta arkkitehtuuritöissä, projektispällikkönä, ylläpidon koordinaattorina ja määrittelijänä pääasiassa räätälöityjen järjestelmien parissa. Vastaaja A:n työ organisaatiossa on painottunut ohjelmistokehityksen alkuvaiheeseen - esiselvitykseen, vaatimusmäärittelyyn ja määrittelyyn – mutta vastaaja on osallistunut myös muihin vaiheisiin aina käyttöönoton jälkeiseen ylläpitoon saakka.

Organisaatiossa on joka vuosi muutama sata IT-projektia, joissa tehdään sekä ylläpitoa että kehitetään uusia räätälöityjä järjestelmiä ja sähköisiä palveluja. Organisaation noin 6000 työntekijästä 600 – 700 työskentelee täyspäiväisesti IT-projekteissa. Lisäksi IT-projekteihin osallistuu lukuisia muita työntekijöitä esim. liiketoiminnan asiantuntijan roolissa. Organisaatio tekee lähes kaiken IT-työn itse. Keskuskoneen ylläpito on ulkoistettu ja toimittajilta ostetaan konsultaatiota ja organisaatiolle soveltuvia valmisratkaisuja

sekä niiden integrointia ja räätälöintiä. Omaan ydinliiketoimintaansa kuuluvat järjestelmät organisaatio tekee itse.

Vastaajalla A on kokemusta vesiputousmallista, RUP:sta ja Scrumista. Sekä vesiputousmallista että RUP:sta organisaatiolla on käytössä oma sovellettu versionsa. Lisäksi joissakin, lähinnä sähköisiin palveluihin liittyvissä projekteissa, on vesiputousmallia sovellettu hyvin vapaasti kunkin projektin tarpeiden mukaisesti. Scrumia organisaatiossa ei ole käytetty kovinkaan paljon, mutta joitakin kokeiluja on ollut. Esimerkiksi organisaation Internet-sivuja tehtäessä käytettiin menetelmänä Scrumia.

Nykyisin laajasti käytössä oleva RUP-pohjainen menetelmä on valittu hyvin välinelähtöisesti organisaatioon hankittujen Rational-tuotteiden seurauksena. RUP-pohjainen menetelmä on ollut käytössä yli viisi vuotta, ja sen käytön alkuvaiheessa menetelmäosaamista ostettiin ulkopuoliselta toimittajalta ja tehtiin erilaisia pilotointeja räätälöityessä ja otettaessa menetelmää käyttöön. Tulevaisuudessa organisaatio aikoo siirtyä ketterämpään suuntaan ja saada ohjelmistokehitykseen sen avulla lisää tehokkuutta. Ketteryys voi tarkoittaa myös ketterämpää RUP:ia. Scrumistakin on organisaatiossa keskusteltu ja sitä on mahdollista kokeilla pienissä projekteissa.

Vastaaja A epäilee, että määrittely ja dokumentointi eivät vaikuta menetelmän valintaan. Nykyisen RUP-pohjaisen menetelmän taipaleellakin on ollut alkukankeutta ja monenlaista kehitysvaihetta. Organisaatiossa on esimerkiksi kokeiltu hyvin väline- ja luokkamallikeskeistä dokumentointitapaa, mutta siitä on luovuttu, ja palattu perinteisempään dokumentointiin. Menetelmän käyttöönotto ja markkinointi henkilöstölle eivät sujuneet parhaalla mahdollisella tavalla.

Vastaaja B työskentelee tietojärjestelmäasiantuntijana vakuutusyhtiössä, jonka toiminta perustuu lainsäädäntöön ja muihin säädöksiin. Vastaaja B on työskennellyt organisaatiossa noin kuusi vuotta. Hän työskentelee nykyisin pääasiassa uusien järjestelmien kehityshankkeissa ja -projekteissa, joissa hänen tehtäviään ovat olleet yhden pitkän Scrum-projektin product owner, testauksen koordinointi, toimittaja-yhteistyö, järjestelmien integrointi, esiselvitys ja vaatimusmäärittely. Vastaaja tekee myös ylläpidon koordinointia ja on aikaisemmin ollut paljon tekemisissä organisaation tietokantojen kanssa.

Organisaatiossa on vuodessa keskimäärin viisi ylläpitoprojektia ja muutama projekti, jossa kehitetään uusia räätälöityjä järjestelmiä ja sähköisiä palveluja. Organisaation 250 työntekijästä 15 työskentelee täyspäiväisesti IT-projekteissa. Lisäksi IT-projekteihin osallistuu lukuisia muita työntekijöitä esim. liiketoiminnan asiantuntijan roolissa. Organisaatio vastaa itse esiselvityksestä, vaatimusmäärittelystä ja hyväksymistestauksesta. Määrittelystä ja sen jälkeisistä ohjelmistokehityksen vaiheista vastaavat toimittajat. Organisaatio on kuitenkin hyvin aktiivisesti mukana erityisesti määrittelyssä ja suunnittelussa. Monitoimittajaympäristössä käytännöt vaihtelevat jonkin verran toimittajittain, riippuen järjestelmästä ja toimittajan kanssa tehdystä sopimuksesta.

Vastaajalla B on kokemusta vesiputousmallista ja Scrumista. Toistaiseksi toimittajalle on annettu vapaus valita menetelmä, mutta tulevaisuudessa organisaatio haluaa enemmän sananvaltaa menetelmän valinnassa. Tulevaisuudessa organisaatiossa aiotaan käyttää vesiputousmallia ja ketteriä menetelmiä. Seuraava iso uuden räätälöidyn järjestelmän kehityshanke tehdään vesiputousmallilla, koska se koetaan helpommaksi hallita kiinteähintaisessa hankkeessa. Vesiputousmalli toimii myös omana ja toimittajan suojana, jotta päästään yhteisymmärrykseen siitä mitä tehdään. Ylläpitoa tehdään yhdessä toimittajan kanssa aika lailla iteroiden ja ketterästi, mutta ilman mitään tiettyä menetelmää.

Vastaajan B mielestä määrittely ja dokumentointi vaikuttavat menetelmän valintaan aika paljon, koska organisaatiolla on kovat vaatimukset dokumentaation suhteen. Eri toimittajat tuottavat erinäköisiä ja eritasoisia dokumentteja, eikä se sinänsä haittaa. Oleellista kuitenkin on, että dokumentaatiosta näkyy, että tiedetään mitä ollaan tekemässä. Master-dokumentit täytyy olla organisaatiolla itsellään. Toimittaja voi halutessaan käyttää dokumentoinnissa myös vapaampia välineitä, kuten esimerkiksi wiki-sivuja. Turhia rajoituksia toimittajille ei haluta antaa, mutta kuvaukset täytyy tehdä yleisesti käytössä olevilla välineillä, jotta niiden ylläpidossa ja käytössä ei myöhemmin tule ongelmia.

Vastaaja C työskentelee projektipäällikkönä julkisen sektorin organisaatiossa, jonka toiminta perustuu lainsäädäntöön. Vastaaja C on työskennellyt organisaatiossa 9,5 vuotta ja sitä ennen IT-tehtävissä yksityisellä sektorilla vuodesta 1985 alkaen. Vastaaja C työskentelee liiketoiminnan kehittämissuhteissa projektipäällikkönä sekä ohjeistaa,

opastaa ja antaa menettelytukea vaatimusmäärittely- ja määrittelytyöhön. Vastaja C on tehnyt myös itse vaatimusmäärittely- ja määrittelytyötä sekä vetänyt vaatimusmäärittely- ja määrittelyprojekteja. Vastaja C osallistuu sekä ylläpito että kehitysprojekteihin, joissa on tehty uusi räätälöityjä järjestelmiä ja sähköisiä palveluja.

Organisaatiossa on meneillään noin 50 projektia, joista suurin osa on ylläpitoprojekteja. Organisaatiossa työskentelee noin 5300 työntekijää, joista IT-tehtävissä noin 200 – 300 henkilöä. Tuki- ja tuotantotehtävien lisäksi IT-henkilöstöä työskentelee liiketoiminnan kehittämissä projekteissa. Projekteihin osallistuu myös lukuisia liiketoiminnan asiantuntijoita. Organisaatio vastaa itse esiselvityksestä, vaatimusmäärittelystä, hyväksymisestä, käyttöönotosta, ja niiden tekemiseen voi tarvittaessa ostaa lisäresursseja toimittajilta, jos omat resurssit eivät riitä. Työnjako toimittajien kanssa perustuu sopimukseen. Toimittajien vastuulla ovat määrittelyn tarkennus tai tekninen määrittely, suunnittelu, toteutus ja testaus.

Organisaatiossa on käytetty vesiputousmallia sekä ”ketterämpää vesiputousmallia” eli vesiputousmallia, johon on projektikohtaisesti sovellettu Scrumin ja ketterien menetelmien piirteitä. Joissakin vesiputousmallin mukaan tehdyissä projekteissa on voinut tunnustaa myös RUP:n piirteitä.

