Henriikka Lampinen

TEKNISEN DOKUMENTAATION HAASTEET KETTERÄSSÄ JÄRJESTELMÄKEHITYKSESSÄ

JYVÄSKYLÄN YLIOPISTO
INFORMAATIOTEKNOLOGIAN TIEDEKUNTA
2019

TIIVISTELMÄ

Lampinen, Henriikka
Teknisen dokumentaation haasteet ketterässä järjestelmäkehityksessä
Jyväskylä: Jyväskylän yliopisto, 2019, 67 s.
Tietojärjestelmätiede, pro gradu -tutkielma
Ohjaaja(t): Halttunen, Veikko

Tässä pro gradu -työssä tarkastellaan teknistä dokumentaatiota ketterissä järjestelmäkehitysprojekteissa. Kirjallisuusosuus pohjustaa sitä seuraavaan empiirisen osuuden tutkimuskysymykseen *"Mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?"* lähestymällä aihetta olemassa olevan kirjallisuuden kautta. Aiempien tutkimusten perusteella haluttiin tunnistaa jo löydetyt käytännön haasteet teknisten dokumenttien tuottamisessa, organisaatioissa, joissa käytetään ketteriä metodeja.

Kirjallisuusosuuden lähtökohtana oli tarkastella ensin dokumentti ja dokumentaatio käsitteinä. Miten dokumentaatiota toteutetaan ja käytetään organisaatiossa sekä miksi dokumentaation tuottaminen on tarpeellista organisaatioille. Lisäksi käsitellään järjestelmäkehitys käsitteenä ja prosessina, sekä mitä organisaatiot tavoittelevat järjestelmäkehityksellä.

Organisaation tarpeet sekä dokumentaatiolle että järjestelmäkehitykselle tunnistettua, voidaan tarkastella dokumentaation prosessia järjestelmäkehityksessä ja tarkemmin ketterissä järjestelmäkehitysprojekteissa. Koska erilaisten dokumenttien tyyppien määrä on rajaton, tässä työssä keskitytään vain tekniseen dokumentaatioon ketterässä järjestelmäkehityksessä.

Empiirisen tutkimuksen tavoitteena oli haasteiden kautta tunnistaa ja ymmärtää syitä niiden takana. Lisäksi haluttiin löytää ratkaisuja ketterien järjestelmäkehitysprojektien dokumentoinnin haasteisiin, jotta organisaatiot saavuttavat dokumentaation tuoman hyödyn. Tuloksena havaittiin, että dokumentaation prosessia tulee johtaa samalla tavoitteellisuudella ja metodein, kuin itse järjestelmäkehitystä. Ketterien järjestelmäkehityksen roolien mukaisesti projektin tulosten johto lankeaa tuotteenomistajan vastuulle.

Asiasanat: dokumentti, dokumentaatio, järjestelmäkehitys, ketterä kehitys, agile, tekninen dokumentaatio

ABSTRACT

Lampinen, Henriikka
Challenges in technical documentation in agile system development
Jyväskylä: University of Jyväskylä, 2019, 67 pp.
Information Systems Science, Master's Thesis
Supervisor(s): Halttunen, Veikko

This Master's Thesis examines technical documentation in agile system development projects. The literary review prepares the empirical section for the research question *What are the challenges of producing, updating and using technical documentation in agile development projects?* approaching the topic through existing literature. Existing studies sought to identify practical challenges that have already be identified in producing technical documents, in organizations using agile methods.

The starting point of the study was to look at the document and documentation as concepts. How documentation is implemented and used in an organization and why documentation is needed for organizations. In addition, system development is addressed as a concept and a process, and what organizations are pursuing with system development projects.

Having identified the needs of the organization for both documentation and system development, the process of documentation in system development and more specifically in agile system development projects can be examined. Since the number of different types of documents is unlimited, this literature review focuses only on technical documentation in agile system development.

The objective of the empirical research was to identify and understand the causes behind the challenges. In addition, solutions to the challenges of agile system development project documentation were sought to help organizations reap the benefits of documentation. As a result, it was discovered that the documentation process should be managed with the same level of ambition and methods, such as system development itself. According to the roles of agile system development, the management of the project's results falls under the responsibility of the product owner.

Tags: document, documentation, system development, incremental development, agile, technical documentation

KUVIOT

KUVIO 1. Vesiputousmalli 14
KUVIO 2. Yksinkertaistettu agile-kehitysmalli 17

TAULUKOT

TAULUKKO 1. Kirjallisuusosuudessa esiin tulleet haasteet 28
TAULUKKO 2. Haastateltavien taustatiedot 34
TAULUKKO 3 Tekniset dokumentit, joita tuotettiin tai käytettiin projekteissa 38
TAULUKKO 4. Tutkimuksen haastatteluissa esiin tulleet haasteet 49

SISÄLLYS

TIIVISTELMÄ....2

ABSTRACT....3

KUVIOT....4

TAULUKOT....4

SISÄLLYS....5

1 JOHDANTO....7

2 DOKUMENTAATIO....9
2.1 Dokumentti....9
2.2 Dokumentit organisaatioissa....10
2.3 Dokumentaation hallinta....11
2.4 Yhteenveto....12

3 KETTERÄT KEHITYSPROSESSIT JÄRJESTELMÄKEHITYKSEN METODINA....13
3.1 Mitä on järjestelmäkehitys....13
3.2 Ketterät kehitysmenetelmät....15
3.2.1 Ketterät metodit käytännössä: Scrum-kehitysmalli....17
3.2.2 Roolit Scrum-prosessissa....18
3.3 Yhteenveto....18

4 DOKUMENTAATIO JÄRJESTELMÄKEHITYKSESSÄ....19
4.1 Järjestelmäkehityksestä syntyvä dokumentaatio....19
4.2 Dokumentaatio ketterässä järjestelmäkehityksessä....20
4.3 Tekninen dokumentaatio....21
4.3.1 Tarpeet tekniselle dokumentaatiolle....22
4.3.2 Tekniset dokumenttityypit....22
4.3.3 Haasteet ketterän kehityksen teknisessä dokumentaatiossa....24
4.3.4 Haasteet eri dokumenttityypeissä....26
4.4 Yhteenveto....28

5 TUTKIMUSMENETELMÄ JA TUTKIMUKSEN KULKU....30
5.1 Tutkimusmenetelmä....30
5.2 Tiedonkeruumenetelmä....31
5.3 Aineiston analysointimenetelmä....32
5.4 Tutkimuksen rajaukset....32
5.5 Haastateltavien tausta ja valinta....33
5.6 Tutkimuksen kulku....34
5.7 Tutkimuksen luotettavuus....36

5.8 Yhteenveto 36

6 TUTKIMUSTULOKSET 37
6.1 Tiedon tallennus 37
6.2 Tiedon kehitys ja päivittäminen 42
6.3 Tiedon jakaminen 43
6.4 Tiedon käyttö 44
6.5 Tiedon johtaminen 46
6.6 Ketterän kehityksen periaatteet dokumentaation prosessissa 48
6.7 Yhteenveto 49

7 TULOKSET JA JOHTOPÄÄTÖKSET 51
7.1 Tulosten vertailu 51
7.2 Teknisen dokumentaation hyödyt ketterissä kehitysprojekteissa 53
7.3 Teknisen dokumentaation haasteiden ratkaiseminen 54

8 YHTEENVETO 59

LÄHTEET 61

LIITE 1 HAASTATTELUKYSYMYKSET 65

LIITE 2 TAUSTATIEDOT JA MONIVALINTA TEKNISISTÄ DOKUMENTEISTA 67

1 JOHDANTO

Dokumentaatio on organisaatioille olennainen tapa tallentaa ja jakaa tietoa organisaation sisällä. Globaali liiketoiminta vaatii toimiakseen kattavaa dokumentaatiota, jotta yrityksen prosessit voidaan kommunikoida organisaation laajuisesti. Dokumentit ovat pääasiassa siirtyneet digitaaliseen muotoon, josta niitä on helppo jakaa internetin välityksellä. (Glushko & McGrath, 2005.)

Oman haasteensa dokumenttien jakamiseen ja käyttöön tuo järjestelmäkehitysprojektit. Maantieteellisesti jakautuneiden tiimien käyttö on lisääntynyt ja työn koordinointi on huomattavasti hankalampaa. Lisäksi työprosessit suurissa organisaatioissa, joissa kehitetään laajoja ja kompleksisia järjestelmiä, kommunikaation tarve kasvaa. Tällaisessa ympäristössä työskentely yhdessä ajantasaisen ja relevantin dokumentaation puutteen kanssa aiheuttaa viivästyksiä prosessissa. (Grinter, Herbsleb, & Perry, 1999.) Puutteellinen dokumentaatio saattaa myös vaikeuttaa yhteistyötä kehitysprosesseissa. Globaalissa järjestelmäkehityksessä dokumentointi, sen ylläpito ja tarkastus on erityisen tärkeää. (Moitra & Herbsleb, 2001.)

Nykypäivän yleisin tapa tuottaa tietojärjestelmiä on ketterät järjestelmäkehitysmetodit (Jeremiah, 2015). Ketterien metodien alle lukeutuu useita erilaisia projektinhallinnan metodeja, joten tässä tutkimuksessa keskitytään niistä käytetyimpään, Scrum-kehitysmetodiin (Measey & Radtac, 2015). Ketterä järjestelmäkehitys kuitenkin pyrkii arvojensa mukaan minimaaliseen dokumentaatioon (Beck ym., 2002), joka sotii perinteisen järjestelmäkehityksen periaatteita vastaan. Perinteissä järjestelmäkehityksessä asianmukaisen dokumentaation tuottaminen kuuluu jokaisen kehitysvaiheen loppuun (Mahalakshmi & Sundararajan, 2008).

Kirjallisuusosuus on luonteeltaan käsitteellisteoreettinen. Siinä lähestytään aihetta aiempien tutkimusten ja alan kirjallisuuden kautta. Aineisto kerättiin Jyväskylän Yliopiston tarjoamista tiedonhakuportaalin tietokannoista, käyttäen apuna IEEE Xplore ja Google Scholar hakukoneita. Näistä hakukoneista etsittiin aihetta käsitteleviä artikkeleita, konferenssijulkaisuja, standardeja ja kirjoja erilaisin hakusanoin. Lähdemateriaaliksi pyrittiin etsimään mahdollisimman uusia tai viimeisimmän tutkimustiedon sisältäviä tieteellisiä julkaisuja. Kirjallisuusosuudessa esitellään useita tutkimuksia, joissa on havaittu ketterien menetel-

mien käytöstä syntyviä ongelmia tiedonjaossa, juuri dokumentaation puutteeseen tai vajavaisuuteen liittyen. Metodin mahdollistaman joustavuuden tarjoaminen kehitysprosessiin aiheuttaa alati muuttuvia toiminnallisuuksia ja liiketoiminnan vaatimuksia järjestelmään, jonka myötä dokumentaatiota on vaikea pitää ajan tasalla.

Tutkimuksen tavoitteena on perehtyä tekniseen dokumentaatioon ketterässä järjestelmäkehityksessä. Erityisenä kohteena on tunnistaa teknisten dokumenttien tuottamiseen ja käyttöön liittyvät haasteet. Tekniseen dokumentaatioon luetaan tässä kirjallisuuskatsauksessa kaikki järjestelmäkehityksen aikana syntyvät dokumentit, joita kehityksestä vastaava tiimi tuottaa. Näillä dokumenteilla on suuri merkitys etenkin perehdytyksessä sekä järjestelmän ylläpitovaiheessa, jolloin täysin uudet kehittäjät pyrkivät ymmärtämään järjestelmää kokonaisuutena. Lisäksi tekninen dokumentaatio kuvaa miten järjestelmä päätyi nykytilaansa, joten se osaltaan myös dokumentoi liiketoiminnallisia ja projektijohdon päätöksiä järjestelmän osalta. (Garousi ym., 2015.)

Kirjallisuudesta teknisen dokumentaation osalta pyritään tunnistamaan ne prosessit, työvaiheet ja tekniset artefaktit, jotka olisivat tuotteen hallittavuuden kannalta kaikesta kriittisimmät asiat dokumentoida ketterissä järjestelmäkehitysprojekteissa. Lisäksi halutaan tunnistaa tavat, millä eri tavoin dokumentaatiota kulutetaan ja tuotetaan järjestelmäprojektin eri vaiheissa.

Empiirisessä tutkimuksessa rikastetaan olemassa olevista tutkimuksista löydettyjä haasteita, ja vastataan tutkimuskysymykseen: *Mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?* Haasteiden kautta pohditaan syitä niiden takana sekä halutaan löytää ratkaisuja, kuinka tuotteenomistajat käytännössä ratkaisevat nämä dokumentaation tuottamiseen ja sen johtamiseen liittyvät ongelmat.

2 DOKUMENTAATIO

Tässä kappaleessa esitellään dokumentti, dokumentaatio ja niiden hallinta käsitteinä ja prosesseina. Lisäksi perehdytään mikä on yritysten tarve käyttää dokumentteja, ja miten nykypäivän liiketoiminta on kehittynyt niin, ettei ilman dokumentteja pystytä toteuttamaan globaalia liiketoimintaa.

2.1 Dokumentti

Dokumentille on useita määritelmiä, englannin kielessä dokumentti terminä määritellään seuraavasti; dokumentti sisältää, säilyttää, kuljettaa ja välittää tietoa (Brown & Duguid, 2017, s. 172). Buckland (1991, s. 48) määrittelee dokumentin olevan geneerinen termi kaikelle fyysiselle resurssille, joka sisältää tietoa. Se voi kirjoitetun tekstin; kirjojen artikkeleiden ym. lisäksi olla diagrammi, kartta, kuva tai äänitallenne. Sprague (2006, s. 30) lisää tähän listaan vielä videon ja animaation. Yksittäiset dokumentit muodostavat dokumentaation, joka yhdistää dokumentin käsitteen tiedon säilytykseen ja hakuun (Buckland, 1991, s. 48).

Brown ja Duguid (2017, ss. 173–175) kuitenkin tarkentavat määritelmäänsä, että dokumentin ainoa tarkoitus ei kuitenkaan ole vain kantaa tietoa paikasta A paikkaan B, vaan osaltaan se myös luo, jäsentelee ja validoi tietoa. Dokumentin mahdollistamalla tiedon luomisella ja jäsentelyllä tarkoitetaan esimerkiksi uutisten tuottamaa lisäarvoa lukijalle. Tieto ei vain odota sen pakkaamista ja kuljettamista lukijoiden saataville, vaan journalistit keräävät jo olemassa olevia dokumentteja, valikoivat niistä osan, yhdistelevät ja muokkaavat niiden sisältämää tietoa sekä reflektoivat tiedon tarkoitusta. Näin lehdistö osaltaan tuottaa ja jäsentelee uuttaa tietoa lukijoilleen uutisten muodossa.

Dokumenttien tuottama validointi viittaa eri lähteiden tuottamaan lisäarvoon dokumentin sisältämälle tiedolle. Tiedon ollessa epävarmaa ihmiset nojautuvat toisen dokumentin lisätietoon eri lähteestä, näin lisäämällä tietoa tiedon päälle. Samasta lähteestä tuotettu lisätieto ei lisää olemassa olevan tiedon luotettavuutta. Toisaalta dokumentin kirjoittajan tai kirjoittajan edustaman ins-

tituution vaikutusvalta voi jo validoida yksittäisen dokumentin sisällön lukijalle. (Brown & Duguid, 2017, ss. 173-177.)

2.2 Dokumentit organisaatioissa

Liiketoiminnassa dokumentti on yksinkertaisimmillaan kirjoitettu teksti, joka kuvaa ja todistaa jälkikäteen jotain aiemmin tapahtunutta vuorovaikutusta, esimerkiksi kuittia rahanvaihdosta. Ensimmäinen ihmishistorian todistettu dokumentti olikin saven palaan kaiverrettu todiste veronmaksusta. Ajansaatossa dokumentit ovat siirtyneet savesta paperille ja paperilta suurilta osin elektroniseen muotoon. Saman skaalan muutos on tapahtunut dokumenttien käytössä; siinä, miten dokumenttien jakaminen mahdollistaa nykyisin yritysten globaalit liiketoimintaprosessit. (Glushko & McGrath, 2005, ss. 4-5.)

Vaikka dokumentaatio ei ole yrityksen liiketoiminnan keskiössä, voi dokumentaation generoida tuottoa tukemalla tuotetta, esimerkkinä tuotteiden käyttäjän manuaalit. Lisäksi dokumentit mahdollistavat organisaation sisäisen kommunikaation esimerkiksi sisältämällä liiketoimintaprosessien kuvaukset. (Sprague, 2006, s. 32.) Kaikki organisaatiot alasta riippumatta tarvitsevat tietoa ja tiedon jakamista. Täyttääkseen kirjoitetun tiedon tarpeet, ihmiset ovat kehittäneet laajan kirjon tallentaa tietoa: laskut, kuitit, paperinen raha, velkakirjat sekä lukematon määrä muita dokumentteja. Dokumentit järjestelevät liiketoiminnassa tapahtuvia vuorovaikutustilanteita ja sisältävät tiivistetysti tiedon, miten liiketoimintaa toteutetaan. Näin dokumentit mahdollistavat organisaatiota tuottamaan arvoa, jota ilman dokumentaatiota ei voisi synnyttää. (Glushko & McGrath, 2005, ss. 4-5.) Tässä työssä kuitenkin käytetään Spraguen (2006, s. 31) määritelmää elektronisesta dokumentin ominaisuuksista: Dokumentti on tilannekatsaus informaatiosta, joka:

- yhdistää monia kompleksisia tiedon tyyppejä
- esiintyy useassa paikassa verkostossa
- on riippuvainen muiden dokumenttien tuottamasta tiedosta
- päivittyy nopeasti tarpeesta
- omaa moniulotteisen rakenteen
- on monen käyttäjän ylläpidettävänä
- edustaa konseptia tai ideaa
- on rakennettu ihmisten ymmärrettäväksi
- säilytetään ja käsitellään yksikkönä

Tällä määrityksellä tässä tutkimuksessa lähestytään myös määritettä *Tekninen dokumentti,* joka avataan käsitteenä myöhemmissä kappaleissa. Tutkimuksessa myös oletetaan, että kaikki järjestelmäkehityksessä syntyvät dokumentit kirjataan elektroniseen muotoon. Näin ollen kaikki tutkimuksessa esitetyt dokumenttityypit omaavat edellä mainitut ominaisuudet.

2.3 Dokumentaation hallinta

Lee ja Hong (2002, s. 19) määrittelevät ja avaavat tiedonhallinnan perusvaiheet. Nämä ovat (1.) tiedon tallennus, (2.) kehitys, (3.) jakaminen ja (4.) käyttö. Näiden lisäksi dokumentin elinkaareen voidaan vielä lukea tiedon päivityksen vaiheen (Sprague, 2006, s. 32). Tässä tutkimuksessa kehitys ja päivitys käsitellään yhtenä kokonaisuutena.

Tallennuksen vaiheessa yksilön omaama hiljainen (tacit) tieto kerätään sisäisistä tai ulkoisista lähteistä ja siitä muodostetaan kirjoitettua tietoa (explicit), jolloin yhden yksilön hankkimaa tietoa jaetaan koko organisaation käyttöön. Dokumenttien tyyppien laajeneminen kuviksi ja ääneksi on luonut uuden tarpeen säilyttää ja linkittää näitä tiedostolohkoja toisiinsa. Tiedon tallennuksen jälkeen se organisoidaan ja analysoidaan, jotta sen perusteella voidaan tehdä strategisia tai taktisia liiketoimintapäätöksiä.

Tiedon jakaminen on prosessi, jossa tietoa jaetaan eri lähteistä tavoitteena luoda uutta tietoa tai ymmärrystä. Organisaatioiden rakenteiden monimutkaistuessa, on jopa mahdotonta toteuttaa globaaleja prosesseja ilman nykyteknologian tuomia tiedon jaon välineitä. Viimeisessä, tiedon käytön vaiheessa on tärkeää, että valittu järjestelmä hallita dokumentaatiota tukee loppukäyttäjän tiedonkäyttötarkoituksia. Tieto on oltava helposti saatavilla, vaikka käyttäjällä ei olisi paljoa tietokoneen käyttöosaamista. Loppukäyttäjä oppii nopeammin ja ymmärtää vaivattomammin uuttaa tietoa, kun saatavilla oleva tieto on monipuolista ja usean median välityksellä jaettua. (Lee & Hong, 2002, ss. 19-22.)

Päivärinta ja Munkvold (2005, s. 2) tiivistävät dokumentaatiolla haetut hyödyt seuraavasti. Organisaation dokumenttien hallinnalla pyritään:

- lisäämään sisäistä ja ulkoista viestintää
- lisäämään arvoa asiakkaille
- parantamaan tiedon luotettavuutta ja laatua ja näin vähentämään virheitä tuotteissa
- luomaan modernia kuvaa sidosryhmille
- parantamaan tehokkuutta ja joustavuutta asiantuntijatyössä
- parantamaan työn mielekkyyttä
- tallentamaan organisaation historiatietoa
- kustannussäästöihin

Näitä hyötyjä halutaan myös tunnistaa tähän tutkimukseen kuuluvista dokumenttityypeistä. Tutkimus keskittyy kuitenkin pääosin dokumentaation prosessissa esiintyviin haasteisiin. Dokumenttien tuottamaa hyötyä organisaatioille, jotka kehittävät järjestelmiä ketterin menetelmin tutkitaan empiirisen osuuden haastattelujen vaiheessa ja ne esitellään tutkimustulosten yhteydessä luvussa.

2.4 Yhteenveto

Tässä luvussa selvennettiin käsitteet dokumentti ja dokumentaatio, sekä niiden erot käsitteinä. Lisäksi keskityttiin dokumentaatioiden käyttöön organisaatiossa, ja haluttiin ymmärtää mitä hyötyjä dokumenttien tuottamisella pyritään tuottamaan organisaation liiketoiminnalle. Luvussa tunnistettiin, että organisaatiot voivat tuottaa hyvin laaja-alaista dokumentaatiota useisiin eri tarkoitusiin ja kanaviin monin eri tavoin. Dokumentit sisältävät yritykselle kriittistä informaatiota, jotta se kykenee tuottamaan asiakkailleen arvoa. Yhteenvetona voi todeta, että organisaatiot eivät kykene tehokkaaseen ja pitkäjänteiseen toimintaa ilman dokumentteja.

Seuraavissa luvuissa keskitytään tutkimuksen tutkimusaiheen käsitteiden ympärille; dokumentaatioon järjestelmäkehityksessä sekä teknisen dokumentaatioon. Tässä luvussa avatut käsitteet pohjustavat seuraavissa luvuissa esiintyviä dokumenttien tarkentavia määrityksiä tutkimusaiheeseen liittyen. Lisäksi dokumentaation prosessin perusvaiheet toistuvat tutkimuksen kirjallisuusosuuden yhteenvedossa sekä empiirisen osuuden haastattelun teemoituksessa.

3 KETTERÄT KEHITYSPROSESSIT JÄRJESTELMÄKEHITYKSEN METODINA

Tässä luvussa esitellään järjestelmäkehitys käsitteenä ja prosessina. Järjestelmäkehitys -käsitteen alle mahtuu lukematon määrä erilaisia prosesseja, viitekehityksiä ja metodeja. Tässä tutkimuksessa on esitelty vain yleisesti kutsuttu *perinteinen* järjestelmäkehityksen malli sekä sen vastakohtana nykypäivänä yleisesti käytössä oleva ketterä järjestelmäkehitysmalli.

3.1 Mitä on järjestelmäkehitys

Järjestelmäkehityksellä tarkoitetaan prosessia, jossa organisaatiolle rakennetaan tietojärjestelmä, joka tukee liiketoiminnan tarpeita. Prosessi sisältää valmistuvan järjestelmän myötä myös esimerkiksi järjestelmädokumentaation, käyttäjämanuaalin, tukisivustot ja järjestelmän käyttöön liittyvän datan. Projekti sisältää kaikki järjestelmäkehityksen keskeiset vaiheet alun järjestelmäkuvauksesta, kehitykseen, validoinnista ja testauksesta, käyttöönoton jälkeiseen ylläpitoon. Järjestelmäkehitysprosessilla on tavoite, budjetti ja aikarajoitteet valmistumiselle. (Sommerville, 2016, ss. 19-22.)