Käytetyt menetelmät ovat vastaajan mukaan historian painolastia: Tehdään niin kuin on aina tehty, eikä ole oikein ollut aikaa miettiä muita vaihtoehtoja. Menetelmän valinta tehdään yhteistyössä toimittajien kanssa ja heidänkin kanssaan on pitkän yhteistyön aikana käytetty vanhaa, totuttua yhteistyömallia. Projektista ja projektipäälliköstä riippuen on voitu kokeilla jotakin ketterämpää.

Tulevaisuudessakin vanhojen järjestelmien ylläpidon osalta jatketaan vanhalla tavalla. Organisaatiossa valmistellaan isoa hanketta, jossa suuri määrä räätälöityjä järjestelmiä tullaan vaihtamaan valmisohjelmistoon. Hankkeen alussa tullaan valitsemaan myös käytettävä menetelmä. Vastaja C uskoo, että määrittely ja dokumentointi vaikuttavat menetelmän valintaan, koska määrittely on organisaation omalla vastuulla. Toimittajalakin on osuutta asiaan, toimittajan vinkit huomioidaan ja lopullinen valinta tehdään yhteistyössä.

4.2 Dokumentointi ja asiakaskokemus vesiputousmallissa

Vastaajilla oli eniten kokemusta vesiputousmallista ja jokainen heistä oli käyttänyt vesiputousmallia nykyisessä organisaatiossaan. Tutkimuksen tulokset vesiputousmallin osalta perustuvat kaikkien vastaajien haastatteluihin.

Taulukko 1. Vaatimusmäärittelyssä ja määrittelyssä tehdyt dokumentit vastaajien organisaatioissa käytettäessä vesiputousmallia.

Kuvaus- tyyppi	Vastaaja A	Vastaaja B	Vastaaja C
Prosessit	Kokonaistyön- kulku (prosessi)	Prosessikuvauksia liiketoiminnan näkökulmasta	Toiminnan prosessikuvaus
Vaatimukset		Vaatimusten kuvauksia (toiminnalliset ja ei-toiminnalliset vaatimukset)	Vaatimusluettelo
Toiminta	Toiminnallisuuden kuvaus, rajapinnat, eräajot, konversio ja siirtymävaihe	Käyttötapauskuksia: interaktiivisia ja ei-interaktiivisia (SOA, eräajot)	Käyttötapauskaavio ja Käyttötapauskuvaukset, Tietovirran kuvaukset sähköisistä palveluista
Käyttöliittymät	Näytöt	Käyttöliittymä-dokumentteja	Käyttöliittymämallit, välillä rautalankamalleja (ylläpidossa vähemmän)
Tulosteet	Päätös- ja kirje-mallit, fraasit, raportit	Tulosteiden määrittely	Tulostemallit
Käyttäjäroolit	Käyttäjäroolit	Käyttäjäroolien kuvaukset	
Käsitteet ja tiedot	Tiedot	Luokkamalli, tietokannan kuvaukset	
Muut	Järjestelmän yleiskuvaus		
	Säädösperusta, jossa kuvataan miten lainsäädäntöä toimeenpannaan järjestelmässä		
		Arkkitehtuurikuvauksia	
		Teknisiä suunnitelmia	
			Käyttäjäohjeet (aloitettu)
			Hyväksymistestitapaukset (aloitettu)

Kaikkien vastaajien organisaatioissa kuvattiin vaatimusmäärittelyssä ja määrittelyssä hyvin samantapaisia asioita käytettäessä vesiputousmallia. Vastaajien mainitsemat dokumentit voidaan jakaa seuraaviin kokonaisuuksiin: Prosessit, vaatimukset, toiminta, käyttöliittymät, tulosteet, käyttäjäroolit, käsitteet ja tiedot sekä muut (taulukko 1). Pienet erot tehdyissä dokumenteissa johtunevat siitä, että eri organisaatioissa käytetään erilaisia dokumentointitapoja ja mallipohjia. Kuvattavat asiat voidaan jakaa eri dokumentteihin eri tavoin. On myös mahdollista, että haastattelutilanteessa vastaaja voi unohtaa mainita jonkun yksittäisen dokumentin, ei kerro tarkasti mitä kaikkea dokumentit sisältävät tai kokee joidenkin dokumenttien kuuluvat enemmän esiselvitys- tai suunnitteluvaiheisiin.

Vesiputousmallissa tehdyn dokumentaation hyvinä puolina pidettiin sen tarkkuutta sekä vuosien varrella vakiintunutta ja tuttua kuvaustapaa. Yksi vastaaja piti erityisesti siitä, että toimialalle tärkeä lainsäädännöllinen näkökulma tuli dokumentaatioissa hyvin esille. Kaksi vastaajista mainitsi erikseen, että menetelmä ei ole vaikuttanut siihen mitä dokumentteja on tehty. Toinen heistä korosti organisaation vaativan, että samat asiat kuvataan menetelmästä riippumatta.

Yksi vastaajista piti vesiputousmallissa tehdyn dokumentaation huonona puolena sitä, että dokumentaatiota ei aina tuoteta riittävän oikea-aikaisesti. Esimerkiksi teknisempää dokumentaatiota tehdään liian varhaisessa vaiheessa eikä sitä tule ylläpidettyä. Toinen vastaajista harmitteli, ettei dokumentaatio ohjannut tarkkuustasoa, tarkkuus vaihteli kirjoittajan mukaan ja välillä määrittelyt jäivät hyvin ylätasolle. Kolmas vastaaja näki huonoina puolina, että tarkalla tasolla kuvaaminen vie paljon aikaa ja kerralla määrittelyä liian suuria kokonaisuuksia. Laadunvarmistus ja iterointi ovat hankalia ja kuluttavat resursseja. Usein määrittely kestää niin pitkään, esimerkiksi vuoden, että sinä aikana tapahtuu paljon muutoksia. Kokonaisuuksia on yritetty pilkkoa pienemmiksi, mutta se on vaikeaa, koska sovellusalueet ovat niin suuria. Yhdessä organisaatioista dokumentointia hankaloittivat myös vanhentuneet, erityisesti visuaalista ilmaisua rajoittaneet, välineet.

4.3 Dokumentointi ja asiakaskokemus RUP:ssa

Ainoastaan vastaajalla A oli laajaa kokemusta RUP:sta. Vastaajan C organisaatiossa vesiputousmalliin oli sovellettu RUP:n piirteitä. Tutkimuksen tulokset RUP:n osalta perustuvat vastaajan A ja vastaajan C haastatteluihin.

Molemmissa organisaatioissa henkilöstöä on valmennettu RUP:n käyttöön. Ensimmäisessä organisaatiossa menetelmää on jalkautettu järjestämällä henkilöstölle infoja ja koulutusputkia. Koulutuksia on ostettu talon ulkoa ja pidetty itse. Toisessa organisaatiossa pidettiin vajaa kymmenen vuotta sitten koulutuskierroksia ja sen jälkeen koulutus on tarvittaessa hoidettu parityöskentelynä. Organisaatiossa on aika pieni vaihtuvuus, joten suurempaa koulutustarvetta ei ole ilmennyt.

Ensimmäisen vastaajan organisaatiossa tehtiin vaatimusmäärittelyssä ja määrittelyssä useita erilaisia lopputuotteita RUP:a käytettäessä. Tehtyjä lopputuotteita ovat esimerkiksi vaatimusluettelo, järjestelmäarkkitehtuurikuvaus, järjestelmän yleiskuvaus, käyttötapauskuvaus, käyttöliittymäkuvaus, SOA-palvelukuvaus, toiminnallisen prosessin kuvaus ja käsitelmä. Vesiputousmalliin verrattuna on syntynyt enemmän erillisiä lopputuotteita, asiat on jaoteltu niihin eri tavalla ja lisäksi on tehty myös käytettävyyteen liittyviä kuvauksia. Käytettävyyssasioita lukuun ottamatta aivan samat asiat löytyvät eri muodossa myös vesiputousmallin mukaisesta dokumentaatiosta. Toisen vastaajan organisaatiossa RUP:sta on poimittu sopivia dokumenttimalleja kuten esimerkiksi käyttötapauskaavio ja käsitelmäkaavio. Määrittelyvaiheessa on tehty ylemmän tason käsitelmärittelyä ja toimittajan puolella lisäksi teknisessä määrittelyssä käsitelmalleja. Dokumentteja on yritetty tehdä iteratiivisesti ja toimittaja on täydentänyt organisaation liiketoimintapuolella tehdyt dokumentit. Haasteena on kuitenkin ollut missä menee teknisen määrittelyn ja suunnittelun raja.

RUP:n mukaisen dokumentaation hyvinä puolina vastaajat pitivät strukturoitua rakennetta ja kokonaisuuden kuvaamista. Erityyppiset asiat kuvataan sellaisissa dokumenteissa, jotka tukevat juuri niiden asioiden kuvaamista. Määrittelyssä pystytään visualisoimaan, esimerkiksi käyttötapauskaavioiden avulla, mistä kokonaisuudesta puhutaan ja miten asiat ja roolit liittyvät toisiinsa. On myös hyvä, että käytetään samanlaista kuvaus-

tapaa ja tehdään samanlaisia dokumentteja. Vastaajat ovat kokeneet dokumentaation myös vastanneen liiketoiminnan vaatimuksia.

Huonoina puolina RUP:n mukaisessa dokumentaatioissa vastaajat näkivät erityisesti dokumenttien suuren määrän ja oikean kuvaustason löytämisen vaikeuden. Lukumäärällisesti voi syntyä paljon dokumentteja, mutta siihen vaikuttaa myös se, miten toiminnallisuus jaetaan. Myös kokonaiskuvan saaminen voi olla välillä vaikeaa. Sekä liiketoiminta että IT-taustan omaaville määrittäjöille oikean tason löytäminen on välillä vaikeaa. Toiset asiat on myös helpompi dokumentoida kuin toiset. Määrittelydokumentaation suuri määrä tekee myös sen katselmoinnista haasteellista ja aikaa vievää.

Yhtenä määrittelyn haasteena vastaajat näkivät puutteet osaamisessa. Määrittelijöiden erilaiset taustat vaikuttavat myös hyvin paljon määrittelyyn ja sen onnistumiseen. Liiketoimintataustaiset määrittelijät ja asiantuntijat tarvitsisivat enemmän koulutusta kuvausten, esimerkiksi käyttötapauskaavioiden, ymmärtämiseen ja tekemiseen. Vesiputousmallin mukaiseen määrittelyyn verrattuna RUP:ssa pitäisi ymmärtää järjestelmän toimintalogiikkaa paremmin. Erilaisten välineiden käyttö lisää määrittelyn riskejä ja toinen vastaajista korostaa, että määrittelyä ei kannata sitoa liikaa eri välineisiin eikä määrittelyvaiheessa kannata käyttää liian monia eri välineitä. Välillä hankaluuksia saattaa aiheuttaa myös se, että ihmiset ovat liian pitkään samalla alueella ja ovat hyvin kiinni vanhoissa järjestelmissä.

Haastavaa on myös saada pidettyä eri vaiheiden, esimerkiksi vaatimusmäärittelyn ja suunnittelun, dokumentaatio samalla tasolla. Toinen vastaaja kertoi, että toimittaja pitää suunnittelun dokumentaatiota heidän vastuullaan ja usein siihen liittyvät ongelmat tulevat ilmi vasta hyväksymistestissä. Vaikka organisaatio saa toimittajalta nähtäväkseen suunnittelun dokumentaation, ei organisaatiolla ole resursseja ja osaamista sen tarkasteluun.

Vastaajat eivät oikein osanneet sanoa, onko kommunikointi RUP:a käytettäessä ollut erilaista verrattuna vesiputousmalliin. Ensimmäisen vastaajan mielestä ryhmätyötä tehdään paljon. Toinen vastaaja koki, että toteutuksen kanssa kommunikointi on parantunut, mutta toisaalta siinä oli etunsa, että vesiputousmallissa tehtiin eri vaiheet selkeäm-

min omina kokonaisuuksinaan. RUP:ssa jo projektin alkuvaiheessakin saatetaan tehdä toteutusta ja puhua teknisemmistä asioista. Se hämmentää joitakin ihmisiä ja ihmiset saattavat joutua tekemisiin aivan uudenlaisten termien ja asioiden kanssa. Esimerkiksi liiketoiminnan asiantuntijat ja lakimiehet ovat joutuneet kuulemaan komponenteista tms. teknisemmistä termeistä.

4.4 Dokumentointi ja asiakaskokemus Scrumissa

Kaikilla vastaajilla oli jonkin verran kokemuksia ja näkemyksiä ketterien piirteiden soveltamiseen projekteissa, vaikka Scrum-kokemukset jäivät vähäisemmiksi. Kokemusten määrään vaikuttaa myös se, että asiakas ei aina toimi menetelmän mukaisesti tai toimitajan käyttämä menetelmä ei edes näy asiakkaan puolelle. Vastaajilla A ja B oli kokemusta projektista, jossa oli käytetty menetelmänä Scrumia. Näistä vastaajan B kokemus oli pidempi ja laajempi. Vastaajan C organisaatiossa vesiputousmalliin oli sovellettu Scrumin piirteitä. Tutkimuksen tulokset Scrumin osalta perustuvat kaikkien vastaajien haastatteluihin.

Yksi vastaajista oli osallistunut projektiin, jossa toimittaja käytti Scrumia. Projektissa tehtiin tukijärjestelmä, jossa oli ohut liiketoimintalogiikka. Määrittelyt kuuluivat organisaation vastuulle ja toteutus toimittajalle. Vastaaja ei ollut mukana projektin alusta asti, vaan meni mukaan projektiin kesken kaiken, kun projektissa tarvittiin apuvoimia ongelmien realisoituessa. Organisaatio ei ollut tehnyt kirjallisia määrittelyjä, koska toimittaja oli vakuuttanut, että niitä ei tarvita. Määrittelyinä toimivat tarjouspyynnön liitteet ja toimittajan ylläpitämä product backlog. Lisäksi toimittajan kanssa pidettiin palavereja, joissa myös vaatimuksia ja toiminnallisuutta käytiin läpi. Projektin edetessä kuitenkin huomattiin, etteivät tarjouspyynnön liitteet ja product backlog riitä. Product backlog oli liian ylätasolla eikä palavereissa toimittajan kanssa käsitellyt asioita kirjattu ylös. Organisaation edustajat olivat useamman kerran jopa kysyneet toimittajalta pitäisikö asiat kirjata ylös, mutta toimittaja oli joka kerran vakuuttanut, että product backlog riittää. Määrittelyjä ei projektin alussa tehty, koska luotettiin toimittajan sanaan. Organisaatio alkoi tehdä määrittelyä vasta sen jälkeen, kun projektissa tuli ongelmia. Tällöin alettiin dokumentoida jo toteutettua ja sitä miten olisi pitänyt toteuttaa. Tehdyt määrittelyt käytiin toimittajan kanssa läpi ja toimittaja esitti niihin liittyen kysymyksiä. Määrittelyjen

tekeminen auttoi myös toimittajaa ja testausta. Vastaaja ei tiedä oliko henkilöstöä koulutettu Scrumin käyttöön, koska ei ollut itse alusta asti mukana projektissa.

Toinen vastaajista osallistui pitkään, yli kaksi vuotta kestäneeseen, Scrum-projektiin, jossa tehtiin uuden räätälöidyn järjestelmän määrittely ja toteutus. Projektin pohjana oli organisaatiossa kilpailutusta varten tehty esiselvitys ja vaatimusmäärittely. Vastaaja oli ollut työssä mukana jo esiselvityksen kommentointivaiheesta alkaen. Esiselvitys ja vaatimusmäärittely tehtiin organisaation omin voimin, mutta mukana oli ollut myös konsultti tekemässä kirjoitustyötä. Scrumin käyttöä määrittely- ja toteutusprojektissa ehdotti kilpailutuksessa valittu toimittaja. Toimittaja myös vaati, että projektiin nimetään asiakkaan puolelta product owner. Product ownerin tehtävään nimettiin vastaaja. Vastaaja itse suhtautui Scrumiin alusta alkaen myönteisesti ja oli menetelmästä innostunut. Organisaatiossa kukaan ei ollut aikaisemmin Scrumia käyttänyt. Vastaaja ei osallistunut projektin sprinttien suunnitteluun, vaikka niihin olisi ollut hyvä osallistua. Sprintin demossa olivat asiakkaan puolelta mukana product ownerin lisäksi projektiin nimetyt organisaation liiketoiminnan asiantuntijat. Vastaaja osallistui product backlogin tekemiseen projektin alussa asiakkaan vaatimusten pohjalta sekä projektin kuluessa lisäsi uusia töitä ja pudotti töitä pois. Vastaaja myös priorisoi product backlogin itemeja yhdessä toimittajan edustajan kanssa, ja koki yhteistyön olleen hyvä ratkaisu ja molempien saaneen hyvää palautetta toisiltaan. Vastaajan mielestä projekti lähti liikkeelle hyvin. Sekä product ownerille että liiketoiminnan asiantuntijoille oli palkitsevaa nähdä, että valmista syntyi. Jossain vaiheessa vastaaja kuitenkin havahtui, että sovituisia demoista oli alettu luisua ja asiakas oli jätetty ulkopuolelle. Asiakas olisi halunnut tietää mitä on tehty, mutta toimittaja ei olisi halunnut näyttää, koska uusia itemeja ei ollut valmiina. Toimittajan puolella ei haluttu demota sellaista, mikä ei ollut vielä valmis kokonaisuus. Kun vastaaja yritti kysellä asiasta, toimittajan puolelta vastattiin, että he eivät tee enää Scrumin, vaan Kanbanin mukaan. Vastaaja tulkitsi sen niin, että asiakkaan ei haluttu olevan niin tiiviisti mukana kuin alkuvaiheessa. Tilanteeseen vaikutti erityisesti kova aikataulupaine. Vastaaja harmittelee, että Scrumin alkuperäinen ajatus menetetään, kun tulee hätä ja paniikki. Se leimaa Scrumia, vaikka ongelma ei ollut Scrumissa. Ei riitä, että ainoastaan sanotaan, että nyt tehdään Scrumilla. Organisaation henkilöstöä ei ole koulutettu tai valmennettu Scrumin käyttöön. Myös toimittajan puolella toimintatapa tuntui olevan monelle uusi. Vastaaja ei tiedä miten toimittajan väkeä oli koulutettu Scrumiin, mutta

toimittaja oli projektia käynnistettäessä kertonut asiakkaalle odotuksistaan Scrumin suhteen. Vastaaja itse oli hankkinut Scrumista tietoa mm. kavereiltaan. Vastaaja ajattelee jälkikäteen, että oman organisaation porukka olisi pitänyt kouluttaa sisäisesti. Ketterämpi määrittely olisi edellyttänyt, että asiakkaan puolella olisi pitänyt toimia toisin ja olla enemmän valmiuksia ketterään määrittelyyn. Ongelmana oli myös, että liiketoiminnan asiantuntijat olivat liian kiinni vanhassa tavassa, eivätkä ymmärtäneet olevansa Scrum-projektissa. He tekivät omaa työtään aivan kuten vesiputousmallissa.