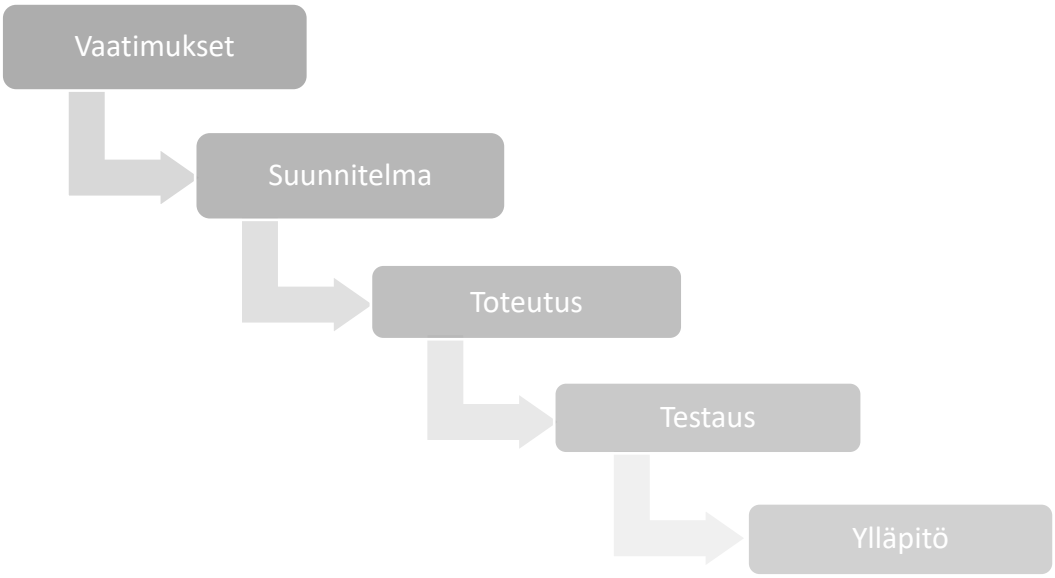
Järjestelmäkehityksessä voidaan tuottaa kahdenlaisia ohjelmistoja: geneerisiä tuotteita, joita voidaan myydä useammalle asiakkaalle, sekä kustomoituja yhdelle asiakkaalle kehitettyjä tuotteita. Kooltaan projektit skaalautuvat hyvin pienestä irrallisesta järjestelmästä kannettavassa laitteessa, suuriin globaalin asiakaskunnan käytössä oleviin internetin välityksellä jaettaviin pilvipalveluihin. (Sommerville, 2016, ss. 20-25.)

Monimutkaista järjestelmäkehitysprosessia on yksinkertaistettu luomalla prosessimalleja. Sommerville (2016, s. 45) luettelee kaikista yleisimmiksi malleiksi vesiputousmallin ja inkrementaalisen kehityksen. Vesiputousmalli, ollessaan näistä perinteisin ja vanhin, on esitelty alempana perinteisenä järjestelmäkehitysmetodina. Inkrementaalisessa kehityksessä järjestelmää kehitetään usean version kautta, jolloin jokaisen version jälkeen haetaan palautetta käyttäjiltä tai muilta sidosryhmiltä. Inkrementaalinen kehitysmalli on ketterien kehitysme-

todien keskeinen lähtökohta. (Sommerville, 2016, s. 50.) Ketterät metodit esitellään vesiputousmallin jälkeen seuraavassa alaluvussa.

Vesiputousmalli

Vesiputousmalli on ensimmäinen julkaistu järjestelmäkehityksen malli, joka saa nimensä siitä, että prosessi toteuttaa järjestelmäkehitystä askelma kerrallaan laskevasti. Se sisältää ja kuvastaa keskeisiä järjestelmäkehityksen vaiheita, jotka toistuvat järjestelmäkehityksessä prosessimallista riippumatta. (Sommerville, 2016, ss. 47-48.) Koska vesiputousmalli kuvastaa selkeästi järjestelmäkehityksen perusvaiheet, on se esitelty tässä työssä perinteisenä metodina. Kuviossa 1 kuvataan vesiputousmallin vaiheet.



KUVIO 1. Vesiputousmalli. Mukaillen Sommervillen mallia (2016, s. 47).

Vaatimusten määrittelyn vaiheessa projektitiimi ja asiakas sopivat kaikki kehitykseen menevät toiminnalliset ja ei-toiminnalliset vaatimukset. Toiminnallisilla vaatimuksilla tässä tarkoitetaan kaikkia toimintoja mitä projektin aikana kehitetään ja ei-toiminnallisilla toimintojen laadulliset määreet, kuten kuinka nopeasti toiminnallisuus toteuttaa sille annetun tehtävän. (Harris, 2018, ss. 205-207.)

Suunnittelun vaiheessa laaditaan kokonaisen järjestelmän arkkitehtuuri, toimintojen väliset suhteet ja määritellään vaadittava laitteisto, joka toteuttaa kaikki edellisen vaiheen toiminnallisuudet (Sommerville, 2016, s. 47). **Toteutuksen** vaiheessa toiminnallisuudet koodataan. Tässä vaiheessa kehitystiimi kehittää kaikki toiminnallisuudet ja niiden väliset riippuvuudet, kuten edellisessä vaiheessa suunniteltiin. Lisäksi jokaisen toiminnallisuuden ei-toiminnalliset vaatimukset implementoidaan. Tässä vaiheessa kehittäjät myös

testaavat koodin toimivuuden. (Harris, 2018, ss. 205-207.) **Testaamisen** vaiheessa järjestelmän eri toiminnallisuudet testataan joko manuaalisesti tai automaatiotestauksen työkaluilla. Testauksen tarkoituksena on verifioida, että kaikki alun vaatimukset ovat täytetty. (Harris, 2018, ss. 205-207.)

Ylläpidon vaihe on järjestelmän elinkaaren pisin vaihe. Järjestelmän ollessa käytössä, ylläpidolliset tehtävät koostuvat pääasiassa mahdollisten koodillisten virheiden korjaamisesta, joita ei aiemmissa vaiheissa saatu poistettua sekä olemassa olevien integraatioiden parantamisesta. Lisäksi ylläpitoon saattaa kuulua uusien ominaisuusien kehittäminen, kun liiketoiminnallisia tarpeita syntyy käytön yhteydessä. (Sommerville, 2016, s. 48.)

3.2 Ketterät kehitysmenetelmät

Ketterä kehitys tai inkrementaalinen kehitys on yleiskäsite joustaville johtamismetodeille ja työskentelytavoille, joiden tavoitteena on lisätä liiketoimintasovellusten relevanssia, laatua ja joustavuutta. Ketterän kehityksen tarkoituksena on valjastaa yrityksen resurssit niin, että se voi jatkuvasti tuottaa matalalla riskillä korkean liiketoimintahyödyn tuotteita, aika- ja budjettirajoitteiden sisällä. Ketterien metodien kautta on pyritty ratkaisemaan IT alan ikuisuusongelmia: määräaikojen ja budjetin ylittyminen, huonolaatuiset tuotteet ja tyytymättömät loppukäyttäjät. (Cooke, 2016, s. 34.) Ketteriä kehitysprojekteja yhdistää se, että tuotteen vaatimukset tapaavat muuttua projektin edetessä. Tämä on käytännössä yleiseksi havaittu tapaus ohjelmistokehitysmaailmassa, jonka vuoksi ketterät metodit ovat korvanneet em. vesiputousmallin käytön. (Sommerville, 2016, s. 50.)

Ketterän kehityksen periaatteiden perusteella on luotu ketterän ohjelmistokehityksen julistuksen *"Agile Manifeston"* (Beck ym., 2002) neljä toteamusta:

- Yksilöitä ja kanssakäymistä enemmän kuin menetelmiä ja työkaluja.
- Toimivaa ohjelmistoa enemmän kuin kattavaa dokumentaatiota.
- Asiakasyhteistyötä enemmän kuin sopimusneuvotteluja.
- Vastaamista muutokseen enemmän kuin pitäytymistä suunnitelmassa.

Measey ja Radtac (2015, ss. 5, 102) avaavat jokaisen edellä mainitun toteamuksen käytännön kautta seuraavasti. **Yksilöitä ja kanssakäymistä vs. menetelmiä ja työkaluja.** Vaikka menetelmät ja työkalut tuovat suuresti hyötyä kehitystiimille ja mahdollistavat ketterät periaatteet tiimissä, eivät ne kuitenkaan tuo arvoa asiakkaalle ilman kykeneviä ja motivoituneita työntekijöitä. Motivoituneiden työntekijöiden tunnuspiirteisiin kuuluu (1.) luottamus kollegoihin, (2.) intohimoinen ja suodattamaton keskustelu tärkeistä aiheista, (3.) tavoitteisiin sitoutuminen, (4.) vaativat vastuuta sitoumuksiin, (5.) tulosorientoituneisuus.

Toimivaa ohjelmistoa vs. kattavaa dokumentaatiota. Ilman dokumentaatiota ohjelmiston ylläpito ja tuki on todella hankalaa. Samaan aikaan kun ketterässä kehityksessä tarkoitukseen sopiva dokumentaatio on toimituksen ytimessä, ajaa kuitenkin toimivan ohjelmiston tuottaminen prioriteeteissa sen ohi. Ket-

terässä kehityksessä dokumentaatio pyritään pitämään mahdollisimman niukkana, ainoastaan sidosryhmille arvoa tuottavaa dokumentaatiota tuotetaan samassa tahdissa kehityssprinttien kanssa.

Asiakasyhteistyö vs. sopimusneuvottelu. Ketterässä kehityksessä pyritään asiakkaan ja toimittajan välille luomaan avoimen keskustelun ilmapiiri ja näin kehittää jatkuva yhteistyö koko projektin ajaksi. Tällä ei tarkoiteta, etteikö sopimuksia solmittaisi ollenkaan, vaan sopimuksen luonne keskittyy enemmän yhteistyön kirkastamiseen kuin sopimusneuvotteluihin. Näin sopimus mahdollistaa tuotteen tarkastelun, mukauttamisen, priorisoinnin ja kaikkien sidosryhmien välisen yhteistyön.

Muutos vs. suunnitelma. Ketterässä kehityksessä muutos on odotettavissa. Tästä syystä, mikäli projekti- tai kehityssuunnitelmia odotetaan tehtäväksi, tehdään ne todella korkealla tasolla ja niiden odotetaan muuttuvan. Niitä voidaan tarkentaa alemmille, yksityiskohtaisimmille tasoille projektin edetessä, mutta edelleen niidenkin oletetaan muuttuvat. Tiimit sitoutuvat ainoastaan sprintin mittaisiin työsuunnitelmiin, jossa sitoumus on yhteisen tavoitteen mukainen.

Agile Manifeston perusperiaatteiden päälle on rakennettu useita tuotteen toimittamiseen tähtääviä viitekehyksiä. Näistä suosituimpia ovat: Extreme programming, Scrum, DSDM (Dynamic Systems Development Method), Agile Projektinhallinta, Kanban, Lean-järjestelmäkehitys ja SaFE (Skaalattu Agile viitekehys) (Cooke, 2016, s. 39). Cooke (2016, ss. 34–35) summaa, että kaikki ketterät kehitysmetodit jakavat samat perusperiaatteet:

- Vähemmän etukäteissuunnittelua; suunnittelu tapahtuu parhaan mahdollisen sen hetkisen tiedon mukaan.
- Matalat prosessirakenteet mahdollistavat muutokset vaatimuksiin koko projektin ajan.
- Tuotteen korkea laatu ja yhtenäisyys.
- Riskien tunnistaminen prosessin alkuvaiheessa.
- Itseohjautuvuuteen kannustaminen; korkean liiketoimintahyödyn tavoittelu.
- Liiketoimintahyödyn jatkuva korostaminen ja sen kautta kehitysten priorisointi.
- Toimivien, testattujen liiketoimintavalmiiden toiminnallisuuksien tuottaminen.
- Jatkuvaan kommunikointiin kannustaminen liiketoiminnan eri yksiköiden ja projektitiimin välillä kasvattaa tuotteen relevanssia, käytettävyyttä ja laatua.

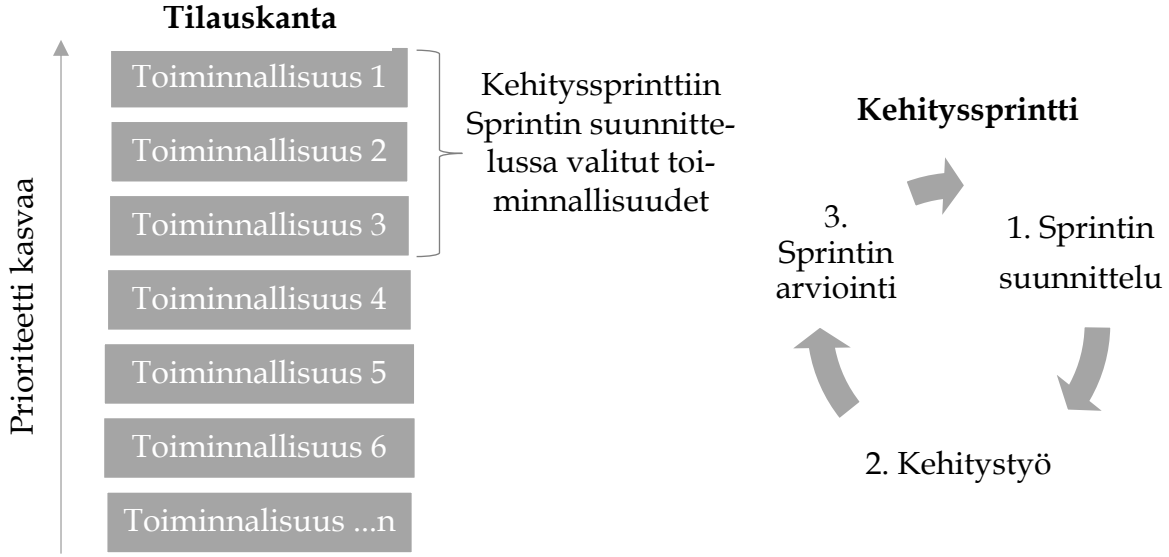
Koska samat perusperiaatteet toistuvat jokaisessa ketterän kehityksen metodissa, esitellään tässä tutkimuksessa vain yksi niistä, Scrum-kehitysmalli.

3.2.1 Ketterät metodit käytännössä: Scrum-kehitysmalli

Scrum-prosessi on ketterä lähestymistapa kehittää innovatiivisia tuotteita ja palveluja. Sitä käytetään viitekehyksenä projektinhallintaan. (Rubin, 2012, s. 1,13.) Se on kaikista ketterän kehityksen viitekehyksistä käytetyin malli (Measey & Radtac, 2015, s. 131). Scrum-mallissa esiintyy kaikki ketterien metodien peruskäsitteistö, työvaiheet ja roolit, joten se on esitelty tässä työssä esimerkkinä ketterästä viitekehyksestä.

Scrum perustuu kolmeen peruspilariin prosessin ohjauksessaan: (1.) mahdollistaa läpinäkyvyys kaikille sidosryhmille, (2.) jatkuva tuotteen tavoitteen tarkastaminen ja (3.) prosessin mukauttaminen niin, että vältetään hairahdukset tavoitteen saavuttamisen tieltä (Measey & Radtac, 2015, ss. 131–132). Kuviossa 2 on esitelty yksinkertaistettu malli Scrum-kehityksen perusprosessista.

Ketterässä kehityksessä kehitystyö alkaa keräämällä järjestelmän ominaisuudet prioriteettijärjestyksessä tilauskantaan (Backlog). Tilauskannan varhaisella priorisoinnilla varmistetaan, että kehityksessä on aina korkeimman prioriteetin ominaisuudet. Tilauskanta on aina suurempi, kuin mitä kehitystiimi pystyy yhden sprintin aikana toteuttamaan, joten tilauskannasta valikoidaan prioriteettijärjestyksessä kehityssprinttiin menevät ominaisuudet. Toteutusvaiheessa järjestelmää kehitetään iteratiivisesti, kehitysprinteissä, jotka vaihtelevat kestoltaan viikosta kuukauteen. Kehityssprinttien aikana itseorganisoituva ja useista osaamisalustoista koostuva tiimi kehittää täysin valmiin ja testatun toiminnallisuuden, joka voidaan välittömästi ottaa käyttöön. Sama prosessi toistuu niin kauan, kunnes jokin resursseista, esimerkiksi aika tai budjetti loppuu. Tässä tilanteessa kaikki tilauskantaan jäävät tuotteet jäävät kehittämättä. (Rubin, 2012, ss. 1-2.)



KUVIO 2. Yksinkertaistettu agile-kehitysmalli. Mukaillen Rubinin mallia (2012, s. 2).

3.2.2 Roolit Scrum-prosessissa

Scrum-kehitys tapahtuu tiimeissä, jossa jäsenet jaotellaan seuraaviin rooleihin. **Tuotteenomistaja** (Product Owner). Tuotteenomistaja on tiimin johtohahmo ja osallistuu toiminnallisuuksien suunnitteluun, priorisointiin ja arviointiin. Hän päättää mitkä ominaisuudet menevät kehityssprinttiin ja missä järjestyksessä. Hän vastaa siitä, että tuotteella on selkeä visio, mihin tiimi pyrkii ja kommunikoi sen kaikille osallisille. Tuotteenomistaja on vastuussa valmistuvan tuotteen onnistumisesta sen laajuuden ja laadun näkökulmasta, mutta myös rahallisesti ja ajallisesti. (Rubin, 2012, ss. 165-184.)

Scrum Master vastaa, että kaikki tiimissä ovat sisäistäneet Scrum-viitekehyksen arvot ja työskentelytavat. Hän ratkoo ongelmia ja poistaa tiimin edestä esteitä, jotka estävät tehokkaan työskentelyn Scrum-mallin mukaisesti. Scrum Master myös suojelee kehitystiimiä kesken Sprintin tulevilta muutoksilta ja häiriötekijöiltä. (Rubin, 2012, ss. 185-194.)

Kehitystiimi koostuu erilaisista teknisistä rooleista kuten: arkkitehti, kehittäjä, testaaja, tietokannan ylläpitäjä ja käyttöliittymäsuunnittelija. Tiimi on itseohjautuva ja omaa tarvittavat taidot laadukkaan järjestelmän tuottamiseen. Tiimi työskentelee yhdessä yhteisen Sprintin tavoitteen saavuttamiseksi. Jokainen jäsen osallistuu tilauskannan arviointiin, Sprintin suunnitteluun sekä prosessien ja tuotteen arviointiin ja muokkaamiseen. (Rubin, 2012, ss. 195-197.)

3.3 Yhteenveto

Tässä luvussa esiteltiin järjestelmäkehitys käsitteenä sekä siihen yleisesti kuuluvat prosessin vaiheet. Järjestelmäkehitys käsitteen alle kuuluu lukuisia erilaisia viitekehyksiä, jotka tarjoavat prosessimalleja järjestelmäkehityksen prosessin toteuttamiseen. Tämän tutkimuksen kannalta tärkeää on ymmärtää, mikä on ketterät kehitysmetodit, ja miten ne eroavat perinteisestä tavasta tuottaa järjestelmiä. Nämä erovaisuudet tunnistettua voidaan seuraavissa kappaleissa perehtyä mitä haasteita ketterien metodien käyttö organisaatioissa tuottaa dokumentaation prosessiin.

Luvussa esiteltiin myös yleisimpään ketterään metodiin, Scrum-viitekehykseen kuuluvat roolit. Näiden roolien mukaisesti eri järjestelmäkehityksen prosessit ja työvaiheet voidaan vastuuttaa tietyille henkilöille kehitystiimissä. Tämän tutkimuksen kannalta Scrum-roolit näkyvät haastateltavien valinnassa. Lisäksi roolien vastuualueiden mukaan haluttiin empiirisessä osuudessa perehtyä myös teemaan: *Tiedon johtaminen*.

4 DOKUMENTAATIO JÄRJESTELMÄKEHITYKSESSÄ

Tässä luvussa perehdytään dokumentaation rooliin järjestelmäkehityksen näkökulmasta. Lisäksi esitellään teknisen dokumentaation käsite, rajaamalla käsitteen alle lukeutuvat dokumenttityypit. Lopuksi keskitytään dokumentaation prosessin haasteisiin ketterissä metodeissa. Luvussa halutaan tunnistaa haasteet teknisen dokumentaation tuottamisessa ketterissä kehitysprojekteissa, joita kirjallisuudesta nousi esiin.

4.1 Järjestelmäkehityksestä syntyvä dokumentaatio

Järjestelmäkehityksessä dokumentaatio on tallenne käyttäjän tarpeista järjestelmään ja vastaa järjestelmän käyttöön ja toimintaan liittyviin tarpeisiin. Tällä tarkoittaan sitä, että dokumentaatiosta löytyy vastaukset siihen, mitä järjestelmän kehityksessä on tapahtunut, jotta päästiin tähän pisteeseen sekä tiedon, mihin tästä ollaan menossa. (Horch, 2003, s. 205.)

Järjestelmän dokumentti on kirjoitettu kuvaus, jolla on virallinen asema tai valtuutus olemassaololleen ja jota voidaan käyttää todistusaineistona aiemmasta toiminnasta. Tästä syystä sen oletetaan olevan oikeellinen ja ajantasainen, jotta sen puoleen voidaan nojautua tarpeen tullessa. (Parnas, 2009, s. 127.)

Tietojärjestelmien käyttöön ja kehitykseen liittyvä dokumentaatio koostuu laajasta skaalasta eri tyyppistä dataa. Skaalan toisessa päässä on hienojakoinen tieto, data, yksityiskohtainen tekninen määritelmä ja toisessa päässä taas suurijakoinen epämuodollinen data, dokumentti. Näiden kahden välille on mahdotonta kuitenkaan vetää selkeää rajaa, jossa data on tarkoitettu vain koneille ja dokumentit ihmisten käyttöön. Eri dokumentaation muodot ovat jatkumo skaalan päästä toiseen. (Glushko & McGrath, 2005, ss. 9-11.)

Järjestelmädokumentaatio vastaa seuraaviin tarpeisiin: (1.) se on sopimuksenmukainen, (2.) se auttaa kehitystiimiä ymmärtämään toiminnallisuudet, jotka kehitetään, (3.) se mahdollistaa kehitystiimiin kommunikaation tulevalle ylläpitotiimille (de Souza, Anquetil, & de Oliveira, 2005, s. 69). Hyvin suunniteltu

dokumentaatio ei vain auta lukijaa ymmärtämään järjestelmää, vaan myös vähentää koulutuksen ja järjestelmätuen tarvetta sekä osaltaan parantaa tuotteen yleistä mainetta. (IEEE Standard, 2011a.) Yleisesti dokumentin sisällön vaatimuksena on, että sen voi ymmärtää vain yhdellä tavalla. Kuitenkin, vaikka dokumentaatio ei täytä sisällöllisiä ja ajan tasaisuuden vaatimuksia, ei se käytännössä menetetä asemaansa aiemmin mainitun virallisen aseman vuoksi. (Parnas, 2009, s. 127.)

4.2 Dokumentaatio ketterässä järjestelmäkehityksessä

Dokumentaatio järjestelmäkehityksessä mukailee järjestelmäkehityksen vaiheita ja vastuita: hallinta, kehitys, testaus ja käyttäjädokumentaation tuottaminen. Kuten järjestelmäkehityksessä asiat muuttuvat, täytyy dokumentaationkin muuttua, jotta se edustaa jatkuvasti järjestelmän nykytilaa. (Horch, 2003, s. 206.)

Perinteisessä järjestelmäkehityksessä dokumentaatio tapahtuu jokaisen vaiheen loppuessa (Mahalakshmi & Sundararajan, 2008). Vesiputousmallissa seuraava vaihe ei voi alkaa, ennen kuin edellisen vaiheen dokumentaatio on valmis ja hyväksytty. Lisäksi, mikäli projektissa tulee muutoksia myöhäisemmissä vaiheissa, esimerkiksi liian kalliita vaatimuksia poistetaan vaatimusdokumenteista, täytyy päätökset hyväksyttää asiakkaalla ja kaikki edellisten vaiheiden dokumentaatio päivittää vastaamaan muutoksia. Luonnollisesti tämä viivästyttää koko projektin etenemistä. Vesiputousmallilla toteutetussa korkean tietoturvallisuusvaatimusten järjestelmän dokumentaatio tuotetaan suurilta osin jopa ennen kehittämisen aloitusta (Sommerville, 2016, s. 46,48.)

Agile Manifeston (Beck ym., 2002) yksi neljästä prinsiipistä on luoda toimivaa ohjelmisto kokonaisvaltaisen dokumentaation sijaan. Projektin hallinnan kannalta tämä tarkoittaa jatkuvaa suoraa keskustelua kehitysprosessin partnereiden kanssa sen sijaan, että luodaan runsaasti projektidokumentaatiota. Tämä ei tietenkään tarkoita, että dokumentaatiota ei pitäisi luoda ollenkaan. Dokumentaatio auttaa osaltaan esimerkiksi noviiseja projektin johtajia noudattamaan *Agile Manifeston* metodien periaatteita, esimerkiksi määritysten muutoksien estämisen kesken kehityssprintin (Cervone, 2011, ss. 19–22).

Dokumenttien luonne on ketterissä metodeissa erilainen kuin perinteisissä projekteissa. Ketterissä projekteissa dokumentit eivät paneudu yksityiskohtiin ja ovat epäformaaleja. Tämän lisäksi perinteisiä dokumentteja, esimerkiksi yksityiskohtaisia teknisiä määrityksiä ja suunnitteludokumentteja ei tuoteta ollenkaan, tai niiden sisältö on huomattavasti pelkistetympi. (IEEE Standard, 2011a.)

Kuten aiemmin on todettu, ketterän kehityksen perusperisaatteiden mukaisesti asiakkaalle on jatkuvasti tuotettava arvoa valmistuvan tuotteen muodossa. Dokumentaatiolla on tässä rooli vain, jos se täyttää arvon lisäämisen vaatimuksen. Stettina ym. (2012, s. 37) ehdottavatkin, että sidosryhmät voivat tilata dokumentaatiota kehitystiimiltä, aivan kuten järjestelmään uusia ominaisuuksia. Näin ollen myös dokumentaation tuottajalla on mielekkäämpää kirjoittaa heti käyttöön menevää dokumentaatiota, jolla tuotetaan välittömästi peräänkuulutettua arvoa.