Kolmannen vastaajan organisaatiossa on käytetty vesiputousmallia, jossa projektikohtaisesti on voinut olla ketteriä tai Scrumin piirteitä. Ketteryyttä on kokeiltu pienemmissä projekteissa esimerkiksi sähköisen asioinnin puolella. Ketteryyttä on haettu myös yhteistyössä tuttujen toimittajien kanssa. Organisaatiossa ketteryys on tarkoittanut lähinnä sitä, että on tehty pienempiä kokonaisuuksia nopeammassa aikataulussa eli iso vesiputous on pilkottu pienempiin osiin. Vastaaja kokee, että ketteryyden ottaminen käyttöön ei kuitenkaan ole aina niin helppoa. Organisaatiossa on mietitty, että voitaisiin tehdä ketterämmin, mutta ei ole oikein ehditty kokeilla, eikä ole ollut yhteistä näkemystä asiaan. Organisaation linja on aika lailla se, että vanhojen järjestelmien ylläpidossa toimitaan saman kaavan mukaan kuten ennenkin, mutta uusia järjestelmiä tai palveluja tehtäessä saatetaan kokeilla ketterämpää. Mitenkään systemaattista kokeilu ei kuitenkaan ole. Organisaatiossa henkilöstöä ei ole mitenkään erityisesti valmennettu ketteryyteen. Kokeiluista on kerrottu ja kokeiluja on tehty myös henkilöstön tai toimittajan edustajien ehdotuksien perusteella.

Vastaajien mukaan vaatimusmäärittelyssä ja määrittelyssä tehty dokumentaatio ei merkittävästi eroa vesiputousmallin tai RUP:n mukaan tehdyistä dokumentaatiosta. Yhden vastaajan organisaatiossa Scrum-projektin alussa ei tehty ensin dokumentaatiota ollenkaan tarjouspyynnön liitteiden ja product backlogin lisäksi. Kun dokumentaatiota alettiin ongelmien ilmaannuttua tehdä, tehtiin käyttötapauksia, käyttöliittymäkuvauksia ja prosessikuvauksia. Niiden tekemisessä käytettiin eräänlaista sovellettua RUP:a, jota oli käytetty aikaisemmin toisessa projektissa ja joka oli tuttu tapa dokumentoida. Toisen vastaajan Scrum-projektissa tehtiin määrittelyssä käyttöliittymädokumentteja, käyttötapauksia ja tulosteiden määrittelyjä. Vaatimusmäärittely oli tehty ja dokumentoitu jo ennen projektia, eikä sen tekemisessä sovellettu Scrumia.

Ensimmäinen vastaaja näkee, että dokumentaatiossa oli hyvää sen tarkkuus ja se, että dokumentaatio helpotti testausta ja toimittajaa. Toinen vastaaja kokee dokumentaation olleen valmiimpaa, koska toteutusta tehtiin samaan aikaan. Käyttöliittymän toteuttaminen antaa ajatusten tueksi nopeasti jotakin konkreettista. Voidaan esimerkiksi hyvissä ajoin huomata, että ei tuon painikkeen olekaan hyvä noin toimia. Vesiputousmallissa muutostarpeet tulevat esille vasta paljon myöhemmin. Kolmas vastaaja näkee hyvänä sen, ettei tehdä aivan niin yksityiskohtaista dokumentointia kuin esimerkiksi vesiputousmallin mukaan on usein tehty. Voidaan esimerkiksi protoilla tai vaikka pilotoida esimerkiksi sähköisen asioinnin palvelua. Esimerkiksi koko kansalle tarkoitettu palvelu julkaistaan ensin pienemmälle käyttäjäryhmälle, esimerkiksi opiskelijoille. Tämä on enemmän tuotantoympäristön testausta, mutta jos käytössä havaitaan kehitystarpeita tai virheitä, niitäkin voidaan usein korjata ennen laajaa tuotantokäyttöä. Myös asiakkailta on pyydetty palautetta palveluista. Joidenkin palveluiden osalta palautetta voi antaa jatkuvasti.

Vastaajien mukaan Scrum-projektissa tehty dokumentaatio vastaa liiketoiminnan vaatimuksia samalla tavalla kuin muitakin menetelmiä käyttäen tehty dokumentaatio. Yksi vastaajista toteaa, että ainakaan Scrumista johtuen ei ole syytä epäillä, ettei dokumentaatio vastaisi liiketoiminnan vaatimuksia.

Yksi vastaajista pitää omassa Scrum-kokemuksessaan erikoisena sitä, että vaikka käytetään Scrumia, käyttötapausmäärittelyt tehdään kuitenkin hyvin pitkälle valmiiksi ennen kuin ne annetaan toteutukselle. Vaikka toisaalta on hyvä syventyä yhteen asiaan kerrallaan, se on myös määrittelyn huono puoli, koska samoihin asioihin palataan aina uudelleen ja uudelleen. Voi käydä niin, että on lähdetty liikkeelle kohdasta A ja kohdassa S huomataan, että A ei voikaan olla tuollainen. Silti A:n eteen on tehty paljon määrittely- ja toteutustyötä. Vastaajan mielestä Scrumissa on haastavaa myös oman organisaation liiketoiminnan asiantuntijoiden työn ohjaaminen ja opettaminen pois vanhasta. Liiketoiminnan asiantuntijat ovat tottuneet vaatimaan pikkutarkkuutta, vaikka kirjoitusvirheet olisi voinut hyvin korjata vasta lopuksi. Toimittajan ja product ownerin olisi pitänyt enemmän ohjata oikeaan suuntaan ja vetää tilanne poikki tarvittaessa toteamalla ”tähän palataan myöhemmin”. Haasteena pitkässä Scrum-projektissa oli myös, että

kokonaisuus ei muodostunut, vaikka oli kokonaisuutta kuvaavia dokumenttejakin. Oli rankempaa tehdä koko ajan vähän verrattuna siihen, että olisi kovemmalla tahdilla puristettu työ valmiiksi. Uupumus iski puolin ja toisin.

Vastaajat eivät näe kommunikointia Scrum-projektissa merkittävästi erilaisena verrattuna vesiputousmalliin tai RUP:iin. Yhden vastaajan mukaan eri ihmisten ja eri toimittajien kanssa on aina erilaista tehdä työtä. Toinen vastaaja kokee, että kommunikointi toimii paremmin, koska tulee enemmän keskustelua ja palautetta. Heidän organisaatiossaan on joissakin projekteissa kokeiltu hyvällä menestyksellä kommunikointia lisääviä aamupalavereja, joissa on käyty tehtäviä ja edistymistä estäviä asioita läpi.

Jokainen vastaajista kokee, että määrittelyä voi tehdä ketterästi. Yhden vastaajan mielestä ketteryys sopii esimerkiksi uuden toiminnallisuuden määrittelyyn, jossa protoilu helpottaa ajatusten synnyttämistä. Silloinkin täytyy kuitenkin dokumentoida, kirjata ylös ja miettiä mitä halutaan tehdä. Ei kuitenkaan ole tarpeen jatkuvasti iteroida ja tarkentaa. Kun haluttu asia on kasassa, voidaan kuvaukselle lyödä valmiin leima. Valmiin määrittelmä oli sovittu ainoastaan yhden vastaajan Scrum-projektissa, mutta siinäkin sitä ei ollut kirjattu ylös. Toinen vastaaja toteaa omasta Scrum-kokemuksestaan huolimatta, ettei ole itse ollut sellaisessa projektissa missä määrittelyä olisi tehty ketterästi. Vastaaja näkee tärkeänä, että tehtäessä määrittelyä ketterästi pitää hyväksyä iterointi ja puhua alusta alkaen samaa kieltä, ettei takerruta käsitteisiin. On tärkeää ensin kuvata yltälasolla mitä tarvitaan ja sitten vasta tarkentaa. Esimerkiksi käyttötapaukset voi ensin kuvata vain karkealla tasolla ja tarkentaa niitä myöhemmin. Kolmas vastaaja ajattelee, että mitä pienempi juttu, sitä enemmän määrittelyä voi tehdä ketterästi. Vastaaja korostaa, että kurin pitäminen ja tavoitteessa pysyminen on hyvin tärkeää. Täytyy katsoa, ettei työmäärä leviä käsistä, jos kehitetään kaikkea kivaa. Hienoja ideoita, joita esimerkiksi toteutuksessa syntyy, ei voi saman tien ryhtyä toteuttamaan, vaan ne täytyy vain kirjata ylös. Ketterää määrittelyä tehtäessä tekijät pitäisi olla myös samassa tiimissä, samassa tilassa ja vain yhdessä projektissa samaan aikaan.

4.5 Ketterien menetelmien soveltuvuus vastaajan organisaatioon

Tutkimuksen tulokset ketterien menetelmien soveltuvuudesta vastaajan organisaatioon perustuvat kaikkien vastaajien haastatteluihin. Vastaajien mielipiteet olivat hyvin samantyyppisiä, ja yhteenvetona voisi todeta, että vastaajien mielestä ketteriä menetelmiä kannattaa käyttää, mutta kannattaa tarkkaan harkita, minkälaisissa projekteissa niitä käytetään.

Jokainen vastaaja on tietyin varauksin sitä mieltä, että ketterät menetelmät soveltuvat heidän organisaatioihinsa. Kukaan heistä ei kuitenkaan käyttäisi ketteriä menetelmiä kaikissa projekteissa. Yksi vastaajista korosti, että täytyy valita kuinka ketterästi tehdään ja missä projektissa. Vastaajan mielestä ketterät menetelmät soveltuvat parhaiten pieniin projekteihin ja sellaisiin, joissa kypsyytaso on riittävä. Vastaaja kokeilisi ketteriä menetelmiä sähköisessä asiointissa, jossa tehdään yksittäisiä palveluja ja joissa innovointi ja uuden kokeilu on tärkeää. Vastaaja ei käyttäisi ketteriä menetelmiä ensimmäisenä suurina ydinliiketoimintaan liittyviä räätälöityjä järjestelmiä tehtäessä, vaikka niissäkin pitäisi saada aikaisemmassa vaiheessa tuloksia ja kommentoitavaa. Toisen vastaajan mielestä ketterät menetelmät soveltuvat ainakin pienempiin projekteihin. Vastaaja ajattelee, että ketteriä menetelmiä ei ole järkevää käyttää, jos aidosti tiedetään mitä pitää tehdä. Jos esimerkiksi käytettävissä on tietty rahasumma ja tietty aika, eivät ketterät menetelmät anna lisäarvoa ja saattavat olla toimittajallekin riski. Vastaaja käyttäisi ketteriä menetelmiä silloin, kun on enemmän vapautta hakea eri vaihtoehtoja. Kolmannen vastaajan mielestä ketteryyttä voisi hyödyntää nykyistä enemmänkin esimerkiksi sähköisissä palveluissa ym. pienissä projekteissa. Suurissa projekteissa vastaaja ei kuitenkaan käyttäisi ketteriä menetelmiä.

Käytettäviä määrittely- ja dokumentointitapoja kannattaa vastaajien mukaan miettiä erikseen siirryttäessä käyttämään ketteriä menetelmiä. Yhden vastaajan mukaan ketteryys ei tarkoita, että mitään ei dokumentoitaisi. Toinen vastaaja korostaa, että määrittelyä ja dokumentaatiota tehtäessä on uskallettava pysyä riittävän ylätasolla ja ottaa mukaan kaikenlaisia ihmisiä. Kolmas vastaaja analysoi, että nykyinen kankea katselmointimenettely ei toimi ketterissä menetelmissä ja toteaa, että ei ole kauhean ketterää odottaa havaintoja pitkään. Hänen mukaansa ketterässä projektissa voisi ehkä enemmän hyväk-

syäkin toiminnallisuutta. Vastaajan mielestä riippuu projektin ja järjestelmän kokoluokasta mitä dokumentaatiota tarvitaan. Lopullista tuotosta kuvaavia dokumentteja ei ehkä tarvittaisi, jos lopullinen tuotos – kuten esimerkiksi käyttöliittymä tai tuloste - syntyisi varhaisessa vaiheessa. Silloin esimerkiksi käyttöliittymästä voisi ottaa kuvaruutukaappaukset. Vastaajan mielestä on vaikea kuvitella, ettei lakitekstiin pohjautuvia käsittelysääntöjä tarvitsisi kuvata, mutta samaan hengenvetoon alkaa ideoida pystyisikö niitä kuvaamaan ainoastaan käyttöohjeissa. Vastaajan mukaan pitäisi kaiken kaikkiaan miettiä mitä voitaisiin tehdä suurelle määrälle määrittelydokumentaatiota. Hänen mukaansa on älytöntä, että tuotetaan todella paljon dokumentaatiota, jonka kokoonpano ei ole hallittua ja joka ei pysy ajan tasalla.

5 Yhteenveto ja johtopäätökset

Opinnäytetyön tavoitteena oli selvittää, mitä etuja ja haasteita määrittelyssä ketterillä menetelmillä on verrattuna vesiputousmalliin. Lisäksi opinnäytetyössä oli tarkoitus analysoida, miten valittu menetelmä vaikuttaa määrittelyyn ja sen dokumentointiin ja vastaako ketterässä projektissa tehty dokumentaatio liiketoiminnan vaatimuksia.

Tutkimuksen tuloksissa korostui näkemys, että ketteriä menetelmiä kannattaa käyttää, mutta täytyy tarkkaan harkita, missä projekteissa ja kuinka ketterästi. Vastaajat olivat hyvin yksimielisiä siitä, että ketterät menetelmät sopivat parhaiten pieniin projekteihin. Ketteriä menetelmiä kannattaisi heidän mielestään kokeilla esimerkiksi sähköisessä asiointissa, jossa tehdään yksittäisiä palveluita ja innovointi ja uuden kokeilu on tärkeää.

Ketterillä menetelmillä on vesiputousmalliin verrattuna sekä etuja että haasteita. Käytetäessä RUP:ia dokumentaation strukturoitu rakenne, visualisointi ja kokonaisuuden kuvaaminen tukevat paremmin määrittelyä. Ryhmätyötä tehdään paljon ja kommunikointi toteutuksen kanssa on parantunut. Toisaalta dokumenttien suuri määrä ja oikean kuvaustason löytäminen aiheuttavat haasteita. Scrumin osalta etuina nousi esille jo Agile Manifestostakin tuttuja asioita: Käyttöliittymän toteuttaminen antaa ajatusten tueksi jotain konkreettista, on palkitsevaa nähdä kun valmista syntyy koko ajan, ei tarvitse dokumentoida niin yksityiskohtaisesti ja voidaan protoilla. Ketterän dokumentoinnin tekeminen on kuitenkin erittäin vaikeaa, vaikka oltaisiin Scrum-projektissa. Olisi hyvä ensin kuvata ylätasolla, mitä tarvitaan, ja sitten iteroiden tarkentaa. Käytännössä helpos- ti käy kuitenkin niin, että käyttötapaukset kirjoitetaan valmiiksi ennen toteutukselle antamista ja liiketoiminnan asiantuntijat tekevät määrittelyä yhtä pikkutarkasti kuin vesiputousmallissakin.

Sekä RUP:n että Scrumin osalta haittapuolina korostuivat erityisesti soveltamisen vaikeus ja osaamisen puute. Menetelmää ei aina osata soveltaa organisaatiolle sopivalla tavalla, ei ole yhdessä sovittu miten tehdään, eikä henkilöstöä ole valmennettu riittävästi uusien toimintatapojen käyttöön. Ketterämpi määrittely edellyttää uudennlaisia toimintatapoja myös asiakkaan puolella, eikä vanhasta pois oppiminen ole kenellekään helppoa. Muutos ei tapahdu hetkessä.

Vesiputousmalli ja RUP painottavat dokumentoimista enemmän kuin Scrum, ja niiden mukaisissa projekteissa yleensä syntyy enemmän dokumentaatiota kuin Scrum-projektissa, mutta sinänsä menetelmä ei vaikuta määrittelyyn ja dokumentointiin. Tutkimuksessa vastaajat olivat yksimielisiä siitä, että ketteriä menetelmiä käytettäessä vaatimusmäärittelyssä ja määrittelyssä tehty dokumentaatio ei merkittävästi eroa vesiputousmallin tai RUP:n mukaan tehdystä dokumentaatiosta, eikä menetelmä vaikuta dokumentointiin. Tutkimuksen tuloksissa oli nähtävissä myös, että määrittelyssä ja dokumentoinnissa toistuivat samat ongelmat menetelmästä riippumatta.

Tietoperustaa kirjoittaessani en löytänyt lähdeaineistosta vastausta kysymykseen, vastaako ketterässä projektissa tehty dokumentaatio liiketoiminnan vaatimuksia. Tutkimuksen mukaan Scrum-projektissa tehty dokumentaatio kuitenkin vastaa liiketoiminnan vaatimuksia samalla tavalla kuin muitakin menetelmiä käyttäen tehty dokumentaatio. Vaatimusmäärittelyä ja toiminnallista määrittelyä tekevät ja lukevat hyvin monenlaiset ihmiset, ja dokumentointitapa pitäisi olla helppo ja kaikkien eri osapuolten nopeasti ymmärrettävissä. Jos dokumentaatio ei vastaa liiketoiminnan vaatimuksia, syy on tässäkin tilanteessa itse dokumentaatiossa, ei menetelmässä.