Measey (2015, s. 107) toteaa että yleisin syy miksi liiketoiminta ja tekninen tiimi eivät ole samalla linjalla asioista, on yhteisen ymmärryksen puuttuminen. Hän väittää, että yksityiskohtaisten dokumenttien käyttö jopa kärjistää tätä ongelmaa, koska on hyvin epätodennäköistä, että osapuolet ymmärtävät sen täsmälleen samalla tavalla. Tästä syystä jatkuvan toimivan ohjelmiston toimittaminen varmistaa, että molemmilla osapuolilla on täysi näkyvyys ja ymmärrys mitä toimitetaan.

Kansainvälien IEEE Standardi (2011a) listaa dokumentaation tuottamiseen liittyvät aktiviteetit ketterissä järjestelmäkehitysprojekteissa:

1. Tuotettavien dokumentaatioiden tunnistaminen.
2. Sisällön ja tarkoituksen määrittely ja kehityksen aikataulutus.
3. Dokumentin tuottamiseen liittyvien standardien tunnistaminen.
4. Kehitys ja julkaisu suunnitelmien ja standardien mukaisesti.
5. Dokumenttien ylläpito määriteltyjen kriteerien mukaisesti.

Dokumentit, joita ketteristä kehitysprojekteista tulisi syntyä ovat: projektisuunnitelma, sprinttisuunnitelmat (Sprint Planning), vaatimusmäärittelyt, korkea tason suunnitelman ehdotelma, testisuunnitelmat, riskirekisteri, käyttäjätarinat (User Story), käyttötapausten kuvaukset, käyttäjäryhmien kuvaukset, tuotteen edistymiskäyrä (Burndown Chart), tehtävälistat, Scrum-raportit, Sprintin loppuarviot ja opetukset. (IEEE Standard, 2011a.)

Koska ketterät projektitiimit mittaavat edistymisensä toimivana ohjelmistona eikä dokumentaation ja aikarajoitteiden kautta, kokee Measey (2015, s. 109), että tiimi on pakotettu toimimaan yhdessä, jotta tavoite täytetään. Hän jopa väittää, että dokumentaatioon ja aikarajoitteisiin sitoutunut tiimi ajautuu erilleen ja tiimissä alkaa muodostumaan syyttelyn kulttuuria.

Dokumentaation määrä ja sisällön yksityiskohtaisuus riippuu esimerkiksi tiimin koosta ja maatieteellisestä levittäytymisestä. Pieni paikallinen tiimi saattaa luottaa vahvemmin henkilökohtaiseen kommunikointiin. Toisaalta suuri kansainvälinen tiimi, joka toimii eri aikavyöhykkeillä, joutuu tuottamaan yksityiskohtaisempaa dokumentaatiota kommunikaation tarkoituksiin ja myöhempää käyttöä varten. (IEEE Standard, 2011a.)

4.3 Tekninen dokumentaatio

Järjestelmädokumentaatio on määritelty (Ruhe, Moussavi, Garousi, Smith, & Garousi, 2013, s. 24) olemaan mikä tahansa artefakti, joka auttaa kommunikoimaan tietoa järjestelmästä kaikille sidosryhmille. Yleisesti järjestelmädokumentaatio voidaan nähdä olevan teknistä tai epäteknistä, ts. käytännönläheistä esimerkkinä käyttömanuaalit.

Teknisen dokumentaation tarkoituksena on helpottaa järjestelmän kehitystä sekä ylläpitotoimia. Lisäksi se auttaa kehittäjää hahmottamaan järjestelmää ja tehostaa näin osaltaan myös kehitystyötä. Tässä työssä teknisen doku-

mentaation käsitteen alle kuuluvat dokumentaation tyypit on esitelty kappaleessa 4.4.2.

4.3.1 Tarpeet tekniselle dokumentaatiolle

Dokumentaation tarkoituksena on auttaa järjestelmäkehittäjiä ymmärtämään järjestelmää. Tämän vuoksi on järjestelmästä synnytettävä dokumentaatiota kaikista sen kehityksen vaiheista. Garousi (2015, ss. 665–666) avaa eri järjestelmäkehityksen vaiheissa syntyvien teknisten dokumenttien käytön seuraavien prosessien vaiheiden alle:

- **Järjestelmän suunnittelunvaiheessa** syntyy järjestelmän arkkitehtuurin kuvaus, se auttaa ymmärtämään miten järjestelmän eri osat ovat keskinäisessä yhteydessä toisiinsa ja miten ne keskustelevat keskenään.
- **Kehityksen vaiheessa** tekstimuotoiset ja graafiset kuvaukset ja diagrammikaaviot, jotka ovat syntyneet suunnittelun vaiheessa ovat eniten käytettyjä ja auttavat kehittäjiä kehitystyössä ja testauksessa.
- **Ylläpitotoimien yhteydessä** dokumentaation tulisi auttaa kehittäjää ymmärtämään järjestelmän toimintaa arkkitehtuurin ja koodin ymmärryksen tasolla.

Grousi ym. (2015, s. 667) erottelevat vielä em. teknisen dokumentaation käytön ja hyödyllisyyden eri kategorioihin. Hyödyllisyys on eroteltu edellisistä käyttötapauksista ja sillä tarkoitetaan sitä, että dokumentaatio auttaa osaltaan ylläpidossa, kehityksessä tai johtamisen päätöksenteossa. Ylläpidon ja kehityksen apuna erityisen tärkeäksi muodostuu dokumentin ymmärtäminen, tiedon sisäistäminen. Parnas (2009) näkee, että tarpeellista olisi dokumentaatiokokoelma, jossa jokainen yksittäinen dokumentti kuvailee järjestelmää eri näkökulmasta. Näin yksittäisten dokumenttien ymmärrettävyys paranee kehitystiimin sisällä.

4.3.2 Tekniset dokumenttityypit

Järjestelmäkehityksen prosessista syntyy useita eri dokumentteja. Horch (2003) niputtaa ne hallinnollisiin dokumentteihin, kehitysdokumentteihin, testidokumentaatioon, käyttäjädokumentaatioon ja koulutusdokumentaatioon. Tässä työssä teknisen dokumentaation käsitteen alle valittiin näistä dokumenttityypeistä vain kehitysdokumentit sekä testidokumentaatio.

Kehitysdokumentteihin kuuluvat vaatimusten määritykset, järjestelmän arkkitehtuuriset sekä kehitystyön suunnitelmat, tietokannan määrittely ja rajapinnan määrittely (Horch, 2003, s. 211). Vaatimusten määritykset syntyvät vaatimusmäärittelyn tuloksena. IEEE standardin (2011b, s. 9) mukaan vaatimusmäärittely on monitieteinen sovitteleva toiminto, tuotteen tilaajan ja toimittajan tai kehittäjän välillä. Sen aikana luodaan ja ylläpidetään niitä vaatimuksia, joihin järjestelmän täytyy vastata. Toiminnon aikana asiakkaan tarpeet käänne-

tään järjestelmävaatimuksiksi. Vaatimusten määritykset tarjoavat ratkaisun suoraan johonkin ongelmaan, ovat mitattavissa, ovat selkeästi rajattuja, määrittelevät järjestelmän suorituskyvyn tietyn käyttäjän käytössä ja järjestelmän valmiudet toteuttaa toiminto. Lisäksi vaatimusmääritysten täytyy olla todennettavissa eli järjestelmä voi demonstroida vaatimuksen toteutumisen.

Ketterissä kehitysprojekteissa vaatimukset koostetaan käyttäjätarinan muotoon. Niissä käyttäjän vaatimukset kirjoitetaan lyhyillä yksinkertaisilla selosteilla (Chopade & Dhavase, 2017, s. 297) sekä käyttötapausskenaarioina, jotka kuivailevat toimintoa yksinkertaisilla avainsanoilla *'kun, olettaen, siten'* ja kattavat kaikki yleisimmät toiminnon käyttötapaukset (Papadopoulos, Kogias, Patrikakis, & Marinou, 2017, ss. 1543, 1546). Käyttäjätarinoiden lisäksi ketterien metodien tuotetusta järjestelmästä pitäisi myös syntyä seuraavia dokumentteja: Sprintin suunnitteludokumentit, käyttäjäryhmien määrittelyt, sprintin edistymiskäyrä sekä kehityssprintin päättyessä sprintin arviointidokumentti (IEEE Standard, 2018, s. 6).

Arkkitehtuuriset ja kehitystyönsuunnitelmat jakavat aluksi järjestelmän vaatimukset toiminnallisiin osiin, jonka jälkeen prosessi pureutuu jokaiseen osaan erikseen ja määrittelee, kuinka se käytännössä koodataan. Arkkitehtuurisen suunnitelman päätteeksi dokumentoidaan lopullinen ratkaisu, joka palvelee tulevaa ylläpitotiimiä järjestelmän kokonaisuuden ymmärtämisen lähtökohtana. Näissä dokumenteissa määritellään toimintojen suorituskyky, tietokantavaatimukset sekä kaikki rajapinnat, joita toiminnallisuus käyttää kommunikoidakseen sisäisten tai ulkoisten toimintojen kanssa. (Horch, 2003, ss. 215-216.) Arkkitehtuurin dokumentaatiolla pyritään muodostamaan ymmärrys järjestelmän rakenteesta ja muodosta, sisältäen perustelut edellisille. Arkkitehtuurin kuvaus asettaa suunnitelmaan pääasialliset päätökset, jotka hallitsevat koko järjestelmää. Tämä kuvaus tukee suunnitteluprosessia dokumentoimalla hyväksytyt ratkaisut. (Taylor & Van Der Hoek, 2007.)

Dokumentoimalla päätökset, jotka tehtiin suunnittelu-, kehitys- ja ylläpitovaiheissa auttaa uusia kehittäjiä sisäistämään syyt päätösten takana ja jo toteutetut tekniset ratkaisut. Eniten dokumentaatiota käyttävätkin uransa alussa olevat kehittäjät, joilla oli työuraa takana alle 5 vuotta. (Garousi ym., 2015, s. 674.) Ilman dokumentaatiota ainoa paikka, mistä olemassa olevasta järjestelmästä saa tietoa, on lähdekoodi.

Kehitysvaiheessa syntyvää lähdekoodia voisikin kutsua kattavaksi dokumentaatioksi, sillä se sisältää kaikki teknologiset aspektit, mitä järjestelmä pitää sisällään. Parnas (2009) kuitenkin muistuttaa, että käytännössä koodi ei ole tarpeeksi tiivistetty tiedonmuoto järjestelmästä. Se sisältää valtavasti pienijyväistä informaatiota, joka ei ole tarpeellista kokonaiskuvaa hahmotettaessa. Tämän vuoksi lähdekoodia ei oteta mukaan teknisen dokumentaation määritelmään tässä tutkimuksessa.

Koodin kommentointi toisaalta on kehittäjille hyvä keino avata luonnollisella kielellä, mitä koodi käytännössä tekee. Ylläpitovaiheessa nämä koodin kommentit ovat kehittäjillä suurimmassa käytössä verrattuna muihin dokumentteihin. (Garousi et al., 2015, s. 672.) Tämän vuoksi koodin kommentointi lisättiin teknisen dokumentaation määritykseen.

Tietokannan ja rajapinnan määrittelyn dokumentti on tarpeellinen silloin, kun projektissa on useita tai monimutkaisia tietokantarakenteita tai rajapintoja. Tilanteissa, joissa järjestelmäkehityksessä luodaan tai muokataan huomattavasti nykyisiä tietokantoja, on tarpeellista luoda tietokantaan liittyvä dokumentaatio. Lisäksi useat ja monimutkaiset kehitettävät rajapinnat pitäisi määritellä, suunnitella ja dokumentoida yhteen paikkaan, jotta kaikki osapuolet voivat nähdä niiden kuvaukset. (Horch, 2003, ss. 216-217.)

Testidokumentaatio sisältää kaikki järjestelmän testaamiseen liittyvät dokumentit; alkaen testaussuunnitelmasta ja päättyen loppuraporttiin. Testauksen vaiheessa järjestelmän toiminnallisuuksia verrataan alkuperäisiin vaatimuksiin ja yritetään löytää järjestelmästä vikoja. Testisuunnitelma koostetaan jo määritysten vaiheessa. Näin säilytetään toiminnallisuuden mitattavuus ja testattavuus. Tämän jälkeen luodaan yksittäiset testitapaukset, jotka testaavat järjestelmän loogisia kokonaisuuksia. Testaamista varten generoidaan testidataa, jolla pyritään ensisijaisesti todistamaan, että järjestelmä toimii määritysten mukaisesti. Tämä data sisältää kaikki sallitut arvot, jotka järjestelmä hyväksyy syötteenä. Mutta tärkeimpänä tässä vaiheessa ovat ne arvot, joita järjestelmä ei saa hyväksyä. Lopuksi muodostetaan yksityiskohtaiset testimenetelmät, jotka kuvailevat askel askeleelta, kuinka jokainen testi etenee. Kaikesta edellä mainituista toiminnoista koostetaan testiraportit, jotka kuvailevat oletetut tulokset sekä todelliset tulokset. Nämä raportit määrittelevät milloin testaaminen päättyy. Ne ovat testaamisprosessin viimeinen vaihe. (Horch, 2003, ss. 217-219.)

Testauksen dokumentaation (testaussuunnitelma, testitapaukset, testimenetelmät, testausraportti) tavoitteena on auttaa tulevaisuuden sidosryhmiä ymmärtämään perustelut testauksen toteutuksen takana. Dokumentoimattomien testien taustalla olevan ajattelun kokoaminen jälkikäteen uudestaan, ilman teknistä tukea, on prosessina hankala tai jopa mahdoton. (Tilley & Parveen, 2012.)

Näiden lisäksi de Souza ym. (2006, ss. 34–35) luettelevat muita teknisiä dokumentteja, joita kirjallisuudessa on mainittu. Näitä ovat esimerkiksi: järjestelmän kuvaus, datamalli, luokkakuvaukset, liiketoiminnan prosessit sekä algoritmien kuvaukset. Kaikki edellä mainitut dokumentit ovat koottu yhteen tämän tutkimuksen haastattelun monivalintapohjaan (liite 2).

4.3.3 Haasteet ketterän kehityksen teknisessä dokumentaatiossa

Järjestelmäkehityksen dokumentaatiossa ja sen käytössä on löydetty useita haasteita, Zahedi (2016, s. 1004) summaa eri tutkimuksista löydetyt yleiset tiedon jaon haasteet ja niiden seuraukset seuraavasti:

1. Heikko ja vaikeaselkoinen dokumentaatio vaatimuksista
2. Vaikeus löytää oikeaa tiedon lähdettä puuttuvan tai vanhentuneen dokumentaation takia
3. Heikko organisatorinen muisti oleellisen dokumentaation puuttuessa.

Ketterässä kehityksessä pyritään minimoimaan dokumentaatio ja keskittymään vain toimivan järjestelmän luomiseen. Kuitenkin joissain projekteissa on tunnistettu, että jopa puolet projektin resursseista kuluu dokumentaation hallintaan liittyviin toimenpiteisiin (Myklebust, Stålhane, Hanssen, Wien, & Haugset, 2014, s. 1).

Teknisessä dokumentaation tuottamisessa toistuvat samat haasteet kuin ketterän kehityksen muussakin dokumentaatiossa. Useat kehitystiimit tuottavat joko liian paljon tai liian vähän dokumentaatiota. Tästä muodostuu kysymys, kuinka paljon dokumentaatiota on tarpeeksi? Käytännöstä löytyy paljon tapauksia, joissa Legacy-järjestelmästä (vanhasta, korvattavasta järjestelmästä) ei löydy tarpeeksi dokumentaatiota tai se on huonolaatuista. Puutteelliset, epäjohdonmukaiset ja vanhentuneet dokumentit ovat järjestelmäkehityksessä yleistä. Lisäksi dokumentaatio prosessina koetaan kalliiksi, hyödyttömäksi ja hankalaksi ylläpitää. (Garousi ym., 2015, s. 665.)

Lethbridgen (2003, s. 36) tutkimuksessa melkein 70% vastaajista oli väittämän "Dokumentaation on aina vanhentunutta verrattuna järjestelmän nykytilaan" jokseenkin tai täysin samaa mieltä. Kuitenkin samoista vastaajista yli 80% koki, että dokumentaatio voi olla hyödyllistä, vaikka se ei aina ole ajan tasalla.

Ketterien menetelmien vastaus dokumentaation eliminoimiseen kehitystyön vaiheessa on korvata kirjallinen dokumentaatio kehittäjän ja käyttäjän välisellä epävirallisella kommunikaatiolla. Souza ym. (2005, s. 69) toteavat tutkimuksessaan, ettei epävirallinen keskustelu voi korvata kirjallista dokumentaatiota kommunikoinnin välineen, kun kehittäjien pitää siirtää tietoa uusille kehittäjille. Ongelma korostuu etenkin, jos kehitystiimistä lähtee henkilöitä (Stettina, Heijstek, & Fægri, 2012).

Aikaisimmissa tutkimuksissa (Zahedi ym., 2016) on todettu että ketterän kehityksen peräänkuuluttaman tacit-tiedonjaon korostaminen vaikuttaa suoraan negatiivisesti dokumentaation tuottamiseen ja ylläpitoon. Sosiaalisiin vuorovaikutustilanteisiin luottava asiakasprosessi, vaikuttaa vaatimusten keräämiseen esimerkiksi niin, että mahdolliset lisätiedot käydään kysymässä suoraan liiketoiminnan edustajalta ja tallennetaan esimerkiksi valkotauluille ja henkilökohtaisiin muistivihkoihin. Tämän vuoksi ketterin menetelmin tuotetun projektin projektitietämys on epäselkeää ja sirpaloitunut vain lähdekoodiin ja testitapauksiin. Tämä epäselvä tieto on sittemmin tallennettu epäjohdonmukaisesti dokumentteihin.

Lisäksi kehittäjien muistin varaan jäävä tieto on vaarassa kadota ilman dokumentaatiota. Garoushi (2015) muistuttaa, että kehittäjille syntyy ajan kuluessa myös toimialan domain-kohtaista tietoa. Se on kriittinen osaamisalue ymmärtää, jotta olemassa olevat dokumentaatiot avautuvat lukijalle. Lisäksi suuremmissa tiimeissä henkilökohtaisen tiedon jakaminen kaikille sidosryhmille ilman kirjallista dokumentaatiota on lähes mahdotonta.

Kuten aikaisemmin on todettu, dokumentaation kirjoittamisen tulisi noudattaa kehityksen vaiheita. Ketterän tiimin itseohjautuvuus määrittää, että yksittäinen tiimin jäsen saattaa toteuttaa useita eri tehtäviä kehityssprintin aikana. Etenkin lyhyissä kehityssprinteissä kaikkien osallistuminen kaikkiin työtehtäviin on erityisen tärkeää. Näin dokumentaatio pidetään samassa tahdissa kehityksen kanssa. (IEEE Standard, 2011.) Stettina ym. (2012) kuitenkin spekuloivat,

että aikapaineiden johdosta dokumentaation tehtävät nimitetään yhdelle henkilölle, yleensä koodaustaidoiltaan heikommalle kehittäjälle. Clear (2003) mainitsee, että dokumentaation prosessi yleisesti nähdään projektin ulkoisena tehtävänä. Sen sijaan, että se nähtäisiin luonnollisena osana kehitysprosessia, jota tuotetaan linjassa järjestelmän kanssa. Hän yleistää kehittäjillä olevan mentaliteetti, että koodi on ensisijaista ja kaikki muu tulee sen jälkeen. Huomionsa jälkeen Clear (2003) muistuttaa, että dokumentaatio voi toimia myös ajattelun tukena, joka kirjoittamisen aikana voisi siirtää ajatukset seuraaviin vaiheisiin.

Vaikka dokumentti olisi alun perin täsmällisesti kirjattu, monissa tapauksissa suurimmaksi ongelmaksi dokumentaatiossa on syntynyt se, että sen annettiin jäädä järjestelmäkehityksen edetessä jälkeen. Tällöin kehityskaaren loppupään vaiheissa, testauksessa ja ylläpidossa, dokumentaatio on täysin hyödytön. (Horch, 2003, s. 206.) Horch (2003, s. 205) kuvailee yleistä tilannetta järjestelmäkehitysprojektissa niin että suurimmassa osassa järjestelmäkehitysprojekteista ei ole kirjattu ylös, miten järjestelmä päätyi nykytilaansa. Ongelmat alkavat jo alun vaatimusten niukoista määritelmistä. Niukoilla vaatimuksilla järjestelmän suunnittelu vain kehittyy projektin edetessä. Selkeän suunnitelman puuttuessa koodi pyrkii vain jäljittelemään alati kehittyvää suunnitelmaa sen sijaan että se jalkauttaisi sitä. Kun koodi ja toiminnallisuus eivät kohtaa, myös testaaminen perustuu pelkän koodin toiminnan varmistamiseen ei niinkään toiminnallisuuden peilaamiseen alkuperäisiin vaatimuksiin. Tällaisesta prosessista syntyvä, käyttäjälle jäävä dokumentaatio on hyvin puutteellinen tai jopa virheellinen.

4.3.4 Haasteet eri dokumenttityypeissä

Vaatimusmäärittelyjen dokumentaatiossa ketterissä menetelmissä on löydetty monia haasteita. Ramesh ja Cao (2014) listavat minimaalisen dokumentaation vaatimuksista tuottavan erityisesti ongelmia kommunikaatiollisesti haasteellisissa tilanteissa, joissa esimerkiksi henkilökunta vaihtuu, vaatimukset muuttuvat nopeasti, asiakas on hankalasti saavutettavissa tai järjestelmän kompleksisuus kasvaa. Toiseksi ongelmaksi muodostuu ei-toiminnallisten vaatimuksien, kuten turvallisuuden ja järjestelmän skaalautuvuuden dokumentointi vaatimusmäärittelyn aikaisessa vaiheessa. Jatkuva asiakaskontakti ja sitä kautta palautteen hakeminen saattaa ohjata toimintaa vain järjestelmän käyttökokemuksen parantamiseen, jolloin esimerkiksi turvallisuusaspekti jää huomioimatta.

Heikkilä ym. (2015) korostavat että puutteellinen dokumentointi vaatimusmäärittelyistä hankaloittaa niiden jäljitettävyyttä takaisinpäin alkuperäiseen asiakasvaatimukseen. Lisäksi he mainitsevat, että etenkin laajemmissa kehitysprojekteissa yksittäiset käyttäjätarinat eivät ole riittävä ainoana dokumentaation lähteenä. Käyttäjätarinat kuvailevat asiakkaan vaatimuksia hyvin yleisesti, eivätkä näin sisällä kaikkea tarinaan liittyviä selityksiä (Desharnais, Kocatürk, & Abran, 2011.). Vaatimusmäärittelyistä syntyvän dokumentaation tulisikin olla laajempaa kuin pelkät käyttäjätarinat ja niiden sisältämät skenaariot, jotka ovat itsessään liian suppeita kuvauksia kokonaisista toiminnallisuuksista. Yksittäinen tarina tarvitsee yksityiskohtaisempaa tietoa dokumentaatiota

varten. Lisäksi toisistaan erillisistä tarinoista tulisi pystyä muodostamaan hierakinen kokonaisuus. (Heikkila, Damian, Lassenius, & Paasivaara, 2015.)

Lopulta dokumentaation avulla syntynyt lähdekoodi pitäisi pystyä jäljittämään alkuperäiseen vaatimukseen ja alkuperäiset hyväksytyt vaatimukset pitäisi olla luotettavasti dokumentoitu. Ilman tällaista jäljitettävyyttä, loppukäyttäjä saa tuotteen, joka tekee kyllä jotain mutta ei mitään sellaista, mitä alun perin tarvittiin tai haluttiin. (Horch, 2003, s. 206.)

Testidokumentaation tuottamisen ongelmana on löytää sen luonnollinen päätepiste. Tilley ym. (2012) nostavat esiin ongelman, että testitapauksia voidaan ajaa loputtomasti järjestelmää vastaan. Jokaisen testitapauksen dokumentaatioon käytetty aika on pois itse testaamisesta. Testaamisen prosessiin kuuluu testitapausten suunnittelu, toteutus ja tuloksien tulkinta. Lisäksi he mainitsevat, että testidokumentaation täydellinen puuttuminen on yleistä etenkin ketterissä kehitysprojekteissa, joissa itse testitapaukset oletetaan palvelevan dokumentaation roolissa.

Tutkimuksissa (Große, Jungmann, & Drechsler, 2015, s. 109) on myös todettu, että vaikka tekniseen dokumentaatioon on kehitetty erilaisia malleja, se ei siltikään täytä vaatimuksia olla ymmärrettävä, kokonaisvaltainen, käytännöllinen ja paikkaansa pitävä. Em. tutkimuksessa myös huomautetaan, että kehittäjät välttävät dokumentaatioon kuuluvia tehtäviä sen ollessa aikaa vievää, ja näin nähdään vievän työaikaa luovimmilta työtehtäviltä.

Yleinen ongelma onkin yksityiskohtaisen dokumentaation ymmärrettävyys. On hyvin epätodennäköistä, että liiketoiminnan sidosryhmän edustajat ja tekniset tiimit ymmärtävät saman dokumentin sisällön samalla tavalla (Measey & Radtac, 2015, s. 107). Ketterän kehityksen perusperiaatteet kehottavatkin välttämään yksityiskohtaista teknisen tuen dokumentaatiota sekä yksityiskohtaisia teknisiä määrityksiä. Tämä johtaa siihen, että teknisen dokumentaation kirjoittajilla ei ole käytettävissä lähdedokumentaatiota, josta voidaan yleistää ominaisuuksien yksityiskohtia (IEEE Standard, 2011a).