Opinnäytetyössä tarkasteltiin räätälöityjen järjestelmien määrittelyä ja dokumentointia asiakkaan näkökulmasta. Tutkimuksen kohderyhmä valittiin julkiselta sektorilta ja vakuutusosalta, koska näiden toimialojen järjestelmissä määrittelyllä ja dokumentoinnilla on perinteisesti ollut tärkeä rooli, jotta ohjelmat toimivat täysin lain kirjaimen ja toimialan ohjeistuksen mukaan ja ohjelmien ylläpito on helppoa. Tutkimustuloksissa näkyi selkeästi, että valituilla toimialoilla tärkeintä on muistaa, miksi järjestelmiä tehdään ja mikä on organisaation perustehtävä. Vastaajien organisaatioissa ei voida lähteä villisti kokeilemaan kaikkea uutta, mutta silti aikaansa kannattaa seurata. Uusista suuntauksista voi poimia myös toimivia piirteitä, vaikkei kokonaista menetelmää ottaisikaan käyttöön. Henkilöstön kouluttamiseen kannattaa panostaa, ja heille on hyvä antaa aikaa ja sopivia työkaluja uuden omaksumiseen. Projektista ei tee ketterää se, että sen nimen eteen lisätään sana ”Scrum”. Jokaisen ketterän projektin alussa on tärkeää sopia ja kirjata ylös yhteiset pelisäännöt, koska ihmisillä on erilaisia käsityksiä ketteryydestä ja eri menetel-

mistä, eikä toisessa organisaatiossa sovellettu tapa välttämättä sovi sellaisenaan toiseen organisaatioon.

Vaatimusmäärittelyssä ja toiminnallisessa määrittelyssä täytyy tietyt asiat aina käydä läpi, viestittää muille osapuolille ja tarvittaessa myös dokumentoida – menetelmästä riippumatta. Jos dokumentaatio on puutteellinen tai muuten riittämätön, se ei ole menetelmän vika. Mikään menetelmä ei sinänsä tuo ongelmia tai vie niitä pois.

Mielestäni opinnäytetyön tavoitteet saavutettiin ja jokaiseen kysymykseen saatiin vastaus. Opinnäytetyö ei tuonut esille mitään sensaatiomaista ja ennakkokäsitykseni pitivät suhteellisen hyvin paikkansa. Sain kuitenkin vahvistusta ajatuksilleni, paljon uutta teoriatietoa sekä uusia näkökulmia. Tutkimukseen liittyvät haastattelut menivät hyvin, vastaukset olivat hyviä, eikä niiden analysointi tuottanut ongelmia. Vastaajien lukumäärä on kuitenkin aika pieni, joten tulosten pohjalta ei voi tehdä yleistyksiä tai laajempia johtopäätöksiä. Vastaajat antoivat mielestäni hyvän ja realistisen kuvan siitä, miten tutkimuksen kohderyhmään kuuluvilla toimialoilla suhtaudutaan ketteriin menetelmiin. Uskon, että tutkimuksen tulokset kiinnostavat vastaavilla toimialoilla, sekä asiakkaan että toimittajan puolella, työskenteleviä IT-ammattilaisia, ja toivon, että moni saa niistä hyviä ideoita omaan työhönsä. Toivon, että organisaatioissa kiinnitetään jatkossa enemmän huomioita koulutukseen otettaessa ketteriä menetelmiä käyttöön ja aloitettaessa uusia projekteja. Toivon myös, että jokaisessa projektissa muistettaisiin aina ensin miettiä rauhassa mitä pitää dokumentoida ja ketä varten, ja sitten sopia millä tavalla ja millä tarkkuudella dokumentointi tehdään.

Suosittelen jatkotutkimusaiheeksi tutkimusta ketterästä määrittelystä. Tutkimuksessa voisi tarkastella miten vaatimusmäärittelyä, toiminnallista määrittelyä ja niiden dokumentointia voisi tehdä aidosti ketterämmin toimialoilla, joiden toiminta nojaa lainsäädäntöön tai tarkkaan ohjeistukseen, ja joilla on perinteisesti tehty tarkkaa dokumentaatiota. Tutkimuksessa voisi hakea vinkkejä ja esimerkkejä siitä, miten dokumenttien lukumäärää ja sivumäärää saisi laskettua ilman, että käytetään teknistä tai syvempää tietoteknistä osaamista vaativia välineitä sekä miten dokumentoinnin ongelmakohtia saataisiin ketteryydellä eliminoitua. Ideoita voisi etsiä esimerkiksi havainnollistaviin esimerkkeihin perustuvasta määrittelystä ja hyväksymistestivetoisesta kehittämisestä (Lind

2013, 3–4). Toinen hyvä jatkotutkimusaihe on mielestäni katselmointikäytännöt ketterissä projekteissa. Yksi haastateltavistani ehdotti, että ketterässä projektissa voisi ehkä enemmänkin hyväksyä toiminnallisuutta suurten dokumenttimäärien tarkan katselmoimisen sijaan. Kolmas jatkotutkimusaihe on laaja kyselytutkimus ketterien menetelmien käytöstä Suomessa. Aihe on mielestäni jatkuvasti ajankohtainen, eikä haittaa, vaikka joku toinen olisi jo vastaavasta aiheesta tehnyt tutkimuksen. Erityisen kiinnostavaa olisi selvittää, kuinka paljon toimiala vaikuttaa ketterien menetelmien käyttämiseen. Minun haastateltavani olivat sitä mieltä, että he eivät käyttäisi ketteriä menetelmiä kaikessa ohjelmistokehityksessä, ja heidän mielestään ketterät menetelmät sopivat parhaiten pieniin projekteihin ja esimerkiksi sähköisten palvelujen tekemiseen. Olisi mielenkiintoista kuulla, onko se yleinen näkemys Suomessa tai minun tarkastelemillani toimialoilla, vai oliko puhdasta sattumaa, että kaikki minun haastateltavani olivat asiassa samoilla linjoilla.

Kokonaisuudessaan opinnäytetyön tekeminen oli hyvin mielenkiintoinen ja antoisa prosessi. Alun perin minun piti tehdä opinnäytetyö aivan toisesta aiheesta, mutta aihe ei tuntunutkaan mielekkäältä enää sen jälkeen, kun vaihdoin työpaikkaa puolitoista vuotta sitten. Uuden aiheen kehittäminen ja rajaaminen veivät aikansa, ja jouduin tekemään opinnäytesuunnitelman kokonaan uudestaan. Uusi aiheeni ”Määrittelyn haasteet siirryttäessä vesiputousmallista ketteriin menetelmiin” sopi minulle kuitenkin mainiosti, koska olen koko viisitoista vuotta kestäneen IT-urani ajan työskennellyt pääasiassa määrittelytehtävissä, dokumentointi on lähellä sydäntäni, olen päässyt kokeilemaan ketteriä menetelmiä ja nykyisessä työpaikassani osallistunut menetelmäkehitykseen määrittelyn osalta. Aloitin opinnäytetyön tekemisen tammikuussa 2013 heti uuden suunnitelman hyväksymisen jälkeen ja tavoitteeni oli saada työ valmiiksi ennen kesää. Kokonaisuikatauluni piti, vaikka eri työvaiheisiin varaamani aika ei aivan suunnitelman mukaisena toteutunutkaan. Teoreettisen osan kirjoittaminen vei paljon enemmän aikaa kun olin arvioinut, haastattelukysymykset syntyivät kirjoittamisen rinnalla ja haastatteluiden tekeminen analysointieineen vei vähemmän aikaa kuin olin arvioinut. Lopullinen opinnäytetyö vastaa suunnitelmaa melko hyvin, vaikka rajauksia on tullut lisää ja tutkimusmenetelmäkin on hiukan muuttunut. Jos nyt tekisin opinnäytetyön uudestaan, muotoilisin ja jaottelisin haastattelukysymyksiä toisella tavalla. Kysymysten jäsentäminen menetelmien mukaan antoi ehkä liikaa painoarvoa menetelmille, vaikka ne eivät tutkimuksen mukaan sinänsä

vaikuta dokumentointiin. Mutta sitä minä en tiennyt ennen haastattelujen tekemistä, kuten en montaa muutakaan esille tullutta asiaa. Vaikka opinnäytetyön aihealue oli minulle monelta osin tuttu, sain prosessin aikana paljon uutta tietoa ja uudenlaisia näkökulmia. Opinnäytetyöstä tulee olemaan minulle varmasti hyötyä myös työpaikallani tulevina vuosina, koska uskon, että ketterät menetelmät ja ketteryys ovat tulleet jäädäkseen. Niiden soveltaminen määrittelyyn ja sen dokumentointiin ei ole aivan helppoa, mutta mahdollista.

Lähteet

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J. 2002. Agile software development methods: review and analysis. VTT:n julkaisu 478. Luettavissa: <http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf>. Luettu: 17.3.2013.

Agile Alliance 2013a. Agile Alliance www-sivujen etusivu. Luettavissa: <http://www.agilealliance.org/>. Luettu: 23.2.2013.

Agile Alliance 2013b. The Agile Manifesto. Luettavissa: <http://www.agilealliance.org/the-alliance/the-agile-manifesto/>. Luettu: 23.2.2013.

Agile Alliance 2013c. What is Agile Software Development? Luettavissa: <http://www.agilealliance.org/the-alliance/what-is-agile/>. Luettu: 5.5.2013.

Agile Finland ry. Agile Finland. Luettavissa: <http://confluence.agilefinland.com/display/af/Agile+Finland+ry>. Luettu: 29.4.2013.

Agile Manifesto. 2001. Manifesto for Agile Software Development. Luettavissa: <http://www.agilemanifesto.org/>. Luettu: 23.2.2013.

Haikala, I. & Mikkonen, T. 2011. Ohjelmistotuotannon käytännöt. 12. uudistettu painos. Talentum Media Oy. Helsinki.

Highsmith, J. 2010. Agile project management: Creating innovative products. 2. painos. Upper Saddle River (NJ): Addison-Wesley.

Hirsjärvi, S. & Hurme, H. 2009. Tutkimushaastattelu: Teemahaastattelun teoria ja käytäntö. Gaudeamus. Helsinki.

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2009. Tutki ja kirjoita. 15. uudistettu painos. Tammi. Helsinki.

IBM 2007. Rational Unified Process data sheet. Luettavissa:
ftp://public.dhe.ibm.com/software/rational/web/datasheets/RUP_DS.pdf. Luettu:
22.2.2013.

Kolehmainen, A. 24.1.2013. Leikkaukset ajavat it-projekteja ketteriin menetelmiin. Luettavissa:
<http://www.tietoviikko.fi/kehittaja/leikkaukset+ajavat+itprojekteja+ketteriin+menetelmiin/a873126?s=u&wtm=tivi-25012013>. Luettu: 29.1.2013.

Kroll, P. & Kruchten, P. 2003. The rational unified process made easy: a practitioner's guide to the RUP. Addison-Wesley. Boston.

Kruchten, P. 2004. The Rational Unified Process: An introduction. 3. painos. Pearson Education. Boston (MA).

Lekman, L. 2011. Lisää onnistumisia Scrumilla. Projektitoiminta, 1/2011, s. 33–35.

Lekman, L. 2012. Julkishallinto siirtyy Agileen. Luettavissa:
<http://larelekman.com/2012/12/02/julkishallinto-siirtyy-agileen/>. Luettu: 22.2.2013.

Lekman, L. 2013. Unelma ketterästä Apotista. Luettavissa:
<http://larelekman.com/2013/02/12/unelma-ketterasta-apotista/>. Luettu: 22.2.2013.

Lind, J. 2013. Acceptance Test Driven Development With VS 2012. Luettavissa:
<http://www.slideshare.net/Tieturi/tech-days-2013-juhani-lind-acceptance-test-driven-development-with-vs2012-v10>. Luettu: 9.5.2013.

McConnell, S. 2002. Ohjelmistotuotannon hallinta. IT Press. Helsinki.

Ojanperä, V. 2012. Ketterä ohjelmistokehitys säästää rahaa ja aikaa. Tietokone 8.6.2012. Luettavissa:
http://www.tietokone.fi/uutiset/kettera_ohjelmistokehitys_saastaa_rahaa_ja_aikaa. Luettu: 3.2.2013.

Pajuoja, P., Huhtala, E., Pursiainen, M. & Knuutila, M. 2011. Tekes kehittää ketterästi. Projektitoiminta, 1/2011, s. 18–21.

Pohjonen, R. 2002. Tietojärjestelmien kehittäminen. Docendo Finland Oy. Jyväskylä.

Paananen, J. 2005. Tietotekniikan peruskirja. Docendo Finland Oy. Jyväskylä.

Rodriguez, P., Markkula, J., Oivo, M. & Turula, K. 2012. Survey on Agile and Lean Usage in Finnish Software Industry. University of Oulu.

Schwaber, K. & Sutherland, J. 2011. The Scrum Guide. Luettavissa:

<http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20FI.pdf>. Luettu: 29.3.2013.

Suomenkielinen Scrum-sanasto. 2012. Luettavissa:

<http://scrumwell.files.wordpress.com/2012/01/suomenkielinen-scrum-sanasto-2012-v1-2.pdf>. Luettu: 14.4.2013.

Sytyke ry & Agile Finland ry. 2013. 376 vuotta ketteriä kokemuksia. Luettavissa:

<https://dl.dropbox.com/u/2022200/376%20vuotta%20ketteri%C3%A4%20kokemuksia.pdf>. Luettu: 29.1.2013.

Teikari, V. 2012. Don't panic - Ketterän kehityksen ostajan opas. Houston Inc. Helsinki.

Trafi 2012. Trafi siirtyi uuteen tapaan hankkia ja toteuttaa suuria tietojärjestelmiä. Luettavissa:

http://www.trafi.fi/tietoa_trafista/ajankohtaista/1961/trafi_siirtyi_uuteen_tapaan_hankkia_ja_toteuttaa_suuria_tietojarjestelmia. Luettu: 22.2.2013.

VersionOne. 2013. 7th Annual state of agile Version one development survey. Luettavissa: <http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf>. Luettu: 14.4.2013.

West, D. & Grant, T. 2010. Agile Development: Mainstream Adoption Has Changed Agility. Forrester Research Inc.

Woodward, E., Surdek, S. & Ganis, M. 2010. A practical guide to distributed Scrum. IBM Press/Pearson. Upper Saddle River (NJ).

Liitteet

Liite 1. Opinnäytetyössä käytetyt termit ja lyhenteet

Alkio	Katso Tuotteen kehitysjonon kohta.
Ei-toiminnalliset vaatimukset	Määrittelevät minkäläisten reunaehto- jen, esimerkiksi vas- teaikojen tai käytettävyyteen liittyvien seikkojen, rajoissa jär- jestelmä täyttää toiminnalliset vaatimuksensa (Pohjonen 2002, 28).
Inkrementaalinen	Inkrementaalinen tarkoittaa pala palalta täydentyvää ja kasva- vaa ohjelmistoa (Kruchten 2004, 189).
Item	Katso Tuotteen kehitysjonon kohta.
Iteratiivinen	Iteratiivisessa ohjelmistokehityksessä työ jaetaan tehtäväksi useassa iteraatiossa. Jokaisessa iteraatiossa toistetaan samoja työvaiheita eli tehdään määrittelyä, suunnittelua, toteutusta ja testausta, jolloin tuote kehittyy inkrementaalisesti. (Kruchten 2004, 60–61.)
Kehitystiimi	Itseohjautuva ja monitaitoinen ryhmä kehittäjiä. Tekee scrumtiimissä tuotteen kehitystyötä tuotteen kehitysjonon pohjalta. (Schwaber & Sutherland 2011, 5).
Ketterät menetelmät	Ketterät menetelmät tarkoittavat joukkoa kevyitä ohjelmisto- kehitysmenetelmiä, joille on tyypillistä tiivis yhteistyö kehittä- jien ja liiketoiminnan edustajien välillä, kasvokkain tapahtu- van viestinnän suosiminen, itseorganisoituvat tiimit ja tuote- versioiden tiheä toimitussykli (Agile Alliance 2013c; Haikala & Mikkonen 2011, 43).

Menetelmä	Katso Ohjelmistokehitysmenetelmä.
Määrittely	Määrittelyssä selvitetään mitä järjestelmän pitäisi tehdä. Määrittely tehdään analysoimalla vaatimusmäärittelyn vaatimukset ja johtamalla niistä järjestelmän toiminnallinen määrittely. (Pohjonen 2002, 31.)
Ohjelmistokehitysmenetelmä	Joukko ohjeita ja menettelytapoja, joissa kuvataan miten järjestelmän kehittämiseen liittyvät tehtävät suoritetaan (Pohjonen 2002, 68).
Product backlog	Katso Tuotteen kehitysjono.
Product owner	Katso Tuoteomistaja.
Päiväpalaveri	Enintään 15 minuuttia kestävä tilannepalaveri, jossa jokainen kehitystiimin jäsen kertoo vuorollaan mitä on tehnyt edellisen palaverin jälkeen, mitä tekee ennen seuraavaa palaveria ja onko hänen etenemisellään esteitä (Schwaber & Sutherland 2011, 9).
RUP	The Rational Unified Process. Iteratiivinen ohjelmistokehitysmenetelmä. (Haikala & Mikkonen 2011, 42.)
Scrum	Viitekehys monimutkaisten tuotteiden kehittämiseen ja ylläpitoon (Schwaber & Sutherland 2011, 3).
Scrummaster	Scrumtiimin palveleva johtaja, joka vastaa siitä, että Scrumtiimi pitäytyy Scrumin teoriassa, säännöissä ja käytännöissä (Schwaber & Sutherland 2011, 6).