Teknisen dokumentaation luominen ja kulutus keskittyy enemmän yksinkertaisiin, mutta tehokkaisiin dokumentteihin, kuin kompleksisiin ja aikaa vieviin dokumentteihin (Lethbridge ym., 2003, s. 38). Sanamääriltään suuret dokumentit myös laskevat projektin kustannustehokkuutta (Garousi ym., 2015, s. 676). Dokumentaation kirjoittaminen nähdään raskaana sivutehtävänä (Stettina ym., 2012). Kehittäjät suosivatkin ja päivittävät niitä dokumenttikirjastoja, joista näkevät suurimman arvon työlleen. Lisäksi, lyhyen kommentin jättäminen esimerkiksi bugiraporttiin ei vaadi paljoa ponnistelua. Samalla periaatteella voi myös perustella koodin kommentoimista, koska se on lyhyt kirjoitus ja samassa tiedostossa koodin kanssa. Näin dokumentaation ylläpitotyön vaiva pysyy matalana. (Lethbridge ym., 2003, s. 38.) Myös UML-kaavioiden päivittäminen koettiin vähemmän työtä keskeyttävänä ja vaivattomampana kuin tekstimuotoisen dokumentaation (Stettina ym., 2012). Spinellis (2010, s. 18) kuitenkin korostaa että koodin kommentoinnissa toistuu samat haasteet kuin muussakin kehityksessä; sen pitäminen ajan tasalla. Pahimmassa tapauksessa koodi muuttuu, mutta kommentti ei, joka johtaa epäjohdonmukaisuuksiin sen lukijalle. Koodi tekee jotain muuta, mitä sen kommentti väittää sen tekevän. Spinellis (2010) alleviivaa, että kommentit pitäisi olla viimeinen keino, miten koodia tulisi do-

kumentoida. Huonolaatuinen koodi, joka ei avaudu lukijalle pitäisi ensisijaisesti kirjoittaa uudestaan, ei kommentoida auki. Toisin kuin nämä nopeat pienen tason kommentoinnit, korkean tason dokumentit eivät menetä relevanssiaan yhtä nopeasti, mikäli pienet yksityiskohdat järjestelmässä muuttuu. Näin ollen niitä ei nähdä yhtä tärkeiksi päivittää. (Lethbridge ym., 2003.)

4.4 Yhteenveto

Tässä kappaleessa esitettiin kirjallisuudessa löydetyt haasteet teknisen dokumentaation tuottamisessa ja käyttämisessä ketterissä järjestelmäkehitysprojekteissa. Tulokset jaotellaan tässä yhteenvedossa dokumentaation prosessin vaiheiden haasteisiin, dokumentaation johtamisen haasteisiin, sekä ketterän kehityksen periaatteiden omaksumiseen liittyviin haasteisiin (taulukko 1). Dokumentin prosessin vaiheet mukailevat luvussa 2.3. esiteltyjä tiedonhallinnan perusvaiheita. Nämä olivat tiedon tallennus, tiedon kehitys ja päivitys, tiedon jakaminen sekä tiedon käyttö (Lee & Hong, 2002, s. 19). Lisäksi kahdeksi viimeiseksi teemaksi valittiin tiedon johtaminen sekä ketterän kehityksen periaatteet dokumentaatiossa. Johtamisen teema lisättiin Scrum-roolien pohjalta. Tutkimukseen valikoitiin vain tuotteenomistajan roolissa tai työtehtävissä työskenteleviä henkilöitä. Tuotteenomistajan toimiessa myös tiimin johtohahmona haluttiin johtamisen näkökulma sisällyttää tutkimukseen. Ketterän kehityksen periaatteet lisättiin teemaksi, jotta dokumentaation prosessin ketteryyttä voidaan arvioida tutkimuksessa. Kirjallisuusosuuden perusteella pystytään vastaamaan osittain tutkimuskysymykseen: *Mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?*

TAULUKKO 1. Kirjallisuusosuudessa esiin tulleet haasteet

Teema	Haasteet
Tiedon tallennus	Resurssien riittämättömyys
	Puutteellinen dokumentaatio
	Puuttuva dokumentaatio
	Vaatimusten huono jäljitettävyys
	Liian raskas dokumentaatio
	Heikko organisatorinen muisti
	Dokumentaation tuottaminen epävirallisiin kanaviin (muistikirjat, white boardit)
	Virheellinen dokumentaatio
Tiedon kehitys ja päivitys	Resurssien riittämättömyys
	Dokumentaation viemä aika muilta työtehtäviltä
Tiedon kehitys ja päivitys	Nopeasti vaihtuvat määritykset
	Dokumentaatio ei ajan tasalla
	Raskaan dokumentaation ylläpito
	Motivaation puute
Tiedon jakaminen	Dokumentaation sirpaloituneisuus
	Tiedon huono löydettävyys
	Dokumentaatiota ei tuotettu yhteisiin kanaviin

Tiedon jakaminen	Tieto löydettävissä vain lähdekoodista
Tiedon käyttö	Tieto vaikeasti tai erilaisesti ymmärrettävissä Liian yksityiskohtainen dokumentaatio Dokumentteja ei koeta hyödyllisiksi Lähdekoodin puutteellinen ymmärrys
Tiedon johtaminen	Dokumentaation suunnitelmallisuuden puute Dokumentaation prosessin puutteellinen seuranta
Ketterän kehityksen periaatteet dokumentaatiossa	Dokumentaation tuottamisen kasaantuminen yksittäiselle, taidoiltaan heikoimmalle henkilölle (Resurssien ketteryys) Ajan varaaminen dokumentaation tuottamiselle Dokumentaatio korvataan epävirallisella keskustelulla Dokumentaatio ei pysy kehityksen mukana

Näillä tuloksilla tätä kirjallisuusosuutta lähdetään täydentämään empiirisellä osuudella. Siinä syvennytään laadullisen tutkimuksen metodein, haastattelun kautta näkevätkö tuotteenomistajat teknisen dokumentaation tuottamisessa samoja haasteita kuin edellä mainituissa tuloksissa esiteltiin. Tässä luvussa esitetty taulukko on esitetty myös tutkimustulosten yhteenvedossa (Luku 6.7) empiirisen osuuden tulosten näkökulmasta samojen teemojen alle. Haastatteluaineistolla täydennetään myös haasteiden listaa uusilla näkökulmilla. Dokumenttien sisällöstä halutaan selvittää, minkälaisia dokumentteja projektista tuotetaan ja mitä erilaisia haasteita eri dokumenttityypeillä on.

Empiirisen osuuden tavoitteena on selvittää miten tuotteenomistajat ratkaisevat edellä mainittuja haasteita käytännössä ja mikä niitä aiheuttaa. Tutkimuksen lähtökohtana on tarkastella, mitä tarpeita tuotteenomistajalla on teknisestä dokumentaatiosta ja miksi he kokevat tarvitsevansa erityyppisiä dokumentteja. Lisäksi halutaan selvittää, miten he johtavat dokumentaation prosessia työssään.

5 TUTKIMUSMENETELMÄ JA TUTKIMUKSEN KULKU

Tässä luvussa esitellään tämän pro gradu -tutkimuksen empiirisessä osassa käytetty tutkimusmenetelmä ja tutkimuksen suunnitelma sekä kuvaillaan tutkimuksen eteneminen. Kirjallisuusosuudesta löydettyjen aiheiden pohjalta muodostui lopullinen tutkimuskysymys, jonka selvittämiseksi toteutettiin empiirisen osuuden tapaustutkimus. Tässä luvussa esitellään tarkemmin empiirisen tutkimuksen tiedonkeruutapa ja siitä syntyneen aineiston analysointiprosessi.

5.1 Tutkimusmenetelmä

Tutkimuksessa käytettiin laadullista eli kvalitatiivista tutkimusmenetelmää. Tuomi ja Sarajärvi (2018) kutsuvat laadullista tutkimusta myös ymmärtäväksi tutkimukseksi, jonka tarkoitus on ymmärtää ilmiöitä. Hirsjärvi (2015, s. 22) täydentää tätä käsitystä täsmentämällä, että kvalitatiivinen ote näkee todellisuuden subjektiivisena sekä moninaisena, kuten tutkittavat todellisuutensa kokevat. Tutkija yrittää näin ollen laadullisen tutkimuksen metodein ymmärtää tutkittavan todellisuutta. Hirsjärvi (2015, s. 22) myös muistuttaa, että todellisuuksia on olemassa yhtä monta kuin sen kokijoita.

Tämä empiirinen osuus toteutettiin tapaustutkimuksen menetelmin. Tapaustutkimuksen tarkoituksena ei ole tuottaa ilmiöstä yleistyksiä, vaan pyritään selvittämään yksityiskohtaista tietoa tosielämän monimutkaisista tapahtumista. Tapauksesta halutaan tuottaa mahdollisimman intensiivinen ja tiheä kuvaus, sen tulkinta ja sitä kautta tapauksen ymmärtäminen. Eri tapaustutkimusten suuntauksista riippuen tavoitteena voi olla ilmiötä kuvaileva, selittävä, uutta teoriaa kehittävä tai olemassa olevaa teoriaa testaava tutkimus. (Eriksson & Koistinen, 2005, ss. 12-15.)

Tapaustutkimuksen kriittisin vaihe on tapauksen määrittely. Tämä voi tapahtua joko ennen aineiston keruuta tai sen jälkeen. (Eriksson & Koistinen, 2005, s. 6.) Tämän tutkimuksen tapaus määriteltiin teoriaosuuden yhteydessä ja se on

teknisen dokumentaation prosessi ja siinä ilmenevät haasteet ketterissä kehitysprojekteissa tuotteenomistajan näkökulmasta. Tapauksesta haluttiin ymmärtää, mihin dokumentaatiota tarvitaan ja miksi sen puutteellisuus tai puuttuminen on haastava tilanne tuotteenomistajille.

5.2 Tiedonkeruumenetelmä

Tutkimus suoritettiin puolistrukturoiduilla teemahaastatteluilla. Tiedonkeruumenetelmä valittiin sen perusteella, että tutkimustapaus tutkii kahta organisatorista prosessia, joita toteutetaan yhtäaikaisesti jopa toistensa kustannuksella. Kun puhutaan ilmiöistä, käytänteistä tai prosesseista, jotka ovat todella monimutkaisia, niitä ei pystytä selittämään kyselymuotoisilla tiedonkeruumenetelmillä (Eriksson & Koistinen, 2005, s. 12). Tällöin haastattelut tarjoavat avoimemman ja yksityiskohtaisemman tiedonkeruumenetelmän.

Kuten Hirsjärvi (2015, s. 34) mainitsee, haastatteluilla toteutetussa tutkimuksessa on mahdollista löytää vastausten takana piileviä motiiveja ja näin ymmärtää merkityksiä eri tavoin, kuin tutkimuksen alussa ajateltiin. Haastateltavalla on merkityksiä luova puoli tutkimuksessa. Aktiivisena osallistujana tutkimukseen haastateltavalta voidaan pyytää selvennyksiä vastauksiin sekä syventää annettua tietoa, kun jo ennalta tiedettiin, että aihe sisältää monitahoisia vastauksia. (Hirsjärvi & Hurme, 2015, s. 35.)

Tutkimushaastattelut voivat olla strukturoituja, puolistrukturoituja, teemahaastatteluja, syvähaastatteluja tai kvalitatiivisia haastatteluja (Hirsjärvi & Hurme, 2015, ss. 43–44). Teknisen dokumentaation käsitteen ollessa tarkasti määritelty kirjallisuuden kautta, mutta dokumentaation prosessin ollessa hyvin monialainen, päädyttiin tässä tutkimuksessa käyttämään puolistrukturoitua teemahaastattelua.

Puolistrukturoidussa haastattelussa kysymykset ovat kaikille samat, mutta kysymysten järjestys voi vaihdella ja haastateltavat voivat vastata kysymyksiin omin sanoin. Teemahaastattelu on puolistrukturoidun haastattelun muoto, jossa haastattelu kohdennetaan tiettyihin teemoihin. Tämä menetelmä keskittää keskustelun tiettyihin, tutkimuksen kannalta oleellisten teemojen ympärille. (Hirsjärvi & Hurme, 2015, ss. 47-48.)

Toisin kuin kvantitatiivisessa, kvalitatiivisessa tutkimuksessa tutkimuksen kohde ja tutkija nähdään olevan vuorovaikutuksessa keskenään. Tämä näkyy siinä, että haastattelija ja haastateltava luovat yhdessä tutkittavaa kohdetta. Vaikka haastattelu on pitkälle standardisoitu, vahvistaa haastattelija haastateltavaa vastauksissa. (Hirsjärvi & Hurme, 2015, s. 23.) Haastattelukysymykset rakennettiin kirjallisuusosuudesta löytyneiden kehitettyjen teemojen alle. Kysymykset kirjoitettiin valmiiksi, ja ne pyrittiin esittämään mahdollisimman yhtenäisesti jokaiselle haastateltavalle. Kuitenkin kysymysten järjestys saattoi vaihdella, jos keskusteltiin syvemmin tietyn teeman alla olevasta kokonaisuudesta. Haastatteluosuus pohjautui vahvasti kirjallisuusosuuden tuottamaan olettamusmalliin, jossa haastatteluiden tuottamalla tiedolla yritettiin vahvistaa olettamukset tai todistaa ne vääräksi.

5.3 Aineiston analysointimenetelmä

Haastatteluilla kerättyä aineistoa voi analysoida kolmella eri tavalla. Tutkijan intuitioon luottaen siirtyen suoraan analyysiin, tai vaihtoehtoisesti purkamalla ja koodaamalla aineisto joko samanaikaisesti tai erottaen purkamisen ja koodaamisen vaiheet, jonka jälkeen siirrytään vasta analyysiin. (Hirsjärvi & Hurme, 2015, s. 136.) Tässä tutkimuksessa edettiin analyysiin tutkijan intuitioon luottaen. Tähän päädyttiin siksi, koska tutkija omaa samat taustat kuin haastateltavat. Haastateltavien taustat on esitelty alaluvussa 5.5.

Analyysi noudatti tapaustutkimuksen pääpirteitä. Analyysi aloitettiin jo haastatteluvaiheessa, jolloin tutkija muodosti alustavia hypoteeseja ja malleja syntyneestä materiaalista. Tämän jälkeen aineisto litterointiin. Litterointi toteutettiin, koska haastattelut olivat pitkiä (30-60 minuuttia), jolloin nauhoitteesta suoraan purkaminen ei ollut mahdollista. Litteroinnin yhteydessä aineistoa selkeytettiin poistamalla turhat toistot ja ylimääräiset osiot. (Hirsjärvi & Hurme, 2015, ss. 136-152.) Yksityiskohtainen analyysin kulku on esitelty luvussa 5.6. *Tutkimuksen kulku*.

Aineiston analyysi toteutettiin lähellä aineistoa, jolloin täsmälliset sanamuodot tulevat käsitellyksi. Hirsjärvi ja Hurme (2015, s. 136) esittävät että tutkijan tulee valita laadulliseen tutkimukseen joko induktiivinen (yksittäisestä yleiseen) tai abduktiivinen (havainnoista selitykseen) päättelytapa. Tuomi ja Sarajärvi (2018) lisäävät vielä deduktiivisen (yleisestä yksittäiseen) päättelytavan laadullisen tutkimuksen analyysin läpikäymiseen. Tämän tutkimuksen tutkimuskysymystä palveli abduktiivinen lähestymistapa, koska tässä tutkimuksessa ei pyritty yleistämään tai vetämään johtopäätöksiä yksilöön liittyen, vaan ymmärtämään kokonaista ilmiötä.

Lopuksi ainestoa lähdetiin analysoimaan esitetyn litteroitujen aineistojen tulkinnan kautta. Analyysin aikana aineistoa tiivistettiin, ryhmiteltiin ja luokiteltiin aiemmin havaittujen teemojen alle. Siitä etsittiin narratiiveja haastateltavien kertomusten perusteella (Hirsjärvi & Hurme, 2015, s. 137).

5.4 Tutkimuksen rajaukset

Tutkimuksen kirjallisuusosuus toimi koko empiirisen osuuden rajaustekijänä. Kirjallisuudesta löydettyjen teemojen perusteella voitiin rajata tietyt dokumentaation prosessit, jotka toistuvat aina, kun dokumentaatiota esiintyy organisaatioissa. Lisäksi tunnistettiin ketterän kehityksen tunnuspiirteet, jotka toistuvat aina ketterin metodein toteutetussa järjestelmäkehityksessä. Näiden prosessien yhtäaikainen toteuttaminen määritti tämän tutkimuksen tapauksen.

Lisäksi molempien prosessien tehokas yhtäaikainen toteuttaminen paljasti useita haasteita olemassa olevissa tutkimuksissa. Yleisimmät haasteet tunnistettiin ja jaoteltiin dokumentaation prosessien, sen johtamisen ja ketterien metodien alle. Tunnistettujen teemojen ja haasteiden pohjalta muodostettiin teemoitettu puolistrukturoitu haastattelurunko.

Tapaustutkimuksessa tutkittiin niitä kirjallisuudessa esitettyjä haasteita dokumentaation prosessissa ketterässä kehityksessä, joita tuotteenomistaja kohtaa päivittäisessä työssään. Siinä haluttiin tunnistaa, mikä aiheuttaa näiden yleisten ongelmien jatkumisen organisaatioiden sisällä ja miksi ne ovat hankala ratkaista. Tutkimuksessa ei tutkittu, millä konkreettisin tavoin tai mihin kanaviin tekninen dokumentaatio tuotetaan. Tutkimuksessa ei myöskään haluttu luoda uutta artefaktia, joka määrittäisi millä tavoin teknistä dokumentaatiota voitaisiin tuottaa käytännössä.

5.5 Haastateltavien tausta ja valinta

Kvalitatiivisessa tutkimuksessa tutkimusyksiköt valitaan tarkoituksenmukaisesti. Koska laadullisen tutkimuksen tavoitteena on pyrkiä ymmärtämään ilmiötä tilastollisen yleistämisen sijaan, haastateltavien tulisi ymmärtää tutkittavaa aihetta syvällisemmin. Näin he voivat tarjota syvällisempää tietoa tutkittavasta aiheesta. (Hirsjärvi & Hurme, 2015, s. 59.) Tähän tutkimukseen valittiin haastateltavat Scrum-viitekehyksen roolituksen ja työtehtävien vastuullisuuden mukaan.

Tutkimukseen valittiin IT-alan asiantuntijoita, jotka olivat toimineet useissa eri rooleissa, tehtävissä ja yrityksissä järjestelmäkehityksen parissa. Oletettiin, että he ovat tuottaneet tai kuluttaneet teknisen dokumentaation käsitteeseen kuuluvia dokumentteja. Kaikki haastateltavat olivat myös toteuttaneet järjestelmäkehitystä ketterien metodien mukaisesti. Näin ollen molemmat tapaustutkimukseen kuuluvat prosessit olivat heille työn kautta tuttuja.

Teknisen dokumentaation sisältäessä kehittäjien, arkkitehtien ja tuotteenomistajan tuottamaa dokumentaatiota, haastateltavien taustat valittiin niin, että haastateltavista löytyi jokaisen roolin edustajia, nykyisessä tai edellisissä työtehtävissään. Lisäksi haluttiin, että haasteltavat olivat toteuttaneet tuotteenomistajan rooliin kuuluvia työtehtäviä, jotta heillä olisi kokonaisvaltainen käsitys järjestelmäkehityksen johtamisesta. Kohdejoukon rajauksella kaikkiin dokumenttityyppeihin liittyviin kysymyksiin saatiin syvällisiä vastauksia. Tuotteenomistajan tehtävien ymmärtäminen auttoi haasteltavia tunnistamaan dokumentaation johtamiseen liittyviä haasteita. Lisäksi he pystyivät myös tunnistamaan eri dokumenttityyppien tuottamisen tai päivittämisen vastuualueita kehitystiimin sisällä. Haastateltavien taustatiedoiksi kysyttiin ikä, nykyinen titteli, työkokemus tuotteenomistajan työtehtävissä vuosina sekä koodin tuottamisen osaaminen. Taustatiedot on esitetty seuraavassa taulukossa (taulukko 2.)

TAULUKKO 2. Haastateltavien taustatiedot

	H1	H2	H3	H4	H5	H6
Ikä	28	29	40	41	38	46
Nykyinen titteli	IT Specialist	Senior Consultant	Web & Digital business Expert	Järjestelmä vastaava	Senior Manager	Senior Manager
Työkokemus vuosina	3,5	6	12	5	7	3
Koodin tuottaminen	Alkeet	Ammatillinen osaaminen	Ei osaamista	Ei osaamista	Ammatillinen osaaminen	Alkeet

Haastateltavien taustiedoista käy ilmi, että kaikilla haastateltavilla on vähintään kolmen vuoden kokemus tuotteenomistajan työtehtävistä. Lisäksi haluttiin karkeasti eritellä haastateltavien osaaminen koodin tuottamisesta, koska se koettiin osan teknisen dokumentaation käsitteen alle kuuluvien dokumenttien ymmärryksen kannalta oleelliseksi. Ikä kysyttiin vain, mikäli vastauksissa olisi löytynyt suuria eroavaisuuksia, olisi niitä voitu yrittää ymmärtää haastateltavien ikäerosta johtuvista tai työuran pituuteen liittyvistä tekijöistä. Haastateltavien tittelillä haluttiin osoittaa, että jokainen haastateltava toimii edelleen IT-alan tehtävissä, ja että tuotteenomistajan työtehtäviä voidaan suorittaa eri tittelien alla.

5.6 Tutkimuksen kulku

Tutkimuksen alustava ongelma tunnistettiin kirjallisuuskatsauksen pohjalta. Kirjallisuusosuudessa havaittiin dokumentaation perusprosessit sekä ketterän kehityksen tuottamat haasteet dokumentaation prosessin eri vaiheissa. Dokumentaation prosessin sekä siinä havaittujen haasteiden pohjalta muodostettiin kysymykset haastatteluihin.

Haastattelut toteutettiin yksilöhaastatteluina kesä-heinäkuussa 2019. Haastattelut toteutettiin haastateltavien työpaikoilla sekä kaupungin kirjaston neuvotteluhuoneissa. Kysymykset esitettiin haastateltaville kysymysrungon pohjalta (liite 1). Ensimmäisenä haastateltavalle annettiin monivalintapohja (liite 2), jonka yläosaan haastateltavaa pyydettiin täyttämään taustatietonsa. Ensimmäisenä varsinaisena kysymyksenä kysyttiin kaikista dokumenteista, joita he olivat käyttäneet tai tuottaneet työssään. Tähän kysymykseen haastateltavat käyttivät monivalintapohjan esimerkkejä. Monivalintapohjaan oli koostettu kirjallisuusosuudessa (luku 4.3.2.) ilmi tulleet, tekniseksi dokumentaatioksi nimetyt dokumentit. Monivalintaosuuden yhteydessä osa haastateltavista vastasi

myös kysymykseen minkälainen dokumentti on kyseessä ja millä tavalla asia on dokumentoitu.

Ensimmäisen osion kaikki kysymykset koskivat dokumentaation tuottamista. Tässä kohdassa tutkimuksen kannalta oleellisten dokumenttien tuottamisen prosessi kysyttiin vielä erikseen. Toisen osion kysymykset koskivat tiedon kehitystä ja päivitystä. Näissä kysymyksissä keskityttiin vain haastateltavan osiossa yksi mainitsemiin dokumentteihin. Kolmannessa osiossa käytiin läpi tiedon jakamisen prosessia. Kysymyksissä ei haluttu paneutua tiedon jakamiseen kanaviin, vaan siihen kenellä on pääsy eri dokumentteihin. Lisäksi haluttiin selvittää, onko tieto helposti löydettävissä sekä ymmärrettävissä. Neljännessä osiossa keskityttiin tiedon käyttöön, jossa pääfokus oli tunnistaa dokumenttien tuottamaa hyötyä käytännön työtehtäviin. Viidennessä osiossa selvitettiin dokumentaation johtamista. Kysymyksillä yritettiin tunnistaa vastuualueita dokumentaation prosessissa. Viimeisessä osiossa selvitettiin, onko ketterien metodien käytäntöjä omaksuttu dokumentaation prosessiin. Mikäli haastateltava vastasi johonkin kysymykseen lyhyesti, vain myöntävästi tai kieltävästi, haastateltavaa kannustettiin haastattelijan puolelta perustelemaan käyttämällä 'miksi' ja 'miten' -kysymyksiä.

Haastattelut kestivät puolesta tunnista tuntiin, riippuen siitä kuinka useaan kysymykseen kukin haastateltava osasi vastata sekä kuinka kiinnostunut haastateltava oli dokumentaation prosessista. Haastatteluja toteutettiin yhteensä kuusi kappaletta. Haastattelut nauhoitettiin puhelimen sanelin-toiminnolla ja litterointiin myöhemmin nauhoitteesta. Litteroinnin yhteydessä poistettiin haastateltavien täytesanat sekä korvattiin mahdolliset yritysten ja ihmisten nimet asiayhteyteen sopivalla tavalla, niin ettei aineistosta käy ilmi, mikä yritys tai kuka henkilö on kyseessä. Ihmisten nimien sijaan litteroitiin mainitun henkilön titteli kyseisessä organisaatiossa. Kaikki mainitut yritykset olivat joko haastateltavan nykyinen tai entinen yritys, joten kaikki viittaukset yrityksiin litterointiin muotoon "edellinen työnantaja" sekä "nykyinen työnantaja".

Litteroinnin jälkeen aineisto analysointiin. Analyysi toteutettiin Hirsjärven ja Hurmeen (2015, s. 144) esittelemien analyysin vaiheiden mukaan. Ensimmäinen vaihe oli analyysi, joka eteni kokonaisuudesta osiin, aineiston luokitteluun ja luokkien yhdistelyyn. Toinen vaihe oli synteesi, jossa edettiin takaisin kokonaisuuteen, tulkintaan ja ilmiön uudelleen hahmottamiseen. Haastattelun ollessa puolistrukturoitu teemahaastattelu jokaisen haastateltavan vastaukset eri kysymyksiin koostettiin teemoittain eri tiedostoihin. Näin kaikki vastaukset samaan kysymykseen saatiin samaan tiedostoon vastausten koostamista ja vertailua varten. Luokittelulla erillisistä vastuksista pystyi hahmottamaan yhteyksiä vastausten välillä. Aineiston luokittelun ja yhdistelyn jälkeen pyrittiin muodostamaan kokonaisuuksia vastauksista jokaisen kysymyksen alle. Nämä kokonaisuudet on raportoitu luvussa kuusi *Tutkimustulokset.* Tutkimustulosten koonnin jälkeen tuloksista koostettiin johtopäätökset, jotka raportoitiin lukuun seitsemän *Tulokset ja johtopäätökset.*

5.7 Tutkimuksen luotettavuus

Tutkimuksen luotettavuutta arvioidaan reliabiliteetin ja validiteetin kautta. Reliaabeliudella tarkoitetaan sitä, että tutkimus voidaan toistaa ja saada sama tulos (Hirsjärvi & Hurme, 2015, s. 186). Tutkimuksen toistettavuutta tukee tutkimuksen yksityiskohtainen kuvaus luvussa viisi (luku 5.6). Lisäksi tutkimuksessa käytetty haastattelurunko on liitetty tähän työhön (liite 1) sekä tutkimuksessa käytetty monivalintapohja (liite 2). Haastattelukysymyksiin saatiin toistaan tukevia vastauksia ja niistä löytyi paljon yhteneväisyyksiä, joka viittaa hyvään reliabiliteettiin.

Validiteetilla tarkoitetaan sisäistä ja ulkoista validiteettia. Sisäinen tarkoittaa, että tutkimuksessa tutkittiin sitä mitä oli tarkoituskin tutkia. (Hirsjärvi & Hurme, 2015, ss. 187-188.) Tässä tutkimuksessa oli tarkoitus tutkia haasteita teknisen dokumentaation tuottamisessa. Aiheeseen perehdyttiin olemassa olevan kirjallisuuden perusteella. Jolloin etsittiin jo tunnistettuja haasteita teknisen dokumentaation tuottamisessa. Kirjallisuuden perusteella muodostettiin teemat, joiden alle haasteet jaoteltiin. Tutkimusainestoa rikastettiin teemahaastattelulla, jossa keskityttiin samoihin teknisen dokumentaation haasteisiin aiemmin tunnistettujen teemojen alla. Näin tutkimuksessa saavutettiin sisäinen validiteetti.

Ulkoinen validiteetti tarkoittaa tulosten yleistettävyyttä. Yleistettävyydellä taas tarkoitetaan, että tutkimus voidaan toistaa toisilla henkilöillä, toisessa tilanteessa ja saada samat tulokset. Tutkimushaastattelun tutkimusyksikön ollessa ihminen on oletettavaa, että muutosta tapahtuu ajan kuluessa. (Hirsjärvi & Hurme, 2015, ss. 186, 188.) Tässä tutkimuksessa hyväksytään, se että tulos on riippuvainen ajasta ja paikasta sekä siitä, että ihminen on muuttuva, eikä näin tutkimuksessa odoteta tulosten yleistettävyyttä. Myös haastateltavien tiukka rajaaminen tiettyihin työtehtäviin pienensi tutkimusyksikköjen lukumäärää, joten tuloksien yleistettävyys ei ole tämänkään puolesta odotettavaa.

5.8 Yhteenveto

Tämä luku kuvaili tutkimuksen kulun sekä esitteli siinä käytetyt tutkimusmetodit. Luvussa esiteltiin tapaustutkimuksen tutkimus-, tiedonkeruu- ja analyysimenetelmät. Tutkimuksen kulku esiteltiin alusta loppuun, kirjallisuuden hyödyntämisestä, haastatteluihin, aineiston analyysiin ja lopuksi tuloksien esiin tuomiseen. Kuvailun tarkoituksena oli auttaa ymmärtämään, miten tutkija on päätynyt lopputuloksiin sekä helpottaa tutkimuksen toistettavuutta. Tutkimuksen kulun kuvailun lisäksi osoitettiin tutkimuksen rajaukset, perusteltiin haastateltavien taustat ja valinnan kriteerit sekä pohdittiin tutkimuksen luotettavuutta reliabiliteetin ja validiteetin kautta. Seuraavassa luvussa esitellään tutkimuksen tulokset teemoittain.

6 TUTKIMUSTULOKSET

Tässä luvussa esitellään tutkimustulokset, jotka saatiin empiirisen osuuden materiaalin analysoinnista ja kirjallisuuskatsauksen löydöksistä. Empiirisen osuuden tapaustutkimuksella oli tarkoitus täydentää kirjallisuusosuuden tuloksia. Tutkimuskysymykseen *"Mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?"* haluttiin löytää syventäviä vastauksia. Haasteiden taustalta haluttiin löytää syitä, jotka aiheuttavat haasteet ketterissä kehitysprojekteissa. Syiden tunnistamisen jälkeen voidaan niihin löytää ratkaisuja. Tutkimustulokset esitellään samassa järjestyksessä, kuin ne olivat kerätty kirjallisuusosuuden yhteenvedossa, jotta tuloksia on helpompi vertailla keskenään.

Tulosten esittely noudattaa tutkimuksessa esiteltyjen teemojen rakennetta. Ensimmäiset neljä kappaletta keskittyvät dokumentaation prosessin vaiheisiin; tiedon tallennus, tiedon kehitys ja päivitys, tiedon jakaminen ja tiedon käyttö. Viides teema käsittelee tiedon johtamista. Viimeinen kuudes teema keskittyy ketterien periaatteiden omaksumiseen dokumentaation prosessiin.

6.1 Tiedon tallennus

Ensimmäisten kysymysten teema oli tiedon tallentaminen. Tämän teeman kysymyksissä haluttiin selvittää, minkälaisia teknisiä dokumentteja järjestelmäkehitysprojekteissa luodaan ja millä tavoin ne on luotu. Haastateltavien ensimmäisenä kysymyksenä oli *minkälaista dokumentaatiota järjestelmäkehityksestä ja järjestelmän teknisestä toteutuksesta tuotat ja kulutat projekteissasi?* Tämän kysymyksen avuksi käytettiin monivalintapohjaa. Kysymyksillä haluttiin selvittää, tuotetaanko ketterissä järjestelmäkehitysprojekteissa niitä dokumentteja, joita kirjallisuudessa mainittiin teknisiksi dokumenteiksi. Osaksi haluttiin myös rajata pois tarkentavat kysymykset tietyistä dokumenteista, jos haastateltava ei ole ko. dokumentin kanssa ollut tekemisissä. Kuitenkin osa haastateltavista valitsi kaikki dokumentit, joita järjestelmäkehitysprojekteissaan tiesivät olevan, mutta eivät tarkentavien kysymysten vaiheessa osanneet kertoa niistä syvällisemmin.

Seuraavassa taulukossa on esitelty, mitä dokumentteja haastateltavat valitsivat (taulukko 3), lajiteltuna vastausten määrän mukaan laskevasti. Monivalintapohjassa mainittujen dokumenttien lisäksi haastateltavat mainitsivat kaksi muuta teknisen dokumentaation muotoa: Yksityiskohtainen testikattavuuden seuranta (H1) ja Domain Event Dokumentaatio (H2). Nämä listattiin taulukkoon omille riveilleen.

TAULUKKO 3 Tekniset dokumentit, joita tuotettiin tai käytettiin projekteissa

	H1	H2	H3	H4	H5	H6	Yht.
Järjestelmän kuvaus	X	X	X	X	X	X	6
Käyttöliittymäsuunnitelmat	X	X	X	X	X	X	6
Käyttöliittymien määritykset	X	X	X	X	X	X	6
Vaatimusmäärittelyt	X	X	X	X	X	X	6
Käyttäjätarinat (User Story)	X	X	X	X	X	X	6
Käyttöskenaariot (Scenarios)	X	X	X	X	X	X	6
Kehityksen aikana tehdyt päätökset järjestelmään ja sen toimintoihin liittyen	X	X	X	X	X	X	6
Järjestelmän tukemien liiketoimintaprosessien kuvaukset	X	X	X	X	X	X	6
Testaussuunnitelma	X	X	X	X	X	X	6
Testitapaukset	X	X	X	X	X	X	6
Kokonaisarkkitehtuurin kuvaus	X	X	X		X	X	5
Sprintin edistymiskäyrä (Burn down chart)	X	X	X	X	X		5
Testausraportti	X	X	X		X	X	5
Käyttäjäryhmien määrittely			X	X	X	X	4
Tietokannan määrittely	X	X	X		X		4
Sprintin suunnitteludokumentti (Sprint planning)	X		X	X	X		4
Rajapinnan määrittely		X	X		X		3
Datamalli	X	X	X				3
Sprintin arviointidokumentti (Sprint Review)	X		X		X		3
Koodiin upotetut kommentit		X	X		X		3
Osa-arkkitehtuurin kuvaus (esim. mikropalveluiden arkkitehtuuri)		X	X		X		3
Arkkitehtuurin rakenteeseen liittyvät päätökset		X	X		X		3
Testimenetelmät		X	X			X	3
Luokkakuvaukset		X	X				2
Algoritmien kuvaukset			X				1
Muita: Yksityiskohtainen testikattavuuden seuranta	X						1
Muita: Domain Event Dokumentaatio		X					1

Tarkentavat kysymykset dokumenttityypeistä koskivat järjestelmävaatimuksia, kokonaisarkkitehtuuria, koodin kommentteja, testauksen dokumentaatiota sekä järjestelmän sisäisiä prosesseja. Nämä oli valittu tutkimuksen keskeisemmiksi dokumenttityypeiksi kirjallisuusosuudessa.

Vaatimusmääritykset kattavat tässä yhteydessä vaatimusmäärittelyt, käyttäjätarinat ja skenaariot. Haastateltavilta kysyttiin, *miten vaatimusmäärittelyjen tekniset tiedot ts. järjestelmävaatimukset dokumentoidaan?* Kaikki vastaajat olivat

dokumentoineet järjestelmävaatimuksia User Story -mallilla. Vaatimuksia kirjoitettiin myös yhteenvetona erilaisiin dokumentaatiotyökaluihin.

Käytännön vaatimusten lisäksi kysyttiin, *dokumentoidaanko ei-toiminnallisia järjestelmävaatimuksia?* Osalle termi ei ollut tuttu, joten haastattelija antoi tähän yleisimmät esimerkit, nopeus ja turvallisuus. Osaksi tämän vuoksi saatiin vastauksia vain nopeuteen ja turvallisuuteen liittyen. Kaikki haastateltavat olivat joskus joutuneet määrittelemään ei-toiminnallisia vaatimuksia, mutta puolet vastaajista (H1, H2, H4) eivät olleet koskaan antanut numeerisia tai mitattavia reunaehtoja järjestelmälle. H5:sen organisaatiossa oli ei-toiminnalliset vaatimukset tarkimmin määritelty, jonka hän summaa:

> Kaikki [ei-toiminnalliset vaatimukset] on tehty niin, että ne ovat mitattavia. Eli ihan sekuntimääriä, prosenttilukuja, kellonaikoja ja muita asioita, mutta niinhän ne taitavat olla, että ne pitää pystyä mittaamaan.

Arkkitehtuurin dokumentaatiosta kysyttiin yleisesti, *onko järjestelmän kokonaisarkkitehtuurista tai sen osista dokumentaatiota* yhdistäen molemmat monivalinnassa olevat arkkitehtuuriin liittyvät dokumentit. Arkkitehtuurin kuvaukset ovat yleensä toteutettu muuten kuin luonnollisella kielellä, joten haastateltavilta kysyttiin, *miten kuvaus on toteutettu?* H5 kertoi olevansa osa arkkitehtuurilautakuntaa, jossa kokonaisarkkitehtuurin ja integraatioiden kuvaukseen käytetään *ArchTech* -työkalua. Mikropalvelujen dokumentit olivat kuitenkin toteutettu epäformaalein tavoin. Muut vastaajat olivat tuottaneet tai kuluttaneet arkkitehtuurin kuvauksia pelkästään epäformaalein graafisin kuvauksin.

Ketterien metodien mukaisesti arkkitehtuuri rakentuu ja täydentyy kehityksen edetessä (Isham, 2008, s. 488). Tämän vuoksi haastateltavilta kysyttiin vielä, *onko arkkitehtuuri kehittynyt tuotteen mukana vai määritelty etukäteen?* Tässä kysymyksessä ei haluttu paneutua ketterän arkkitehtuurin metodeihin, vaan todistaa että jatkuvasti muuttuvan arkkitehtuurin dokumentaatiota täytyy jatkuvasti päivittää. Yleisesti kaikki haastateltavat totesivat, että järjestelmien arkkitehtuuri on elänyt tuotteen mukana. Vain H4 koki, ettei arkkitehtuurille ollut luotu mitään suunnitelmaa etukäteen. Muut haastateltavat kertoivat, että arkkitehtuurista on luotu alustava luonnos ennen kehityksen aloittamista. Tähän luonnokseen on voinut nojautua päätöksissä, mutta se on aina mukautunut kehityksen aikana (H2).

Vain ammatillisen osaamisen koodin tuottamisessa omaavat H2 ja H5 pystyivät vastaamaan syvällisemmin kysymykseen, *käytetäänkö koodiin upotettuja kommentteja dokumentaationa?* H5 koki, että koodin kommentit ovat IT-tukifunktioissa avainasemassa virheenjäljityksen yhteydessä. Etenkin Legacy-järjestelmän kanssa tekemisissä olevissa tukifunktioissa ei koodia ole kommentoitu, joka on johtanut siihen, että on menetetty hiljainen tieto tekijän mukana. H2 tiivisti oman tapansa koodin kommenttien tuottamiseen tehokkaasti:

> Mitä tapaa olen itse suosinut ja miten yleensä projekteissani on ollut [käytössä], on ollut, että vain sellaiset outoudet on yritetty dokumentoida kommentteihin. Eli esim. joku päätös, että miksi näin on tehty. Yleensä yritetään olla dokumentoimatta kommentoimalla "Mitä", vaan enemmän "Miksi". On pyritty siihen, että koodi itsessään on dokumentaatio "Mitä".

Testauksen dokumentteja oli annettu useampi vaihtoehto haastateltaville, joten heiltä kysyttiin, *tuotetaanko testauksesta dokumentaatiota, jos kyllä; miten?* Haastateltavat saivat vapaasti kertoa, miten eri testauksen dokumentteja tuotetaan. Testauksen dokumentaatiosta löytyi suuria eroja haastateltavien välillä. Testaussuunnitelman vain H5 tiesi pohjautuvan IEEE-standardiin. Tuotteenomistajan luoman testaussuunnitelman tarkoituksena on resursoida ja priorisoida testitapaukset (H1). Tarkoituksena on saada varmuus, milloin on testattu tarpeeksi, jotta hyväksytty järjestelmä vastaa odotuksia (H2). Testitapaukset luotiin kaikilla haastateltavilla käyttäjätarina- ja skenaariomallin mukaisesti, jolloin testitapaukset tuotettiin kaikki käyttäjätarinan perusteella. Testitapauksien dokumentoinnilla haettiin tehokkuutta; varmistettiin ettei testata turhaan useasti samaa tapausta sekä myös, että kaikki tapaukset tulevat varmasti testattua (H4).

Testauksesta tuotetuista raporteista haluttiin lähtökohtaisesti nähdä paljonko testitapaukset kattavat koodia ja kuinka moni testitapaus ajettiin onnistuneesti läpi (H2). H1 koki myös, että ulkoistetusta manuaalitestauksesta tuotetaan jopa tarpeettoman yksityiskohtaista raportointia. Testauksesta raportoitiin hänelle kattavuuden ja lopputulosten lisäksi myös testaajien tehokkuudesta kertovia lukuja. Kaikki vastaajat olivat olleet tekemisissä automaatiotestauksen kanssa. Automaation suurimpana hyötynä on, että saadaan aina ajan tasaisia testausraportteja, joita luettiin eri testiraportointityökaluista.

Järjestelmän prosessien kuvauksista selvitettiin, millä eri tavoin niitä on dokumentoitu. Esitetty kysymys oli, *onko järjestelmän sisäiset prosessit ts. järjestelmän kannalta oleelliset tai järjestelmän tukemat liiketoiminnan prosessit dokumentoitu, jos kyllä; miten*? Yleisesti voi sanoa, että kukaan vastaajista ei ollut tyytyväinen prosessien dokumentaatioon. Suurin ongelma oli, että sitä oli aivan liian vähän, ne eivät ole ajan tasalla ja niitä ei ole kootusti saatavilla (H1, H2, H5, H6). Jos prosessikuvia löytyy, ne ovat ylätason kaavioita (H6). Puolet vastaajista mainitsi, että prosessien tarkat kuvaukset löytyvät vain koodista (H1, H2, H5). Prosessikuvista tiesi tarkasti H6, joka työskentelee läheisesti liiketoiminnan edustajien kanssa.

> … meillä on aika hajanaisesti niitä [järjestelmän prosessikuvauksia]. Toisaalta business tekee omia Flow Chartejaan ja IT on tehnyt omiaan, sitten meiltä puuttuu se, joka vetäisi ihan kaiken yhteen. Että siellä olisi sekä asiakas, että meidän sisäiset käyttäjät, että prosessi, että järjestelmät. Näitä ei ikävä kyllä ole olemassa.

Kaikista dokumenteista, jotka haastateltavat olivat valinneet monivalintaan, haluttiin selvittää, *miksi juuri tällaisia dokumentteja on päätetty tuottaa*? Kysymyksellä haettiin dokumenttien tuottamaa arvoa organisaatiolle. Hyötyjä löydettiin useasta teemasta. Nämä olivat kehityksen seuranta (H1, H4), toiminnan ohjaus (H1, H3), perehdytys ja koulutus (H1, H2, H5), järjestelmän historiatieto (H4, H6), kommunikointi (H2, H3, H4, H6). Kehityksen seurannan dokumenteilla voitiin arvioida tavoitteiden saavuttamista ja arvioida toteutusta. H4 löysi Spiritin suunnittelun, seurannan ja arvioinnin dokumenteista seuraavia hyötyjä:

> Se on enemmän sellaista pidemmän tähtäimen projektijohtamista, että pystytään suunnittelemaan jatkosprintit järkevästi. Siinä tapahtuu oppimista kaikilla osapuolilla. Siinä suurin hyöty.

Toimintaa ohjaavilla dokumenteilla voitiin resursoida ja priorisoida työtä, ja näin saavuttaa tehokkuutta. Turhat työvaiheet haluttiin poistaa ja näin välttää myös tuplatyötä. (H1.) Perehdytyksen mainitsi puolet haastateltavista tämän kysymyksen kohdalla. Dokumentaatio nopeuttaa uuden työntekijän perehdytysprosessia ja vähentää kouluttajan työtaakkaa, kun kysymyksiin löytyy vastaus dokumenteista. H5 tiivisti ajatuksen perehdyttämisestä:

> …kun 13 vuotta on ollut koodimaailmassa niin huomaa, että dokumentaation merkitys kasvaa, kun tiimin tekijät vaihtuvat ja sen kun tulee kysymyksiä, että miten joku asia toimii niin dokumentaatio nopeuttaa huomattavasti verrattuna siihen koodista kahlaamiseen.

Järjestelmän historiatiedon dokumentoinnilla saavutettiin jatkokehityksessä hyötyä, kun aiemmat ratkaisut ja päätökset niiden takana on tiedossa. Kommunikointiin kuului niin kehitystiimin sisäinen kuin ulkoinenkin kommunikointi (H3, H4). Dokumentaatiolla varmistetaan, että kaikilla osallisilla on sama kuva järjestelmästä. Sidosryhmille dokumentteja käytettiin kommunikointiin kehityksen tilanteesta, laadusta sekä siivittämään päätöksentekoa. (H6.) Näiden lisäksi H2 vielä mainitsi, että on mukana projektissa, jossa dokumentaatio on osa myytävä tuotetta. IT-ratkaisua ei voi myydä tai käyttää ilman kattavaa teknistä rajapintadokumentaatiota.

Vastuualueista kysyttiin, *mitä dokumentteja haastateltavat itse tuottavat ja mitkä he kokevat olevan muiden kehitystiimin jäsenten tai organisaation edustajien vastuulla*. Tällä hetkellä pääosin tuotteenomistajan roolissa toimivat kokivat tuottavansa tai olevansa vastuussa suurimmasta osasta dokumenteista (H1, H3, H4). Kokonaisarkkitehtuuriin kuuluvat dokumentit osoitettiin arkkitehdeille. Koodin kommentit, osa-arkkitehtuuri, tieto- ja datamallit sekä tietokannankuvaukset osoitettiin kehittäjille. Testauksen dokumentit osoitettiin testauksen tai laadunhallinnan tiimille. H2 oli ainoa, joka toimii täysin tasavertaisessa tiimissä, ja koki että kaikki tiimin jäsenet tuottavat tasavertaisesti dokumentaatiota. Tuotteenomistajan vastuusta H5 mainitsi, että tuotteenomistaja huolehtii, että kaikki dokumentit tulee tuotettua, mutta ei ole yksin vastuussa niiden tuottamisesta.

Teoriaosuudessa mainittiin, että tekninen dokumentaatio tuottaa eniten arvoa ylläpitotoimiin sekä jatkokehitystehtävien kehitystiimille. Tämän vuoksi haastateltavilta kysyttiin, *mitkä dokumentit koet tärkeimmiksi ylläpitoa ja jatkokehitystä ajatellen?* Haasteltavien ajatusta haluttiin vahvistaa mainitsemalla, että oletetaan että em. tehtäviä tulee hoitamaan täysin uusi kehitystiimi, joka poistaisi kaiken hiljaisen tiedon tiimistä. Haastateltavien listattua mielestään tärkeimmät dokumentit kysyttiin, *miksi he kokevat valitsemansa dokumentit tärkeimmiksi*. Vastaukset olivat yhtenäisiä kahden asian suhteen; valitut dokumentit kuvaavat joko järjestelmän nykytilaa (H1, H2, H3, H5, H6) tai historiaa (H1, H2, H3, H4). Historiatiedossa korostui, mitkä päätökset vaikuttivat nykyisen tekniseen toteutuksen syntymiseen (H4).

Samaan aiheeseen liittyen kysyttiin vielä, *tuotetaanko ylläpitoa ja jatkokehitystä varten vielä erikseen dokumentaatiota*. Osalle haastateltavista, jotka takeltelivat kysymyksen kanssa tai kielsivät yhdellä sanalla tällaisen toiminnan, esitettiin vielä lisäkysymys, *oletko itse kirjoittanut jotain perehdytys- tai ohjeteksti - materiaalia, korvaavalle työntekijälle kun olet lähtenyt projektista.* Tähän viisi kuudesta haastateltavasta pystyi samaistumaan. Täysin uutta dokumenttia tuotettiin vain, mikäli jokin kehitys oli jäänyt täysin kesken, jolloin paljon hiljaista tietoa - muistioita, sähköpostikeskusteluja täytyi kerätä käsillä olevaan asiaan liittyen (H3, H4). Tällaisia tilanteita oli esimerkiksi uudet toiminnot, jotka oli jo suunniteltu nykytilanteen päälle. Lisäksi H4 kertoi käyneensä uuden tekijän kanssa läpi kaikki kanavat, mistä eri dokumentteja löytyy. Näiden ohella haastateltavat olivat vain päivittäneet jo olemassa olevia kuvauksia (H1, H2, H3).

6.2 Tiedon kehitys ja päivittäminen

Seuraava teema oli tiedon kehittämisen ja päivityksen prosessit. Haastateltavilta kysyttiin *mitä dokumentteja on päivitetty edes kerran sen luomisen jälkeen, ja mitä ei ole koskaan päivitetty.* Kaikki haastateltavat sanoivat, että kaikkia dokumentteja on päivitetty edes kerran luomisen jälkeen, jos dokumentti oli olemassa. H1 kyseenalaisti, että onko dokumentti kuitenkaan ajan tasalla, vaikka sitä on päivitetty. H2 mainitsi, että kehityksen seurantaan ja toimintaa ohjaavat dokumentit ovat 'pois heitettävää' (throw-away) kertakäyttöistä dokumentaatiota, jota ei sen käsittelemisen jälkeen ole tarkoituskaan päivittää. Tällaisia ovat esimerkiksi jo kehityssprintin läpikäyneet käyttäjätarinat. H1 näki, että kertakäyttöiset dokumentit päivittyvät myös, kun niistä tehdään uusi versio. Esimerkiksi muuttamalla aiempaa toiminnallisuutta tai sen osaa uudella käyttäjätarinalla. Muutosten dokumentaatio nähtiin kaikista vaikeimmaksi asiaksi ylläpitää (H3, H4).