Scrumtiimi	Itseohjautuva ja monitaitoinen tiimi, joka koostuu tuoteomistajasta, Scrummasterista ja kehitystiimistä (Schwaber & Sutherland 2011, 4).
Sidosryhmä	Projektin sidosryhmiin kuuluvat kaikki tahot, joihin projekti vaikuttaa hetkellisesti tai pidempään. Sidosryhmiä voivat olla esimerkiksi käyttäjä, alihankkija tai viranomainen. (Haikala & Mikkonen 2011, 155.)
Sprintin tehtävälista	Sisältää sprinttiin valitut tuotteen kehitysjonon kohdat sekä suunnitelman niiden toteuttamiseksi (Schwaber & Sutherland 2011, 13).
Sprintin demo	Katso Sprinttikatselmus.
Sprinttikatselmus	Tilaisuus, jossa sprintin päätteeksi tarkistetaan demonstroimalla sprintin aikana tehty tuoteversio ja kerätään siitä palautte (Schwaber & Sutherland 2011, 10).
Sprintin retrospektiivi	Sprinttikatselmuksen jälkeen pidettävä tilaisuus, jossa tarkastellaan miten sprintti meni ja missä asioissa seuraavassa sprintissä pitäisi parantaa (Schwaber & Sutherland 2011, 11).
Sprintin suunnittelu-palaveri	Palaveri, jossa Scrumtiimi suunnittelee mitä sprintin aikana tehtävä tuoteversio sisältää ja miten sen vaatima työ tehdään (Schwaber & Sutherland 2011, 8).
Sprintti	Sprintin suunnittelupalaverista, kehitystyöstä, päiväpalaverista, sprinttikatselmuksesta ja sprintin retrospektiivistä koostuva enintään kuukauden mittainen jakso, jonka aikana tehdään julkaisukelpoinen tuoteversio (Schwaber & Sutherland 2011, 7).

Toiminnallinen määrittely	Katso Määrittely.
Toiminnalliset vaatimukset	Määrittelevät mitä järjestelmän pitäisi tehdä, miten se toimii ulkopuolelta tarkasteltuna, miten eri sidosryhmät sitä käyttävät ja miten se kommunikoi ympäristönsä kanssa (Pohjonen 2002, 28).
Tuoteomistaja	Vastaa Scrumtiimissä kehitystiimin työn ja tuotteen arvon maksimoimisesta sekä tuotteen kehitysjonon hallinnasta (Schwaber & Sutherland 2011, 4).
Tuotteen kehitysjono	Järjestetty lista kaikista tuotteen vaatimuksista, ominaisuuksista, toiminnoista, korjauksista ja parannuksista (Schwaber & Sutherland 2011, 11–12).
Tuotteen kehitysjonon kohta	Tuotteen yhden vaatimuksen, ominaisuuden, toiminnon, korjauksen tai parannuksen kuvaus, prioriteettijärjestys ja työ-määräarvio (Schwaber & Sutherland 2011, 12).
Vaatusmäärittely	Eri sidosryhmien järjestelmälle asettamien toiminnallisten ja ei-toiminnallisten vaatimusten kerääminen ja kuvaaminen (Pohjonen 2002, 28).
Valmiin määritelmä	Scrumtiimin yhteinen näkemys siitä, mitä valmis työ tarkoittaa. Auttaa arvioimaan milloin tuoteversioon liittyvä työ on valmis (Schwaber & Sutherland 2011, 14.)
Vesiputousmalli	Eteenpäin kulkeva prosessi, jossa tietojärjestelmän kehittämisen vaiheet seuraavat suoraviivaisesti toisiaan esitutkimuksesta aina ylläpitoon saakka (Pohjonen 2002, 40).

XP

Extreme Programming. Iteratiivinen ja ketterä ohjelmistokehitysmenetelmä. (Haikala & Mikkonen 2011, 43.)

Ylläpito

Tuotantokäytössä olevan tietojärjestelmän pitäminen toimintakunnossa virheiden korjauksilla, jatkokehityksellä tai muilla muutoksilla (Pohjonen 2002, 37).

Liite 2. Haastattelukysymykset

Haastattelu opinnäytetyötä varten

Opinnäytetyö: Haaga-Helia Ammattikorkeakoulun Tietojenkäsittelyn koulutusohjelman opinnäytetyö ”Määrittelyn haasteet siirryttäessä vesiputousmallista ketteriin menetelmiin”.

Opinnäytetyön tekijä: Piia Lehtonen, puh: xxx-xxx xxxx, piia.lehtonen@xxxx.fi

Näkökulma ja rajaukset

Opinnäytetyössä ja haastattelussa tarkastellaan erityisesti räätälöityjen järjestelmien määrittelyä ja dokumentointia. Määrittelyllä tarkoitetaan sekä vaatimusmäärittelyä että toiminnallista määrittelyä eli sitä dokumentaatiota, jonka työstämisessä liiketoiminnan edustajat ovat tiiviisti mukana, jonka he hyväksyvät, johon testaus perustuu ja jota hyödynnetään järjestelmän ylläpidossa sen koko elinkaaren ajan. Dokumentaation osalta opinnäytetyön ulkopuolelle rajataan käyttäjien ohjeistus, tekninen dokumentaatio ja käyttöpalveluiden (esim. operointi) ohjeistukseksi tehty dokumentaatio sekä dokumenttien katselmointi. Opinnäytetyön ulkopuolelle rajataan myös projektinhallinta, kustannukset, ostaminen, kilpailutus sekä toimittajien ja asiakkaiden väliset sopimukset.

Haastatteluun vastanneiden nimiä tai organisaation tarkkaa nimeä ei julkaista opinnäytetyössä.

Kysymykset

1. Taustatiedot organisaatiosta ja haastateltavasta

- Organisaation koko?
- Oma roolisi organisaatiossa ja IT-projekteissa?
- Mistä ohjelmistokehitysmenetelmistä sinulla on kokemusta nykyisessä organisaatiossa ja aiemmissa työpaikoissa?

2. IT-projektit organisaatiossa

- IT-projektien määrä keskimäärin vuodessa?
 - ylläpito
 - uusien järjestelmien kehittäminen
- Työnjako toimittajien kanssa:
 - mitä tehdään itse?
 - mitä tekevät toimittajat?

3. Organisaatiossa käytetyt menetelmät

- Mitä ohjelmistokehitysmenetelmiä organisaatiossa on käytetty?
 - vesiputous
 - Scrum
 - RUP
 - ei menetelmää/ en tiedä
 - muu (mikä?)
- Millä perusteella käytetyt menetelmät on valittu?
- Mitä menetelmiä organisaatiossa aiotaan käyttää tulevaisuudessa? Miksi?
- Millä tavalla määrittely ja dokumentointi vaikuttavat menetelmän valintaan?

4. Dokumentointi ja asiakaskokemus eri menetelmissä

Vesiputousmalli (jos käytetty organisaatiossa)

- Minkälaisia dokumentteja on tehty vaatimusmäärittelyn ja toiminnallisen määrittelyn osalta?
- Mikä dokumentaatioissa on ollut:
 - hyvää?
 - huonoa?

RUP (jos käytetty organisaatiossa)

- Minkälaisia dokumentteja on tehty vaatimusmäärittelyn ja toiminnallisen määrittelyn osalta?
- Miten dokumentointi on muuttunut verrattuna vesiputousmalliin?
- Mikä dokumentaatioissa on ollut:
 - hyvää?
 - huonoa?
- Vastaako dokumentaatio liiketoiminnan vaatimuksia?
- Mikä määrittelyssä on ollut:
 - hyvää?
 - huonoa?
- Mitä etuja ja haasteita määrittelyssä on ollut verrattuna vesiputousmalliin?
- Onko kommunikointi ollut erilaista verrattuna vesiputousmalliin?
- Miten henkilöstöä on koulutettu ja valmennettu RUP: käyttöön?

Scrum (jos käytetty organisaatiossa)

- Minkälaisia dokumentteja on tehty vaatimusmäärittelyn ja toiminnallisen määrittelyn osalta?
- Miten dokumentointi on muuttunut verrattuna vesiputousmalliin ja RUP:iin?
- Mikä dokumentaatioissa on ollut:
 - hyvää?
 - huonoa?
- Vastaako dokumentaatio liiketoiminnan vaatimuksia?
- Mikä määrittelyssä on ollut:
 - hyvää?
 - huonoa?
- Mitä etuja ja haasteita määrittelyssä on ollut verrattuna vesiputousmalliin?
- Voiko määrittelyä tehdä ketterästi? Mitä vinkkejä ketterään määrittelyyn haluat antaa?
- Onko kommunikointi ollut erilaista verrattuna vesiputousmalliin ja RUP:iin?
- Onko valmiin määritelmä
 - sovittu?
 - kirjattu ylös?
- Miten henkilöstöä on koulutettu ja valmennettu Scrumin käyttöön?

5. Ketterien menetelmien soveltuvuus omaan organisaatioon

- Soveltuvatko ketterät menetelmät organisaatioosi?
- Voiko niitä käyttää kaikissa IT-projekteissa? (perustelee)
- Mitä kannattaa mielestäni huomioida määrittelyssä ja dokumentoinnissa siirryttäessä käyttämään ketteriä menetelmiä?