Vastuualueiden osalta haluttiin selvittää, *kuka on vastuussa eri dokumenttien päivityksestä*, tässä haluttiin selvittää, mikäli dokumentin luoja ja päivittäjä on eri henkilö tai rooli. Päivityksestä vastaava rooli oli aina sama kuin dokumentin luoja (H1). Tuotteenomistaja nähtiin edelleen päivityksestä vastaavana, mutta ei lähtökohtaisesti itse tee päivitystyötä (H4). Myös H1 koki, että tuotteenomistaja ylläpitää osaa dokumentaatiosta, mutta ei koko kokonaisuutta. Tämä korostuu etenkin järjestelmän tukemien prosessien ja järjestelmään liittyvien päätösten osalta. Haastateltava peräänkuulutti liiketoiminnan edustajien osuutta myös teknisen dokumentaation ajantasaisuuteen (H1). Dokumentaation automatisoimiseen pyrki H2, joka näki myös, että dokumentaation luominen pitäisi sisällyttää osaksi kehitystyötä. Näin dokumentti pysyy väkisinkin ajan tasalla.

Päivityksen tiheydestä kysyttiin, *millä aikavälillä eri dokumentteja päivitetään*? Kehityksen tahtiin päivittyvä dokumentaatio koettiin yleisimmäksi aikaväliksi; Päivittäin käytettävät dokumentit päivitetään joka päivä ja kehityssprintin dokumentit päivitetään sprinttivälein (H1, H2, H3, H5). Isomman kuvan dokumentit päivitettiin tarpeen mukaan, eli silloin kuin niihin kaivattiin muutoksia, tai kun ne eivät olleet enää ajan tasalla (H1, H4, H6). Luonnollisesti kaikki automatisoidut dokumentit päivittyivät aina muutosten yhteydessä (H2).

Dokumentin päivitykseen liittyvistä motivaatiotekijöistä kysyttiin, *koetaanko jonkun dokumentin päivitys tärkeämmäksi kuin muiden.* Kehityksen ja toiminnan ohjaamisen dokumentit olivat vastaajien mielestä tärkeimpiä, jotta kehitys ei hidastu puutteellisen dokumentin vuoksi (H1, H2, H4). H2 myös korosti, että puutteellinen perehdytykseen käytettävä dokumentaatio hidastaa kehitystyötä, joten hän näki sen päivittämisen motivaatiot korkealla. Isomman kuvan dokumenttien päivityksen vastauksissa oli eroja. H5 koki niiden päivittämisen helpommaksi, kuin yksityiskohtaisempien dokumenttien. Hän näki, että ylätason dokumenttien päivityksen prosessista oli vaivattomampaa pitää kiinni koska ne eivät vaadi päivittämistä tiheään tahtiin. Sama ajatusmalli toistui myös yksityiskohtaisimmissa dokumenteissa; jos ne muuttuvat päivittäin, on niiden päivittäminen lisätyötä ja toisaalta myös turhaa, koska samaan dokumenttiin pitää palata joka päivä. Osa haastateltavista (H1, H2, H6) taas näkivät ison tason dokumentit unohtuvan päivityssyklistä, koska järjestelmän iso kuva muuttuu harvemmin, ei ylemmän tason dokumentteihin palata niin usein. H3 summaa vielä yleisen motivaatiotekijän:

> Pakko on paras motivaattori näissä monissa, mutta kyllä ne [dokumentit] on lähtökohtaisesti vanhentuneita kaikissa projekteissa.

Edelliseen esitettiin lisäkysymys teoriasta löytyneet väitteen mukaan motivaatiotekijöistä, *päivitetäänkö jotain teknistä dokumenttia, jos se tuottaa suoraa hyötyä omiin työtehtäviin.* Tähän kaksi (H2, H6) haastateltavaa sanoi, että kaikki dokumentit ovat sellaisia. H2 perusteli edellisen, etteivät he tee mitään turhaa ja kaikki dokumentaatio on heille hyödyllistä. H3 sanoi, että suora hyöty auttaisi dokumenttien päivityksessä, mutta sitä ei kukaan vapaaehtoisesti tule tekemään. H4 näki, että dokumentit ovat tarkoitettu yhteiseen käyttöön toteamalla:

> Mikään näistä ei ole pelkästään minun käyttöön. Kaikkiin liittyy sitten joku muukin ihminen, niin sitten se ei ole pelkästään, että sitä tekisi vain oman selustan turvaamiseksi. Eli siinä on joku syy sille toisellekin dokumentoida sitä asiaa.

6.3 Tiedon jakaminen

Kolmannessa teemassa keskityttiin tiedon jakamiseen ja sen välineisiin. Ensimmäisenä kysyttiin, *miten dokumentaatio on saatavilla kehitystiimin sisällä,* johon annettiin auttava lisäkysymys, *missä kanavissa dokumentit jaetaan.* Mikäli vastaukseksi saatiin vain lista työkaluja, kysyttiin seuraavaksi, *kenellä kaikilla on pääsy ko. työkaluun.* Vastaukset kysymyksiin olivat hyvin yhtenäisiä.

Pääasiassa kaikki kehityksen hallintaan ja ohjaukseen liittyvä jaettiin keskitetysti kehityksenhallintaan käytetyssä työkalussa. Kaikilla haastateltavilla oli käytössä joko Jira, Target Process tai Trello, johon kaikilla kehitystiimiin kuuluvilla oli täysi näkyvyys projektinsa osalta. Isomman kuvan dokumentit olivat jakautuneet vahvasti eri kanaviin. Järjestelmän kuvaukset ja laajemmat vaatimusmäärittelyt olivat keskittyneet Wiki-tyyppisiin kirjastoihin, joihin kaikilla kehitystiimin jäsenillä oli myös pääsy. Laajemmassa kuvassa järjestelmään liit-

tyvät dokumentit olivat aiemmin mainittujen kanavien lisäksi sirpaloituneet SharePoint:iin, Google Docs:iin, pikaviesti-kanaviin (Slack, Skype) ja tiedostopalvelimille. H1 mainitsi vielä, että kehitykseen liittyvät päätökset olivat vain hänen saatavillaan, omissa muistiinpanoissa.

Tiedon löydettävyydestä ja jäljitettävyydestä kysyttiin, *koetko sinulle oleellisen tiedon olevan helposti löydettävissä.* Kyllä tai ei -vastausten jälkeen kysyttiin aina; *miksi?* Tiedon sirpaloitumisen takia koettiin, että tiedon etsimiseen tuli käytettyä liikaa aikaa (H1, H2, H4, H6). Lähtökohtaisesti tieto oli kuitenkin aina löydettävissä, poissulkien ne tapaukset, joissa dokumentaation työkalua ei käytetty eikä tietoa ole olemassa kirjallisessa muodossa (H1). H2 kävi hakemassa liiketoiminnan prosesseihin liittyvän tiedon suoraan liiketoiminnan edustajilta suullisesti. Kaksi haastateltavista (H3, H5) oli ollut itse suunnittelemassa dokumentaation prosessia tai valitsemassa työkaluja. He kokivat, että tieto on paremmin löydettävissä kuin ne vastaajat, jotka eivät ole olleet päätöksenteossa mukana. H4 kertoi, että tiedon sirpaloituneisuus eri kanaviin johtaa siihen, että seuraajalleen hän lähinnä kertoo, mistä dokumentit löytyvät, koska dokumentit ovat hajautuneet niin moneen eri paikkaan.

Tiedon yhtenäisyydestä kysyttiin, *koetko sinulle oleellisen tiedon olevan helposti ymmärrettävissä.* Kaikki vastaajat kokivat ymmärtävänsä dokumentaation sisällön, jos heillä siihen on pääsy. H1 näki, että koodin kommentit olivat ainoa dokumentaatio, jossa voisi tulla ymmärrys vastaan sen käytön ohjeistuksen puutteen vuoksi. H4 summasi, ettei tuotteenomistajalla, joka ei itse tuota lähdekoodia ole ymmärrystä sen lukemiseen. Toisaalta hän myös mainitsi, ettei tämä koskaan ole ollut este työnteolle, koska joku muu pystyy lähdekoodin hänelle tulkkaamaan ymmärrettäväksi.

6.4 Tiedon käyttö

Neljäs teema dokumentaation prosesseissa oli tiedon käyttö. Ensimmäisenä rajattiin pois ne dokumentit, joita haastateltava ei tosiasiassa käytä itse, kysymällä *mitä teknistä dokumentaatiota käytät työssäsi projektiisi liittyen.* Vastausten jälkeen kysyttiin, *miksi? Mitä hyötyä nämä dokumentit tarjoavat työllesi?* Dokumenttien käytöstä tunnistettiin useita erilaisia hyötyjä. Yleisin vastus oli päivittäisen työn seuranta ja johtaminen. Dokumentilla ohjattiin kehitystyötä, seurattiin ja pyrittiin kasvattamaan tehokkuutta (H1, H2, H3, H5, H6). Vaatimusmäärittelyt ja käyttäjätarinat osaltaan mahdollistavat työnteon, toimimalla rajapintana tuotteenomistajan ja kehittäjien välillä (H3). Kaikki haastateltavat mainitsivat eri tavoin, että dokumenteilla tehostettiin kommunikointia niin kehitystiimin kuin sidosryhmien välillä. H2 koki, että kommunikaatio on tehokkaampaa, kun asiaa mallintaa ja luonnostelee kollegalle, kuin jos vain yrittäisi sanallisesti selittää toimintoja.

Isomman kuvan dokumentaatiota käytettiin kokonaiskuvan hahmottamiseen ja perehtymiseen uuteen työhön (H1, H4). Lisäksi niistä haettiin historiatietoa aiemmasta kehitystyöstä, kun alettiin kehittää uutta vanhan päälle (H3). H3 myös mainitsi, että dokumentit auttavat tuotteenomistajaa priorisoimaan ja

tekemään päätöksiä. Päätöksentekoon dokumentit toivat varmuutta myös, jos päätöksiä haettiin ylemmiltä tahoilta kehitykseen liittyen. Näiden lisäksi H5 toi esiin, että käyttää olemassa olevia dokumentteja puuttuvien dokumenttien tukena.

Seuraavaksi haluttiin selvittää, onko mitään sellaista tietoa, mitä haastateltavat tarvitsevat, mutta sitä ei ole dokumentoitu. Heiltä kysyttiin, *minkälaista dokumentaatiota tarvitset työssäsi, mutta sitä ei ole saatavilla tai se ei ole ajan tasalla?* Yleisimmäksi vastaukseksi tähän kysymykseen annettiin liiketoiminnan prosessien kuvaukset (H1, H2, H5, H6). Ne eivät olleet joko ajan tasalla, olivat sirpaloituneet useaan paikkaan tai puuttuivat kokonaan. Liiketoiminnan prosessin läpiviemiseksi saatettiin käyttää useampaa tietojärjestelmää, joten tieto on hajautunut eri projektien välille (H5).

Arkkitehtuurikuvauksiin ja muihin vahvasti teknisiin kuvauksiin toivottiin useampaa näkökulmaa, joissa asia esitettäisiin esimerkiksi tuotteenomistajan tarpeiden kannalta (H1, H3). Teknisiin kuvauksiin toivottiin vielä selkeämmät reunaehdot (H1, H3). Lisäksi teknisempiin dokumentteihin tarvittiin joku tulkkaamaan tietoa (H3) ja toisaalta myös lisäkuvauksia, mitä mikäkin tieto tai linkitys tarkoittaa (H1). Historiatietoa myös tarvittiin nykyistä enemmän, koska muutoksia dokumentoitiin heikosti (H4). H1 toivoin yksittäisen toiminnallisuuden kannalta kehitys -ja korjaushistoriaa sekä viimeisintä ajankohtaista tietoa toimintoon liittyen.

Edelliseen kysymykseen haettiin tarkentavia vastauksia kysymällä, *miksi tarvitset juuri tällaista dokumentaatiota?* Liiketoiminnan prosessisen dokumentaatiolla haluttiin ymmärtää loppukäyttäjän tarpeita ja tavoitteita paremmin. Tällä tiedolla pystyttäisiin kehittämään parempaa käyttökokemusta ja laadukkaampaa järjestelmää, joka palvelee loppukäyttäjää tehokkaammin. Tällä hetkellä haastateltavat kokivat, että heidän pitää osittain arvata, mitä loppukäyttäjä haluaa järjestelmältä. (H2, H5, H6.) H2 kuvailee liiketoiminnan prosessien puutteiden ongelmaa:

> […] on hyvä, kun tekee mahdollisimman pienesti ja iteroi mahdollisimman aikaisin loppukäyttäjän nähtäväksi ja ottaa niiltä palautetta vastaan, niin me ollaan ratkaistu se ongelma sillä, että mennään nopeasti. Mutta jos haluaisi tehdä jotain isompaa ja hartaammin, niin se [loppukäyttäjän tarpeiden ymmärtäminen] olisi aika tärkeää.

Teknisten kokonaisuuksien kuvaukset nähtiin työn elinehtona (H1, H3). Tuotteenomistajan on ymmärrettävä, mitä dataa on saatavilla ja miten se liikkuu järjestelmän sisällä tai eri järjestelmien välillä (H1). Tekniset reunaehdot asettavat rajat sille, mitä tuotteenomistaja voi järjestelmältä edellyttää ja vaatia (H3). Muutosten tehokkaammalla dokumentaatiolla nopeutettaisiin käynnissä olevaa kehitystyötä (H1). Myös virheiden tunnistaminen helpottuisi, kun ymmärtää mitä vaatimuksia toimintoon on jo määritelty (H4).

6.5 Tiedon johtaminen

Teoriaosuudessa selvitettiin, että dokumentaation prosessia tarvitsee johtaa kuten järjestelmäkehitystä. Tämän vuoksi viidenneksi teemaksi koostui tiedon johtaminen. Haastateltavilta kysyttiin aluksi, tiesivätkö he, *suunniteltiinko dokumentaation tuottamista tai hallintaa ennen projektia?* Dokumentaation suunnitelmallisuus oli vähäistä haastateltavien projekteissa. Lähinnä johtamista tapahtui työkalujen valinnassa, jolloin niitä on hankittu tarpeen perusteella ja pohdittu investoinnin järkevyyttä (H3). Kuitenkaan työkalun hankinnan jälkeen dokumentaation johtaminen oli jäänyt (H1). Vain H5 vastauksessa kuvailtiin suunnitelmallisuutta dokumentaation kokonaisvaltaiseen hallintaan:

> Me tehtiin jonkunlainen lista, mitä dokumentaatiota pitää saada pihalle, ja mitä pitää alkaa tuottamaan. Ja sitä on muutettu matkan varrella sen mukaan, mitkä on koettu oikeasti järkeväksi. Siellä oli alkuun paljon enemmänkin dokumentaatiota, joista tiputettiin osa ja muokattiin osaa, että mitkä ovat oikeasti valideja.

Tuotteenomistajan ollessa vastuussa koko järjestelmän tilasta, haluttiin selvittää, *johdetaanko tai seurataanko dokumentaation prosessia jatkuvasti projektissa?* Vastaajilta, jotka myönsivät johtamista tapahtuvan (H3, H5), kysyttiin *miten ja kenen toimesta* dokumentaation prosessia johdetaan tai hallitaan. H5 myönsi olevansa ainoa, joka seuraa dokumentaation päivittymistä oman tuotteensa osalta. H3 totesi, että Scrum-prosessi pakottaa tietynlaisen dokumentaation tuottamiseen, jotta kehityssykli ei keskeydy:

> […] tuotteenomistajan roolissa käytännössä se prosessi olisi pysähtynyt, jos niitä dokumentteja, jotka kuuluvat omaan tonttiin ei olisi tuotettu. Eli käytännössä kun mentiin jonkinasteisessa puhtaassa Scrumissa, niin ne oli pakko olla. Ja se oli tuotteenomistajan vastuulla hallita, eli tiimi olisi muuten ollut tekemättä mitään […]

Kaksi vastaajaa (H2, H6), näkivät että dokumentaation johtamisen vastuu on jakautunut kaikille tasavertaisesti. H2 kuitenkin koki, että tasa-arvoisessa tiimissä tämä ei ole ongelma. H6 kuvaili tilannetta seuraavasti:

> Johtajuus selkeästi puuttuu, ei siellä kukaan katso, että kaikki dokumentaatio on. Ehkä jokainen omalta osa-alueeltaan vahtii sitä dokumentaatiota.

Tässä vaiheessa, kun kaikki dokumentaation vaiheet olivat käyty läpi, kysyttiin haastateltavilta ongelmakeskeisesti, *mitkä koet suurimmiksi haasteiksi nykyisessä tavassa tuottaa tai ylläpitää dokumentaatiota?* Yleisimmät ongelmat olivat ajan ja resurssien puute dokumentaation päivitykseen. Haasteita loi ketterän kehityksen prosessille luonteenomaiset jatkuvat muutokset, jotka osaltaan loivat lisää painetta dokumentaation päivitykseen (H1). Dokumentin jäädessä muutoksista jälkeen, sen päivityksen taakka kasvaa, joka osaltaan siirtää päivitystä edelleen. Näin dokumentti jää kokonaan prosessista pois. (H3.) Lisäksi dokumentin arvoa ei koeta korkeaksi, silloin kun sitä ei akuutisti tarvitse. Esimerkiksi pereh-

dytystilanteessa osa dokumenteista tehostaa toimintaa huomattavasti, jolloin ajantasaisen dokumentin hyödyt tulee esiin (H2).

Automatisointi ja dokumentaation tuottamisen sulauttaminen kehitysprosessiin poistaisi päivittämiseen liittyviä työtehtäviä, mutta kaikkea dokumentaatiota ei pysty automatisoimaan (H2). Yhdessä sovittujen käytänteiden puute nousi myös ongelmaksi (H3). Kehitystiimin sisällä ei tiedetä miten dokumentaatiota pitäisi tuottaa ja mihin, jolloin puuttuu yhteinen konsensus siitä, miten dokumentteja tuotetaan. Sisällön ongelmaksi muodostui myös se, että vanha ja uusi tieto on sekaisin, jolloin tieto ei ole siistiä. Dokumentit ovat myös puutteellisia tai niistä puuttuu jotain keskeistä. (H1.) Dokumentaatiolle tarvitaan selkeä struktuuri ja vastuualueet (H6).

Tuotteenomistajan tehtäviin kuuluu vahvasti eri sidosryhmille viestiminen, joten haastateltavien taustan mukaisten työtehtävien liittyessä tähän, kysyttiin, *tuleeko sidosryhmiltä tilauksia (ts. tarpeita) dokumentaatio tuottamiseen liittyen?* Mitään uutta dokumentaatiota ei kukaan haastateltavista ollut tuottanut sidosryhmille. Lähinnä sidosryhmiltä tulee kyselyitä, onko dokumentit ajan tasalla (H1, H2). Ajan tasaisia dokumentteja on sitten käytetty kommunikointiin (H2), työn laadun arviointiin (H2) tai tarjousten pyytämiseen ulkopuolisilta toimijoilta (H5). Uusilta ulkopuolisilta toimijoilta, joiden toimeksiantoon kuuluu koodin tuottaminen, on tullut myös pyyntöjä teknisen toteutuksen kokonaiskuvauksista (H5). Nämä osuvat kuitenkin enemmän perehdytyksen alle. Sidosryhmien tilaukset liittyivät enemmän käyttömanuaaleihin ja käytön opastukseen (H1, H4), jotka eivät kuulu tämän tutkimuksen piiriin.

Sidosryhmien ollessa mukana eri tasoilla projektissa haluttiin vielä varmistaa, ettei sidosryhmille viestintä ole minkään dokumentaation tuottamisen pääasiallinen motiivi kysymällä, *tuotetaanko jotain teknistä dokumentaatiota vain, koska on raportoitava sidosryhmille?* Kukaan haastateltavista ei tunnistanut sellaisia teknisiä dokumentteja, joiden tuottamisen tai päivityksen motivaatio tulisi suoraan sidosryhmiltä. Mikäli kehityksen tilanteessa oli aikataulutusongelmia, niistä on kommunikoitu eri kehityksen seurantadokumenteilla sidosryhmille (H4).

Tiedon johtamisen teeman viimeinen kysymys liittyi myös tuotteenomistajan työtehtäviin. Koska tuotteenomistajat tilaavat kehityksen vaatimusmäärittelyjen kautta, haluttiin selvittää, tilataanko dokumentaatiota samalla tavalla kuten koodin tuottamista. Haastateltavilta kysyttiin, *tilaavako he itse dokumentaatiota kehitystiimiltä, mikäli havaitsevat jollekin dokumentille tarvetta.* Tällä kysymyksellä haluttiin myös osittain selvittää, nähdäänkö dokumentaatio yhtä arvokkaana kuin kehitystyö tuotteenomistajan näkökulmasta. Suurin osa haastateltavien tilauksista on ollut päivitys- tai täydennystoiveita olemassa oleviin dokumentteihin (H2, H3, H5). Nämä täydennykset saatiin myös käydä vain keskusteluina (H1). Uutta dokumentaatiota tilattiin, jos olemassa olevasta projektista ei ollut minkäänlaista teknistä dokumentaatiota luotuna (H1, H4). Palavereita varten dokumentaatiota pyydettiin päivittämään, jotta voidaan käsitellä ajan tasaista tietoa (H3, H6). Tuotteenomistajan rooli on kuitenkin vahva dokumentaation tuottamisen kannalta, kuten H6 tiivisti:

> [...] jos tilaan [dokumentaatiota], niin tilaan itseltäni.

6.6 Ketterän kehityksen periaatteet dokumentaation prosessissa

Agile Manifeston peruspilarien mukaisesti pyritään minimaaliseen dokumentaatioon. Silloin kun dokumentaatiota tuotetaan, se pitäisi tuottaa myös ketterien periaatteiden mukaisesti. Viimeiseksi teemaksi valittiin täten ketterän kehityksen periaatteet dokumentaation prosessissa.

Ensimmäiseksi kysyttiin, *tuotetaanko dokumentaatiota iteratiivisesti?* Tähän yleensä vastattiin yhdellä sanalla myöntävästi, joten haastattelija lisäsi lisäkysymyksen, *kerääntyykö dokumenttien tuottaminen ja päivittäminen tiettyyn ajanjaksoon?* Lisäyksellä saatiin luultavasti todenmukaisia vastauksia, koska kaikkia dokumentteja päivitettiin ajoittain, mutta ei välttämättä kehityksen kanssa samassa tahdissa. Scrum-prosessiin kuuluvat dokumentaatiot päivittyvät iteratiivisesti kehityksen mukana (H1). Lisäksi kehittäjien tuottama dokumentaatio, joka on jatkuvassa käytössä usealla kehittäjällä, päivittyy iteratiivisesti (H6). Isomman kuvan dokumentaatiot koettiin ongelmalliseksi. Esimerkiksi arkkitehtuurin dokumentaatiot päivitettiin joko tarpeen mukaan (H1) tai ne kasaantuvat resurssien puutteen vuoksi, jolloin niiden päivitystä joudutaan jopa odottamaan kehitystiimin osalta (H6). Resurssipulan vuoksi myös prosessinmukaiset dokumentaatiot saattavat kasaantua pieneen ajanjaksoon juuri ennen kehityksen aloitusta, mikäli vain yksi tuotteenomistaja on vastuussa vaatimusmääritysten kirjoittamisesta (H4). Tahtotila kuitenkin on tuottaa dokumentaatiota iteratiivisesti, kuten H5 summaa:

> [Dokumentaatiota] kerätään aika pitkälle iteratiivisesti [kehityksen] mukana, vaikka [dokumentaatio] tulee vähän perässä, […] ei kuitenkaan ole niin, että kerätään [päivittäminen ja tuottaminen] johonkin ajankohtaan, että tehdään viikko dokumentaatiota.

Seuraava kysymys keskittyi resurssien ketteryyteen, jota selvitettiin kysymyksillä, *tuottavatko kaikki kehitystiimin jäsenet dokumentaatiota, vai kerääntyykö dokumentaation tuottamisen tai päivittämisen vastuu tietyille avainhenkilöille.* Vain H2 koki, että dokumentaatio todella tuotetaan ketterästi resurssien osalta. Vastauksista nousi esille kaksi titteliä, joille dokumentaation tuottamiseen kuluu eniten resursseja: tuotteenomistaja (H1, H3, H4) sekä kehitystiimin johtohenkilöt (H1, H3, H5, H6), joilla Scrum-tiimissä viitataan Scrum Master -roolissa toimiviin henkilöihin. Junioritason kehittäjiltä ei dokumentaation tuottamista vaadita (H1), tai dokumentaation vastuuta tai tehtäviä ei ole alun perinkään heille annettu (H3).

Viimeinen kysymys liittyi dokumentaatiolle varattuun resurssiin. Se sivusi myös dokumentaation arvon näkemistä kehitystyöhön verrattuna. Haastateltavilta kysyttiin, *varataanko dokumentaation tuottamiselle resursseja kehitysprintin aikana?* Vastaukset jakaantuivat kahteen osaan. Suurin osa haastateltavista koki, että dokumentaation tuottamiseen ja päivittämiseen kuluva aika kuuluu tehdä oman työn ohella (H1, H4, H5, H6), eikä siihen varattu erikseen aikaa kehityssprinteistä (H1, H5, H6). Toisaalta dokumentaatiolla varattiin aikaa kehityssprintiltä (H2, H3), ja dokumentaation tuottaminen kuului ”valmiin määri-

telmään" (Definitions of Done) (H2). Tällöin käyttäjätarina ei ole valmis ennen kuin tarpeellinen dokumentaatio on tuotettu. H1 pohti ajan varaamisen suhdetta dokumentaation tuottamiseen näin:

> … [dokumentaatio] oli pitänyt huomioida paremmin ajankäytössä, jolloin sitä olisi tullut tehtyä enemmän. Palaan taas siihen johtamiskysymykseen, että ei sitä myöskään millään tavalla johdeta niin ei sitä ajatella niin.. [tärkeänä kuin kehitystä]

6.7 Yhteenveto

Kuudennen luvun tarkoitus oli tuottaa yksityiskohtaista tietoa teknisen dokumentaation prosessin haasteisiin. Tulokset täsmentävät kirjallisuuskatsauksessa löydettyjä haasteita. Tulokset esitellään seuraavassa taulukossa (taulukko 4), samassa muodossa ja teemoittain, kuten kirjallisuuskatsauksen tulokset. Molemmissa; haastatteluissa ja kirjallisuuskatsauksessa esiin tulleet haasteet on merkitty asteriskilla (*).

TAULUKKO 4. Tutkimuksen haastatteluissa esiin tulleet haasteet

Teema	Haasteet
Tiedon tallennus	Resurssien riittämättömyys*
	Puutteellinen dokumentaatio*
	Puuttuva dokumentaatio*
	Liian raskas dokumentaatio*
	Dokumentaation tuottaminen epävirallisiin kanaviin (muistiinpanot)*
	Ei-toiminalliset määritykset ei mitattavia
	Ei yhtenäistä tapaa dokumentoida
	Epäformaalit dokumentaation tavat
	Dokumentaation sirpaloituneisuus
	Dokumentaatio vain ylätasolla, yksityiskohtaisuus puuttuu
	Dokumentaatiosta ei yhteenvetoa
	Dokumentin arvoa ei tunnisteta
Tiedon kehitys ja päivitys	Resurssien riittämättömyys*
	Muutosten hallinta nopeasti muuttuvien vaatimusten vuoksi*
	Dokumentit ei ajan tasalla*
	Motivaation puute*
	Liiketoiminnan edustajien osuuden puuttuminen
	Käytännön hyödyn puute
Tiedon jakaminen	Dokumentaation sirpaloituneisuus*
	Tiedon huono löydettävyys*
	Dokumentaatiota ei tuotettu yhteisiin kanaviin*
	Tieto löydettävissä vain lähdekoodista*
	Tiedon eri kanaviin perehdyttäminen, tiedon käyttöön opastus puuttuu
	Vain suullinen tieto saatavilla
Tiedon käyttö	Puuttuva dokumentaatio
	Puutteellinen dokumentaatio
	Historiatiedon puutteellisuus

Tiedon käyttö	Muutosten dokumentaation puutteellisuus Sirpaloitunut dokumentaatio Usean järjestelmän välille hajautunut dokumentaatio Yhden näkökulman kautta tuotettu dokumentaatio Teknisen toteutuksen reunaehdot puuttuvat Tiedon tulkki tai lisäkuvaukset puuttuvat Ajankohtainen tieto puuttuu Arvauksen varaan jäävät päätökset Uusi ja vanha tieto sekaisin keskenään
Tiedon johtaminen	Puutteellinen johtaminen Struktuurin puute Yhteisten käytänteiden puuttuminen Vastuun jakaantuminen Vastuualueita ei nimetty
Ketterän kehityksen periaatteet dokumentaatiossa	Dokumentaation tuottamisen ja ylläpidon vastuu keskittyy avainhenkilöille * Dokumentaatiolle ei varata aikaa* Dokumentaatio ei pysy kehityksen mukana* Resurssien puute Dokumentaation tuottaminen ja päivitys kerääntyy tiettyihin ajanjaksoihin

Haastatteluista saatiin kerättyä lukuisia uusia haasteita, sekä dokumenttikohtaisia haasteita teemojen alle. Monessa teemassa toistuivat jo kirjallisuusosuudessa esiin tulleet haasteet. Toisaalta *tiedon käytön* ja *tiedon johtamisen* teemojen alle ei haastatteluissa ilmentynyt yhtään samaa haastetta, mitä kirjallisuudesta löydettiin. Taulukossa esitettyjä haasteita on pohdittu seuraavassa luvussa *Tulokset ja Johtopäätökset.*

7 TULOKSET JA JOHTOPÄÄTÖKSET

Tässä luvussa esitellään kirjallisuuskatsauksessa tunnistetut haasteet teknisessä dokumentaatiossa ketterissä kehitysprojekteissa, ja verrataan niitä empiirisessä osuudessa löydettyihin haasteisiin. Näiden tulosten perusteella vastataan tutkimuskysymykseen *mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?* Lisäksi luvussa esitetään myös johtopäätöksiä vastaamalla kysymykseen, mitä syitä haasteiden takana on, sekä tunnistetaan hyötyjä teknisen dokumentaation tuottamiseen ja ratkaisuja teknisen dokumentaation haasteiden selättämiseksi.

7.1 Tulosten vertailu

Kirjallisuusosuuden tarkoituksena oli löytää olemassa olevissa tutkimuksissa tunnistetut haasteet teknisessä dokumentaatiossa projekteissa, joita toteutetaan ketterin menetelmin. Empiirisen osuuden haastatteluaineistolla pyrittiin tunnistamaan uusia haasteita, joita aiemmissa tutkimuksissa ei mainittu sekä täydentämään kirjallisuusosuuden haasteita ja tunnistamaan syitä niiden taustalla.

Kirjallisuusosuuden perusteella aineistoa käsiteltiin teemoittain. Ensimmäisiksi teemoiksi valikoitui dokumentaation perusprosessin vaiheet. Niistä ensimmäinen on *tiedon tuottaminen.* Haastatteluissa esiin tulleet haasteet keskittyivät dokumentaation tuottamisen tapojen eriäväisyyksiin. Dokumentteja tuotettiin epäformaalein metodein jokaisen itse hyväksi toteamalla tavalla. Lisäksi dokumenttien sisältämä tieto oli kirjattu ylös vain ylätasolla, jolloin yksityiskohtaiset tiedon puuttuivat. Toisaalta koettiin myös, että dokumentaatiosta puuttui yhteenveto, joka kiteyttäisi dokumentaation tiedon. Myös dokumenttien arvoa ei nähty silloin, kun niitä pitäisi tuottaa. Kirjallisuudessa esiin tullutta haastetta vaatimusten huonosta jäljitettävyydestä ei mainittu ollenkaan. Osaksi ehkä siitä syystä, että tuotteenomistajan työtehtävät keskittyvät vaatimusmäärittelyjen tuottamiseen liiketoiminnan vaatimusten perusteella. Näin ollen tuotteenomistajat omaavat paljon hiljaista tietoa siitä, mistä vaatimukset ovat tulleet.

Seuraava teema on *tiedon kehitys ja päivitys*. Tämän teeman alla kirjallisuus ja haastattelut toivat esiin hyvin samankaltaisia haasteita. Haastatteluissa tuli lisänä ilmi, että dokumentaatiosta puuttui liiketoiminnan näkökulma, jolloin tieto päivitettiin vain kehitykseen osallistuvien näkökulmasta. Lisäksi dokumenttien käytännön hyötyä ei koettu tarpeeksi suureksi, jotta niitä olisi päivitetty.

Kolmas teema oli *tiedon jakaminen*. Myös tämän teeman alla toistuivat samat haasteet niin kirjallisuudessa kuin haastatteluissa. Muita haasteita löytyi haastattelujen pohjalta opastuksen ja perehdytyksen puute tiedon eri kanaviin ja tiedon käyttöön. Vastauksista myös korostui, että puutteellisen dokumentaation vuoksi, ainoa tiedon jaon kanava on suullinen dokumentaatio, joka joudutaan aina erikseen hakemaan ko. henkilöltä.

Viimeinen teema dokumentaation prosessiin liittyen on *tiedon käyttö*. Tulokset eriävät suuresti kirjallisuuden ja haastatteluiden välillä. Kirjallisuudessa korostettiin, ettei dokumentaatio ole ymmärrettävää lukijalle tai se on ymmärrettävissä eri tavalla kuin kirjoittaja sen tarkoitti. Haastatteluissa ei suoraan myönnetty, että tietoa ei ymmärretä, mutta mainittiin että osalle dokumenteista tarvitaan tulkki tai lisäselvityksiä. Näitä toivottiin aiemmissa osuuksissa esiin tulleiden dokumentaation puuttumisen, puutteellisuuden, sirpaloitumisen ja vanhentuneen tiedon vuoksi. Lisäksi haastatteluissa korostui, että tieto dokumenttiin on tuotettu vain yhdestä näkökulmasta. Tällöin se ei palvele järjestelmäkehityksen päätöksentekoa tai lukijan muita tarpeita. Tällaisia tarpeita on esimerkiksi järjestelmän reunaehtojen asettamat rajoitukset tai usean järjestelmän välille hajaantuneen dokumentaation ymmärtäminen. Ymmärrettävyys kärsi myös, mikäli dokumentissa sekoittui vanhentunut historia tieto ja uusi päivitetty tieto.

Viides teema keskittyi *tiedon johtamiseen.* Kirjallisuudessa esiin nousi dokumentaation suunnitelmallisuus sekä seuranta. Haastatteluissa nousi samat teemat eri näkökulmista. Nähtiin että, dokumentaatiota johdettiin puutteellisesti ja siitä puuttui struktuuri. Näiden lisäksi johtamisen haasteiksi koettiin vastuiden epäselvyys ja jakaantuminen sekä yhteisten käytänteiden puuttuminen.

Viimeinen teema on *ketterän kehityksen periaatteet dokumentaatiossa.* Kirjallisuudesta ja haastatteluissa toistuivat samat haasteet; dokumentaation tuottaminen kasaantuu yksittäisille henkilöille, dokumentaatio ei pysy kehityksen mukana ja dokumentaatiolle ei varata aikaa. Kirjallisuudessa mainittiin, että dokumentaation tuottaminen keskitetään taidoiltaan heikommille kehittäjille. Haastatteluissa päädyttiin päinvastaiseen tulokseen, dokumentaation tuottaminen kasaantui tiimin avainhenkilöille. Lisäksi haastatteluissa mainittiin, ettei dokumentaatiota voida kehittää ketterin metodein resurssien puutteellisuuden vuoksi. Näin dokumentaation tuottaminen ja päivitys keskittyy tiettyihin ajanjaksoihin.

7.2 Teknisen dokumentaation hyödyt ketterissä kehitysprojekteissa

Haastattelun tuloksena löydettiin useita hyötyjä eri teknisistä dokumenteista, mikäli dokumentit ovat olemassa ja ajan tasalla. Päivittäisen työn seurantaan ja johtamiseen tarkoitetut dokumentit auttavat tuotteenomistajaa ohjamaan toimintaa oikeaan suuntaan sekä seuraamaan kehitystyötä. Dokumentit luovat suunnitelmallisuutta tekemiseen, mikä auttaa resursoinnissa sekä tavoitteiden ja toteutuksen arvioinnissa. Vahvalla suunnitelmallisuudella nostetaan työn tehokkuutta. Lisäksi suunnitelmallisuus ja arviointi mahdollistavat oppimisen kehitystiimin sisällä. Tällaisia dokumentteja olivat esimerkiksi kehityssprintin suunnittelu- ja arviointidokumentit. Scrum-prosessiin vahvasti kuuluvat dokumentit mahdollistavat työnteon tuotteenomistajan ja kehitystiimin välillä. Näitä ovat esimerkiksi vaatimusmäärittelyt ja käyttöliittymiin liittyvät dokumentit.

Järjestelmän historiatiedon dokumentointi vauhditti jatkokehityksen aloittamista ja sen toteutusta. Uutta on helpompi tuottaa vanhan päälle, kun ymmärtää, miksi jotain on toteutettu olemassa olevalla tavalla. Näitä hyötyjä saatiin esimerkiksi dokumentoimalla kehityksen aikana tehdyt päätökset järjestelmään ja sen toimintoihin liittyen. Vaatimusmäärittelyjen hyvä jäljitettävyys ja muutosten dokumentointi toivat myös samoja hyötyjä.

Järjestelmän nykytilaa kuvaavilla dokumenteilla tehostettiin perehdytys- ja koulutustyötä. Myös kehitystyö on tehokkaampaa, kun päätöksiä voidaan tehdä ajankohtaisen tiedon perusteella. Dokumentit myös toivat varmuutta päätöksentekoon, myös silloin, kun päätöksiä haettiin sidosryhmiltä tai johtotasolta.

Järjestelmän tukemien prosessien ajantasaisella ja kattavalla dokumentaatiolla parannetaan mahdollisuuksia vastata liiketoiminnan odotuksiin järjestelmän toiminnoista. Ymmärtämällä loppukäyttäjän tavoitteita saavutetaan parempi käyttökokemus ja tuotetaan laadukkaampaa järjestelmää, joka palvelee loppukäyttäjää mahdollisimman tehokkaasti.

Teknisiä ratkaisuja kuvaavat dokumentit tukevat kehitystyötä asettamalla vaatimusmäärittelyille reunaehdot. Myös kehittäjien perehdytystyö tehostuu, kun dokumentit vastaavat mahdollisiin perehdytyksessä esiin nouseviin kysymyksiin.

Kokonaista järjestelmää kuvaavat tai mallintavat dokumentaatiot auttavat kokonaiskuvan hahmottamisessa, joka korostuu etenkin perehdytystyössä. Kokonaisarkkitehtuurin ja järjestelmän kuvaus mahdollistavat tällaisia hyötyjä. Näillä dokumenteilla myös varmistetaan, että kehitystiimillä ja sidosryhmillä on yhteinen ymmärrys järjestelmästä ja sen toiminnoista.

Kaikkia dokumentteja käytettiin jonkinlaiseen kommunikointiin, niin kehitystiimin sisällä kuin sidosryhmille ulospäin. Dokumentit mahdollistavat tehokkaampaa kommunikaatiota ja työskentelyä tiimin sisällä sekä kommunikoinnin ulospäin kehityksen tilasta ja järjestelmän laadusta. Mikä tahansa olemassa oleva dokumentti on myös olematonta parempi. Olemassa olevat doku-

mentit, vaikkakin puutteelliset tai vanhentuneet tukevat puuttuvaa dokumentaatiota.

7.3 Teknisen dokumentaation haasteiden ratkaiseminen

Ketterä dokumentaation prosessi on käytännössä samanarvoista kuin ketterä koodin tuottaminen. Molemmissa prosesseissa täytyy noudattaa *Agile Manifeston* perusperiaatteita. Seuraavaksi on esitetty teknisen dokumentaation tuottamisen haasteet ja mahdolliset ratkaisut niihin *Agile Manifeston* neljän periaatteen teeman alle.

Toimivaa ohjelmistoa enemmän kuin kattavaa dokumentaatiota.

Ketterien metodien periaatteiden mukaisesti dokumentaatio on pidettävä minimissään, etenkin mikäli se ei tuota jatkuvasti arvoa asiakkaalle. Asiakkaalle tuotettava arvo pitää näkyä heti käytännössä, toimivana ohjelmistona. Teknistä dokumentaatiota voidaan tarkastella saman lähtökohdan läpi; mikäli jokin dokumentaatio ei tuota arvoa sen käyttäjille, ei sitä tulisi ketterien metodien mukaisesti tuottaa. Teknisen dokumentaation yhteydessä asiakasta edustaa kehitystiimin jäsenet.

Tutkimuksessa tunnistettiin dokumentteja, joiden päivitys koettiin raskaaksi tai niiden päivitykseen ei löytynyt motivaatiota. Koska teknistä dokumentaatiota tuotetaan vain kehitystiimin tarpeisiin, tulisi se siis arvioida tuottamansa hyödyn mukaan. Dokumentaation tuottamaa arvoa tulisi siis tarkastella projektin edetessä jatkuvasti. Mikäli dokumentille ei nykyhetkessä tai tulevaisuuden skenaarioita läpikäymällä löydetä tarpeellista arvoa, tulisi sen tuottaminen ja päivittäminen keskeyttää.

Mikäli dokumentaatio koetaan arvokkaaksi kehitystyössä, sen sisällön tuottaminen ja päivittäminen voidaan priorisoida kuten kehitystehtävät. Korkeamman prioriteetin dokumentaatio on tällöin kehitystiimin ensisijaisten tehtävien joukossa. Mikäli jokin dokumentti tunnistetaan erityisen kriittiseksi pitää ajan tasalla sen automatisointiin panostaminen saattaa maksaa itsensä takaisin ajan kanssa. Jos dokumentin ylläpitoa ei teknisesti pystytä automatisoimaan sen tuottaminen pitää sisällyttää *valmiin määritelmään*. Näin dokumentaation tehtävät eivät jää muiden työtehtävien jalkoihin.

Kehittäjät suosivat nopeasti päivitettäviä kirjastoja ja hakemistoja, jotka tuottavat suoraa hyötyä kehitystyöhön. Yksityiskohtaiset hakemistot vaativat kuitenkin juuri jatkuvaa päivitystä, jotta ne pysyvät ajan tasalla jatkuvasti muuttuvassa järjestelmässä. Yksityiskohtainen tieto menettää heti validiteettinsa kun osa, jota se kuvailee muuttuu. Dokumentaatio, joka on sulautettu prosessiin, otetaan huomioon yhtä arvokkaana kuin mikä tahansa muu kehitykseen liittyvä työtehtävä. Päivittämiseen liittyvät ongelmat poistuvat, mikäli päivitys kuuluu päivittäisiin työtehtäviin. Uuden työvaiheen lisääminen pakottaa myös kehitystiimin ja sen johdon luomaan ja sopimaan yhteiset tavat tuot-

taa ja päivittää dokumentaatiota. Tuotteen, tässä yhteydessä teknisen dokumentaation laatu pysyy yhteisten toimitapojen ja kehityksen tahtiin tehtävän päivityksen myötä korkeana. Näin saadaan ajantasaista ja yhtenäistä dokumentaatiota.

Vastaamista muutokseen enemmän kuin pitäytymistä suunnitelmassa

Ketterässä kehityksessä muutos on odotettavissa. Tästä syystä suunnitelmat tehdään ylätasolla, ja niiden oletetaan jo luomisvaiheessa muuttuvan. Dokumentaatiota ei pitäisi nähdä eri tavalla. Vaikka dokumentin on oltava stabiili siinä mielessä, että sen pitää olla helposti löydettävissä ja päivitettävissä, on se muutoksille yhtä altis kuin kehitystyökin. Suunnitelma tehdään aina sen hetkisen parhaimman tiedon mukaan, mutta edes dokumentaatiossa se ei voi olla staattinen. Suunnitelman laatimisessa voidaankin jo tunnistaa riskit heti prosessin alussa.

Yksityiskohtainen dokumentaatio, joka menettää ajantasaisuutensa heti muutoksen tapahtuessa, voidaan tunnistaa dokumentaatioksi, joka helposti jää kehityksestä jälkeen. Suuremman perspektiivin dokumentit ja järjestelmäkuvaukset eivät menetä sisältönsä oikeellisuutta pienten muutosten sattuessa. Näin ollen, vaikka monikin asia järjestämässä muuttuu, voi isoa kuvaa kuvaileva dokumentti silti olla täysin validi ja paikkansapitävä tai vähintään käytännöllinen uuden työntekijän perehdytyksessä.

Ketterässä kehityksessä dokumentaatiota halutaan tuottaa kevyesti, jotta se olisi yhtä helposti ja nopeasti muutettavissa kuin itse ohjelmisto. Kevyet ja nopeat dokumentaatiotavat kuitenkin jakautuvat eri työkaluihin, lähdekoodiin, bugikirjastoihin, käyttötapauskuvauksiin ja skenaarioihin. Esimerkiksi yksittäinen käyttäjätarina ei tuota tarpeeksi tietoa itsessään lukijalle, vaan tulisi siihen ymmärrettävyyden ja kokonaisuuden hahmottamisen vuoksi lisätä oleellinen dokumentaatio käyttäjätarinaan liittyvistä muista toiminnallisuuksista ja päätöksistä sen tuottamisen takana. Suunnitelma varmistaa, että toiminnallisuus on jäljitettävissä alkuperäiseen vaatimukseen. Eri kanavissa jaetut dokumentit täytyy linkittää toisiinsa ja sisältö koota loogiseksi kokonaisuudeksi.

Matalat prosessirakenteet mahdollistavat muutokset dokumentaation tuottamiseen ja päivittämiseen koko projektin ajan. Aiemmin mainittujen hyötyjen tarkastelun jälkeen voidaan todeta, että nykyinen tapa tuottaa jotain dokumentaatiota ei ole tehokas tai että koko dokumentaation ylläpito ei tuota arvoa kenellekään. Lisäksi on huomioitava, että projektin johdon on tunnistettava, mikäli jollekin uudelle dokumentaatiolle on tarvetta silloin kun luovutaan toisesta. Lisäksi on tunnistettava, mikäli järjestelmäkehityksen puolella on syntymässä tai syntynyt jo täysin uusi kokonaisuus, joka pitää sisällyttää dokumentaation prosessiin.

Yksilöitä ja kanssakäymistä enemmän kuin menetelmiä ja työkaluja.

Vaikka menetelmät ja työkalut tuovat suuresti hyötyä kehitystiimille ja mahdollistavat ketterät periaatteet tiimissä, eivät ne kuitenkaan tuo arvoa asiakkaalle ilman kykeneviä ja motivoituneita työntekijöitä. Sama periaate toistuu doku-

mentaatiossa. Kuten aiemmin on todettu, tekninen dokumentaatio ei tuota suoraa todennettavaa arvoa ulkoiselle asiakkaalle. Lähtökohtaisesti sen tuottama arvo on vain kehitystiimin jäsenten todennettavissa

Motivoituneet työntekijät mahdollistavat dokumentaation arvon tuoton. Kuten ketterässä kehityksessä tiimin täytyy olla itseohjautuva ja siihen tulee kannustaa. Kehitystiimin jäsenten on saatava olla mukana valitsemassa dokumentaation työkaluja sekä päättämässä, mitä ja miten dokumentoidaan. Osallistaminen vahvistaa omistajuutta. Tästä syytä työkalut ja metodit eivät voi tulla projektin johdolta alaspäin. Tällöin ajaudutaan kirjallisuusosuudessa kuvailtuun ongelmaan; tutkimuksissa havaittiin, että kehittäjät näkevät dokumentaation tuottamisen toissijaisena tehtävänä, ja jopa välttelevät sen tuottamista.

Ketterien metodien perusluonteessa toistuu resurssien ketteryys. Kaikki työtehtävät ovat samanarvoisia, kun tuote on toimitettava asiakkaalle sovitussa aikataulussa. Käytännössä ylhäältä tulevan epämotivoivan työtehtävän tekemisen ongelma ratkaistaan niin, että dokumentaation tuottaminen allokoidaan yhdelle tekijälle. Yhden tekijän vastuulle jäävät dokumentaation tuottamiseen liittyvät tehtävät, jäävät helposti kehityksestä jälkeen. Tämä tarkoittaisi käytännössä sitä, että dokumentaatiosta vastaava joutuisi jokaisen sprintin jälkeen pyytämään jokaiselta kehittäjältä raportin muutoksista ja uusista ominaisuuksista, joita he ovat toteuttaneet.

Eri dokumenttien tuottamiseen tarvitaan myös erilaista osaamista. Mikäli dokumentti on lähtökohtaisesti tarkoitettu kehittäjien käyttöön, on luonnollista, että kehittäjä kirjoittaa sen. Myös eri lähtökohdat antavat erilaista perspektiiviä dokumentaation tuottamiseen; perehdytykseen tarkoitettua dokumenttia lukee uusi tiimin jäsen täysin erilaisella näkökulmalla, kuin senioritasoinen tekijä. Näin ollen työhön perehtyvällä kehittäjällä on paremmat lähtökohdat päivittää perehdytysdokumentaatiota.

Resurssien ketteryyden ja tehokkuuden kannalta dokumentoitavat artefaktin koodin kirjoittanut kehittäjä dokumentoi ominaisuuden paljon nopeammin ja kokonaisvaltaisemmin, kuin ulkopuolelta tuleva dokumentaation tuottaja. Itseohjautuva tiimi ja korkeasti motivoituneet yksilöt ratkaisevat työn kulminoitumiseen liittyvät ongelmat, ja tavoittelevat dokumentille korkeaa hyötyä.

Asiakasyhteistyötä enemmän kuin sopimusneuvotteluja.

Nopeassa tuotannossa on hyvät puolensa, etenkin tilaajan näkökulmasta; järjestelmä voidaan heti ottaa käyttöön, jolloin sen oletettu tuotto pystytään todentamaan nopeasti. Kuitenkin tuotteen elinkaaren pitkäjänteinen ajattelu saattaa unohtua, kun keskitytään vain erillisten ominaisuuksien nopeaan tuottamiseen ja toimittamisen.

Tuotettavaan järjestelmään pitää myös olla helppo jatkokehittää uusia ominaisuuksia ja muokata vanhoja. Näiden lisäksi järjestelmän tulee olla myös helposti ylläpidettävä. Jatkokehityksen ja ylläpidollisten toimien helppous korostuu etenkin tilanteissa, joissa järjestelmä on tilattu ulkoiselta toimijalta. Työn projektiluontoisuuden vuoksi on hyvin todennäköistä, että tuotteen ylläpitotoimet ja jatkokehitystyö lankeaa kehitystiimille, joka ei ole työskennellyt alkuperäisessä kehitystiimissä.

Perinteisessä järjestelmäkehityksessä korostetaan juuri ylläpidon helppoutta kattavalla dokumentaatiolla. Ketterissä metodeissa ei raskasta tai yksityiskohtaista dokumentaatiota haluta tuottaa ollenkaan. Tästä voisi vetää yksinkertaisen syy–seuraus -johtopäätöksen, että ketterillä metodeilla tuotettu järjestelmä olisi hankalampi ylläpitää. Dokumentaation puutteellisuuden vuoksi kehittäjät eivät hahmota järjestelmää kokonaisuutena eivätkä näin pysty tuottamaan ylläpitotehtäviä tehokkaasti.

Ei voida olettaa, että järjestelmän ylläpitoon ja jatkokehitykseen liittyvää teknistä dokumentaatiota osaisi tilata tuotteen tilaaja, liiketoimen edustaja. Asiakkaan näkökulmasta vain tuotteen käyttöön liittyvät dokumentit tuottavat heille välittömästi arvoa. Täten teknisen dokumentaation tilaaminen, määrittely ja tuottaminen jää projektitiimin vastuulle, tulevaisuuden ylläpitotiimiä silmällä pitäen. Näin ollen ketterän tiimin on tuotettava dokumentaatiota, jota asiakas ei ole varta vasten tilannut. Tilauksen saadakseen esimerkiksi tuotteenomistaja joutuu myymään ajatuksen tällaisen dokumentaation tilaamisesta asiakkaalle.

Teknisessä dokumentaatiossa asiakas on kehitystiimi, jolloin sen hyöty on oltava yhtä konkreettinen tiimille kuin koodin ulkoiselle asiakkaalle. Koska dokumentaatiolla ei ole suoraa liiketoimintatarvetta, jää arvon todentaminen kehitystiimin jäsenten vastuulle. Dokumentaatio tuotteena vaatii saman prosessin kuin kehitysyhteistyö asiakkaan kanssa. Siihen kuuluu avoimen keskustelun ilmapiirissä käyty tuotteen tarkastelu, mukauttaminen, priorisointi ja kaikkien sidosryhmien välinen yhteistyö. Tarkastelulla, mukauttamisella ja priorisoinnilla poistetaan turhat dokumentit ja työvaiheet, tehostetaan dokumentaation prosessia sekä keskitytään vain suurinta hyötyä tuottavien dokumenttien ylläpitoon. Yhteistyöllä sidosryhmien kanssa mahdollistetaan, että dokumentit tuottavat arvoa tiimin sisällä varmistamalla, että dokumentit ovat tarpeellisia niin tuotteenomistajille kuin kehittäjille. Erityisesti kuitenkin yhteistyöllä sidosryhmien kanssa halutaan osallistaa liiketoiminnan tai toisen kehitysprojektin edustajat dokumentaation prosessiin. Haastatteluissa kävi ilmi, että liiketoiminnallisten järjestelmän prosessien kuvauksissa on eniten haasteita. Yhteistyö mahdollistaisi dokumentin sisällön rikastamisen ulkoisten sidosryhmien tuottamalla tiedolla prosessien kulusta eri yksiköiden tai järjestelmien läpi. Jatkuva avoin kommunikointi eri yksiköiden ja projektitiimin välillä kasvattaa dokumentaation relevanssia, käytettävyyttä ja laatua.

Loppupäätelmänä tässä tutkimuksessa teknisestä dokumentaatiosta ja sen tuottamisesta ketterissä metodeissa on, että dokumentaation tuottamista ja tilaamista täytyy johtaa kuten kehitysprojektia itseään. Scrum-mallin roolien mukaisesti tämä johtamisen tehtävä lankeaa tuotteenomistajalle. Tämän henkilön tulee tunnistaa jo tilauskannan priorisoinnin vaiheessa, mitkä artefaktit ja tekniset ratkaisut tulisi dokumentoida. Työ pitää sisällyttää itse kehityssprinttiin, siitä vastaaville henkilöille ja valvoa, että se toteutetaan ketterien metodien periaatteiden mukaisesti. Muutosten ja päätösten teon vaiheessa, tuotteenomistajan tulisi myös ylläpitää dokumentaatiota siitä, mitä päätöksiä hänen omistamansa tuotteen tulevaisuudesta on päätetty.

Dokumentaation hallintaan on löydettävä eri dokumenttityypeille sopivat työkalut. Näin ollen jo järjestelmän suunnittelun vaiheessa olisi hyvä tunnistaa sellaiset toiminnallisuudet, jotka tarvitsevat erillisen tai spesifin tavan doku-

mentoida sen toteutusta. Esimerkiksi, jos tunnistetaan, että järjestelmässä on paljon teknistä rajapintatyötä, on hyvä suunnitella valmiiksi työkalut, joilla sellaista voidaan tehokkaasti ja kevyesti dokumentoida.

Dokumentaatiotyölle on myös varattava kehityssprintistä oma aikansa ja se on käsiteltävä omana tilauksenaan, jotta liian tiukka aikataulu tai tilauksen puute ei edesauta työn välttelyä tai sen tärkeyden väheksymistä. Kehitysajan varaaminen muulle työlle kuin itse kehitykselle, täytyy tietenkin hyväksyttää asiakkaalla, etenkin mikäli tuotetta tuotetaan tilaaja–toimittaja -asetelmalla. Tässä tilanteessa teknisen dokumentaation tarve on myytävä asiakkaalle tuotteen ylläpidettävyyden tai kehitystyön tehokkuuden kannalta.

Projektin loppua ajatellen, prosessissa sirpaloituneen dokumentaation kasaaminen loogiseksi kokonaisuudeksi saattaa vaatia kehitystiimiltä tai projektin johdolta suuren määrän työtunteja. Tämä korostuu etenkin, mikäli dokumentaatiota ei tuoteta samassa tahdissa kehitystyön kanssa. Tämän vuoksi olisi projektinhallinnallisesti järkevää sisällyttää dokumentaatio päivittäiseen kehitystyöhön. Kehitystyön tahdissa tehtävän dokumentaation merkitys korostuu etenkin siinä vaiheessa, mikäli projektista poistuu tekijöitä. Jälkikäteen menetetyn hiljaisen tiedon haaliminen kasaan voi olla mahdotonta tai ainakin suuritöisempää kuin ao. henkilön itse toteuttamana.

8 YHTEENVETO

Tämän pro gradu -tutkielman tarkoituksena oli tunnistaa, minkälaisia haasteita teknisen dokumentaation prosessissa on ketterissä järjestelmäkehitysprojekteissa. Tutkimuskysymyksenä oli: *"Mitä haasteita teknisen dokumentaation tuottamiseen, ylläpitoon ja käyttöön liittyy ketterissä järjestelmäkehitysprojekteissa?"*. Haasteiden kautta pohdittiin syitä niiden takana sekä haluttiin löytää ratkaisuja, kuinka tuotteenomistajat käytännössä ratkaisevat nämä dokumentaation tuottamiseen ja sen johtamiseen liittyvät ongelmat.

Jotta tutkimuskysymykseen päästiin vastaamaan, esiteltiin ensimmäiseksi, mitä ketterä järjestelmäkehitys tarkoittaa, ja miten se eroaa perinteisestä järjestelmäkehityksestä. Lisäksi perehdyttiin siihen, miksi dokumentaation tuottaminen koetaan haasteelliseksi ketterässä järjestelmäkehityksessä. Vertaamalla perinteistä järjestelmäkehitystä ketterin metodein toteutettuun kehitysprojektiin havaittiin ristiriita dokumentaation prosessissa. Ketterien metodien periaatteiden mukaisesti kehityksessä keskitytään toimivan ohjelmiston tuottamiseen kattavan dokumentaation sijaan. Perinteinen järjestelmäkehitys peräänkuuluttaa kehitystyön ja ylläpitotehtävien sujuvuutta kattavan dokumentaation avulla. Järjestelmäkehitysala on kuitenkin siirtynyt vahvasti ketteriin metodeihin.

Ketterät metodit pilkkoivat järjestelmäkehityksen prosessin iteratiivisiin kehityssprintin mittaisiin jaksoihin. Dokumentaatiota ei voi tässä tilanteessa tuottaa perinteisen kehitysmallin mukaisesti, jokaisen järjestelmäkehitysprosessin vaiheen jälkeen. Dokumentaation tuottamisen ja ylläpidon prosessit hajaantuvatkin ketterässä kehityksessä kehitystyön sekaan. Dokumentaatiota tuotetaan tarpeen mukaan, iteratiivisesti projektin edetessä. Näiden havaintojen perusteella lähdetiin keräämään haasteita dokumentaation prosessissa ketterissä järjestelmäkehitysprojekteissa, niin olemassa olevasta kirjallisuudesta kuin empiiriseen osuuden haastatteluiden kautta.

Empiirisessä osuudessa haluttiin rikastaa kirjallisuudesta löydettyjä tuloksia siitä, miten Scrum-prosessin mukaiset projektin johtohahmot, tuotteenomistajat kokevat teknisen dokumentaation prosessin haasteet omassa työssään ketterissä kehitysprojekteissa Haasteiden keräämisen yhteydessä haluttiin ymmärtää, mitä syitä haasteiden syntymisen takana on, ja miten niitä ratkaistaan tai voisi ratkaista.

Tutkimuksen kirjallisuuskatsaus ja tapaustutkimus tuotteenomistajien näkökulmasta teknisen dokumentaation tuottamisen prosessin haasteista osoittivat, että dokumentaation prosessiin ei ole käytännössä omaksuttu ketterien metodien periaatteita kokonaisvaltaisesti. Dokumentaatiota ei koeta yhtä arvokkaaksi kuin lähdekoodin tuottamista, joten dokumentaatio jää kehityksestä auttamatta jälkeen. Ketterien periaatteiden mukaista resurssien joustavuutta ei harjoiteta dokumenttien luomisessa ja päivittämisessä. Tällöin dokumentaation koettu arvo laskee entisestään, kun tieto ei ole ajan tasalla tai se on puutteellista.

Tutkimuksen tuloksia voidaan käyttää projektijohdon toteuttamaan dokumentaation suunnitteluprosessiin. Tutkimuksen haasteiden avulla voidaan tunnistaa riskejä teknisen dokumentaation prosessissa. Riskit tunnistamalla projektin johto voi välttää yleisimmät sudenkuopat dokumentaation tuottamiseen liittyen, ketterän kehityksen metodein tuotetussa järjestelmäkehitysprojektissa. Lisäksi voidaan hyödyntää syitä tunnistettujen haasteiden takana, ja pohtia ratkaisuja ennen kuin haasteet konkretisoituvat. Projektinjohto voi myös löytää valmiita ratkaisuja siihen, miten teknisen dokumentaation prosessia voidaan toteuttaa ketterien metodien mukaisesti. Dokumenttien prosessin johtoa ja seurantaa voidaan korostaa tuotteenomistajan näkökulmasta, sekä jakaa vastuualueita eri roolien välillä.

Tämän tutkimuksen pohjalta mielenkiintoiseksi jatkotutkimusaiheeksi voitaisiin valita muiden Scrum-roolien edustajien, kuten kehittäjien tai Scrum Mastereiden näkemys teknisen dokumentaation haasteista. Tuotteenomistajat tuottavat vain vaatimusmäärittelyyn ja liiketoimintatarpeeseen liittyvät dokumentaatiot. Muiden Scrum-roolien vastuulle jää myös suuri osa teknisen dokumentaation käsitteen alle kuuluvista dokumenteista. Nämä dokumentit ovat kriittisiä ylläpidon ja jatkokehityksen sujuvuuden takaamisen kannalta.

LÄHTEET

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2002). *Manifesto for Agile Software Development Twelve Principles of Agile Software*.

Brown, J. S., & Duguid, P. (2017). *The Social Life of Information: Updated, with a New Preface*. Boston: Harvard Business School Publishing Corporation 2017.

Buckland, M. K. (1991). *Information and information systems*. Westport: Preager Publishers.

Cervone, H. F. (2011). Understanding agile project management methods using Scrum. *OCLC Systems and Services*, *27*(1), 18–22.

Chopade, R. M., & Dhavase, N. S. (2017). Agile software development: Positive and negative user stories. *2017 2nd International Conference for Convergence in Technology, I2CT 2017*, *2017-Janua*, 297–299.

Clear, T. (2003). Documentation and agile methods: striking a balance. *ACM SIGCSE Bulletin*, 1–2.

Cooke, J. L. (2016). *PRINCE2 Agile : An Implementation Pocket Guide: Step-by-step Advice for Every Project Type*. IT Governance Publishing.

de Souza, S. C. B., Anquetil, N., & de Oliveira, K. M. (2005). *A study of the documentation essential to software maintenance*. (January), 68.

de Souza, S. C. B., Anquetil, N., & de Oliveira, K. M. (2006). Which documentation for software maintenance? *Journal of the Brazilian Computer Society*, *12*(3), 31–44.

Desharnais, J. M., Kocatürk, B., & Abran, A. (2011). Using the COSMIC method to evaluate the quality of the documentation of Agile user stories. *Proceedings - Joint Conference of the 21st International Workshop on Software Measurement, IWSM 2011 and the 6th International Conference on Software Process and Product Measurement, MENSURA 2011*, 269–272.

Eriksson, P., & Koistinen, K. (2005). *Monenlainen tapaustutkimus*. Helsinki.

Garousi, G., Garousi-Yusifoğlu, V., Ruhe, G., Zhi, J., Moussavi, M., & Smith, B. (2015). Usage and usefulness of technical software documentation: An industrial case study. *Information and Software Technology*, *57*(1), 664–682.

Glushko, R. J., & McGrath, T. (2005). *Document Engineering : Analyzing and Designing Documents for Business Informatics and Web Services*. Cambridge,

Mass : The MIT Press. 2005.

Grinter, R. E., Herbsleb, J. D., & Perry, D. E. (1999). The Geography of Coordination: Dealing with Distance in R&D Work. *Proceedings of the ACM Conference on Supporting Group Work*, 306–315.

Große, C. S., Jungmann, L., & Drechsler, R. (2015). Benefits of illustrations and videos for technical documentations. *Computers in Human Behavior*, *45*, 109–120.

Harris, M. D. S. (2018). *The Business Value of Software*. Auerbach Publications. 2018.

Heikkila, V. T., Damian, D., Lassenius, C., & Paasivaara, M. (2015). A Mapping Study on Requirements Engineering in Agile Software Development. *Proceedings - 41st Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2015*, 199–207.

Hirsjärvi, S., & Hurme, H. (2015). *Tutkimushaastattelu: Teemahaastattelun teoria ja käytäntö*. Helsinki: Gaudeamus Helsinki University Press.

Horch, J. W. (2003). *Practical Guide to Software Quality Management*.

IEEE Standard. (2011a). INTERNATIONAL STANDARD ISO / IEC / IEEE Systems and software engineering - Architecture description. *Ieee Standards*, *2011*.

IEEE Standard. (2011b). INTERNATIONAL STANDARD ISO / IEC / IEEE Systems and software engineering — Life cycle processes — Requirements engineering. *Ieee Standards*, *2011*.

IEEE Standard. (2018). Systems and software engineering — Developing information for users in an agile environment. *Ieee Standards*, *2018*.

Isham, M. (2008). Agile architecture IS possible - You first have to believe! *Proceedings - Agile 2008 Conference*, 484–489.

Jeremiah, J. (2015). Survey: Is agile the new norm? Noudettu osoitteesta: techbeacon.com website: https://techbeacon.com/app-dev-testing/survey-agile-new-norm

Lee, S. M., & Hong, S. (2002). An enterprise-wide knowledge management system infrastructure. *Industrial Management and Data Systems*, *102*(1), 17–25.

Lethbridge, T. C., Singer, J., Forward, A., & Consulting, D. (2003). How Software EngineeUse Documentation : The State of the Practice Documentation : *IEEE Computer Society*, 35–39.

Mahalakshmi, M., & Sundararajan, D. M. (2008). International Journal of Emerging Technology and Advanced Engineering Traditional SDLC Vs Scrum Methodology-A Comparative Study. *Certified Journal*, *9001*(6), 2–6.

Measey, P., & Radtac. (2015). *Agile Foundations : Principles, Practices and Frameworks*. Wiltshire, England : BCS 2015.

Moitra, J., & Herbsleb, D. (2001). Global software development. *IEEE Software*, *18*(2), 16–20.

Myklebust, T., Stålhane, T., Hanssen, G. K., Wien, T., & Haugset, B. (2014). Scrum, documentation and the IEC 61508-3: 2010 Software standard. *PSAM 2014 - Probabilistic Safety Assessment and Management*.

Päivärinta, T., & Munkvold, B. E. (2005). Enterprise content management: An integrated perspective on information management. *Proceedings of the Annual Hawaii International Conference on System Sciences*, *00*(C), 96.

Papadopoulos, I., Kogias, D., Patrikakis, C., & Marinou, C. C. (2017). Enhancing the student's logical thinking with Gherkin language. *IEEE Global Engineering Education Conference, EDUCON*, (April), 1543–1547.

Parnas, D. L. (2009). Future of Software Engineering, Precise Documentation: The Key to Better Software. *Communications in Computer and Information Science*, *31*, 2.

Ramesh, B., & Cao, L. (2014). Requirements validation techniques : A case of prototype of the Technical Institute for Administration Thamir Naeem Jassim Assist Lecturer Technical Institute for Administration / computer system. *Journal of AL-Qadisiyah for computer science andmathematics*, *6*(1), 1–10.

Rubin, K. S. (2012). *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley.

Ruhe, G., Moussavi, M., Garousi, G., Smith, B., & Garousi, V. (2013). *Evaluating usage and quality of technical software documentation*. 24.

Sommerville, I. (2016). *Software engineering* (10. p.). Boston: Pearson, 2016.

Spinellis, D. (2010). Code Documentation. *IEEE Software*, *27*(4), 175–178.

Sprague, R. H. (2006). Electronic Document Management: Challenges and Opportunities for Information Systems Managers. *MIS Quarterly*, *19*(1), 29.

Stettina, C. J., Heijstek, W., & Fægri, T. E. (2012). Documentation work in agile teams: The role of documentation formalism in achieving a sustainable practice. *Proceedings - 2012 Agile Conference, Agile 2012*, 31–40.

Taylor, R. N., & Van Der Hoek, A. (2007). Software Design and Architecture: The once and future focus of software engineering. *FoSE 2007: Future of Software Engineering*, 226–243.

Tilley, S., & Parveen, T. (2012). On the similarities and differences between program documentation and test documentation. *IEEE International Professional Communication Conference*, 1–2.

Tuomi, J., & Sarajärvi, A. (2018). *Laadullinen tutkimus ja sisältöanalyysi* (Uudistettu). Helsinki: Kustannusosakeyhtiö Tammi.

Zahedi, M., Shahin, M., & Ali Babar, M. (2016). A systematic review of knowledge sharing challenges and practices in global software development. *International Journal of Information Management*, *36*(6), 995–1019.

LIITE 1 HAASTATTELUKYSYMYKSET

1. Tiedon tallennus: dokumenttien tuottaminen

- Minkälaista dokumentaatiota järjestelmäkehityksestä ja järjestelmän teknisestä toteutuksesta tuotat ja kulutat projekteissasi? (Monivalinta: Liite 2)
- Miten vaatimusmäärittelyjen tekniset tiedot ts. järjestelmävaatimukset dokumentoidaan?
    - Dokumentoidaanko ei-toiminnallisia järjestelmävaatimuksia?
- Onko järjestelmän kokonaisarkkitehtuurista tai sen osista dokumentaatiota? (esim. graafinen kuvaus)
    - Miten kuvaus on toteutettu?
    - Onko arkkitehtuuri kehittynyt tuotteen mukana vai määritelty etukäteen?
- Käytetäänkö koodiin upotettuja kommentteja dokumentaationa?
- Tuotetaanko testauksesta dokumentaatiota?
    - Jos kyllä; miten?
- Onko järjestelmän sisäiset prosessit (ts. järjestelmän kannalta oleelliset / / järjestelmän tukemat liiketoiminnan prosessit) dokumentoitu?
    - Jos kyllä; miten?
- Käy läpi kaikki listaamasi dokumentit: Miksi juuri tällaisia dokumentteja on päätetty tuottaa?
    - Millä tavoilla dokumentaatiot on tuotettu? (luonnollinen kieli, kuviot, kirjastot…)
    - Vastuualueet: Mitä dokumentaatiota tuotat itse ja mitä muu projektitiimi tuottavat?
- Mitkä dokumentit koet tärkeimmiksi ylläpitoa ja jatkokehitystä ajatellen? Olettaen, että toinen tiimi hoitaa nämä tehtävät
    - Miksi koet valitsemasi dokumentit tärkeimmiksi?
    - Tuotetaanko ylläpitoa ja jatkokehitystä varten vielä erikseen jotain dokumentaatiota

2. Tiedon kehitys ja päivitys

- Mitä dokumentteja on päivitetty edes kerran sen luomisen jälkeen? Mitä dokumentteja ei ole koskaan päivitetty?
    - Kuka on vastuussa päivityksestä?
- Millä aikavälillä eri dokumentteja päivitetään? (päivittäin, sprintin jälkeen, kvartaali…)
- Koetaanko jonkun dokumentin päivitys tärkeämmäksi kuin muiden?
    - Jos kyllä; minkä ja miksi?
- Päivitetäänkö jotain teknistä dokumenttia vain, jos se tuottaa suoraa hyötyä omiin työtehtäviin?

3. Tiedon jakaminen

- Miten dokumentaatio on saatavilla projektitiimin sisällä? Ts. Missä kanavissa dokumentit jaetaan?
- Koetko sinulle oleellisen tiedon olevan helposti löydettävissä?
    - Miksi?
- Koetko sinulle oleellisen tiedon olevan helposti ymmärrettävissä?
    - Miksi?

4. Tiedon käyttö

- Mitä teknistä dokumentaatiota käytät työssäsi projektiisi liittyen?
    - Miksi? Mitä hyötyä nämä dokumentit tarjoavat työllesi?
- Minkälaista dokumentaatiota tarvitset työssäsi, mutta sitä ei ole saatavilla tai se ei ole ajan tasalla?
    - Miksi tarvitset juuri tällaista dokumentaatiota?

5. Tiedon johtaminen

- Suunniteltiinko dokumentaation tuottamista tai hallintaa ennen projektia? Miten?
- Johdetaanko tai seurataanko dokumentaation prosessia jatkuvasti projektissa? Miten ja kenen toimesta?
- Mitkä koet suurimmiksi haasteiksi nykyisessä tavassa tuottaa tai ylläpitää dokumentaatiota?
- Tuleeko sidosryhmiltä tilauksia (ts. tarpeita) dokumentaation tuottamiseen liittyen?
    - Tuotetaanko jotain teknistä dokumentaatiota vain, koska on raportoitava sidosryhmille?
- Tilaatko itse dokumentaatiota projektitiimiltä, mikäli havaitset jollekin dokumentaatiolle tarvetta?

6. Agilen periaatteet dokumentaatiossa

- Tuotetaanko dokumentaatiota iteratiivisesti?
    - Vai kerääntyykö dokumenttien tuottaminen johonkin ajanjaksoon?
- Tuottaako kaikki kehitystiimin jäsenet dokumentaatiota?
    - Vai kerääntyykö dokumentaation tuottamisen tai päivittämisen vastuu tietyille avainhenkilöille?
- Varataanko dokumentaation tuottamiselle resursseja kehityssprintin aikana?

LIITE 2 TAUSTATIEDOT JA MONIVALINTA TEKNISISTÄ DOKUMENTEISTA

Ikä:
Nykyinen titteli:
Työkokemus tuotteenomistajan tehtävissä vuosina:
Koodin tuottaminen: Ei osaamista – Alkeet - Kohtalainen osaaminen – Ammatillinen osaaminen

Minkälaista dokumentaatiota järjestelmäkehityksestä ja järjestelmän teknisestä toteutuksesta tuotetaan projektissasi?

- Järjestelmän kuvaus
- Käyttöliittymäsuunnitelmat
- Käyttöliittymien määritykset
- Käyttäjäryhmien määrittely
- Vaatimusmäärittelyt
- Käyttäjätarinat (User Story)
- Käyttöskenaariot (Scenarios)
- Kokonaisarkkitehtuurin kuvaus
- Osa-arkkitehtuurin kuvaus (esim. mikropalveluiden arkkitehtuuri)
- Arkkitehtuurin rakenteeseen liittyvät päätökset
- Kehityksen aikana tehdyt päätökset järjestelmään ja sen toimintoihin liittyen
- Järjestelmän tukemien liiketoimintaprosessien kuvaukset
- Tietokannan määrittely
- Rajapinnan määrittely
- Algoritmien kuvaukset
- Datamalli
- Luokkakuvaukset
- Sprintin suunnitteludokumentti (Sprint planning)
- Sprintin arviointidokumentti (Sprint Review)
- Sprintin edistymiskäyrä (Burn down chart)
- Koodiin upotetut kommentit
- Testaussuunnitelma
- Testitapaukset
- Testimenetelmät
- Testausraportti
- Muita, mitä?
    -
    -
    -
    -