Tarja Lauhikari

Asiakkaan osallistuminen tietoturvan kehittämiseen hankittaessa vahvaa suojausta vaativia ohjelmistojärjestelmiä

Tietotekniikan pro gradu -tutkielma

17. heinäkuuta 2017

Jyväskylän yliopisto

Tietotekniikan laitos

Tekijä: Tarja Lauhikari
Yhteystiedot: tarja.lauhikari@gmail.com

Ohjaaja: Martti Lehto

Työn nimi: Asiakkaan osallistuminen tietoturvan kehittämiseen hankittaessa vahvaa suojausta vaativia ohjelmistojärjestelmiä

Title in English: Customer participation in developing information security with acquisition of software systems requiring strong protection

Työ: Pro gradu -tutkielma

Suuntautumisvaihtoehto: Ohjelmisto- ja tietoliikennetekniikka

Sivumäärä: 115+3

Avainsanat: Ohjelmistokehitysprosessi, ketterä kehitys, tietoturvavaatimukset, uhka-arviot, suojaustasot, auditointi

Keywords: Software development process, agile development, security requirements, threat assessment, security levels, auditing

Tiivistelmä: Digitalisaatio lisää kyberuhkien mahdollisuutta puolustusvoimien ohjelmistojärjestelmissä. Tietoturvaa täytyy jatkuvasti kehittää, jotta järjestelmien monipuolinen käyttö pysyisi turvallisena. Tutkimuksessa haastateltiin kymmentä asiantuntijaa. Tutkimuksen tarkoitus oli selvittää tietoturvaan vaikuttavat tekijät ja menetelmät järjestelmähankinnassa. Tutkittiin, mitä asiakkaan tulisi ottaa huomioon ohjelmistokehitysprosessissa, jotta se tukisi tietoturvan kehitystä. Lisäksi vaatimusmäärittelyä pohdittiin tietoturvan kannalta. Tärkeäksi aiheeksi nousi myös yhteistyö suunnittelijoiden, asiakkaan ja tietoturva-asiantuntijoiden välillä. Tutkimuksen mukaan asiakkaan tulisi osallistua tiiviisti ohjelmiston kehitysprosessiin. Kehitettävästä järjestelmästä tulisi tehdä uhka-arvio, sekä tietoturvavaatimuksissa tulisi ottaa huomioon käyttöympäristö, käytettävyys, ohjelmiston ympärillä oleva järjestelmäkokonaisuus ja rajapinnat. Lisäksi lakisääteiset ohjeistukset pitäisi tulkita suunnittelijalle ymmärrettävään muotoon.

Abstract: Digitalization increases the likelihood of cyber incidents in the software systems of the Defence Forces. Continuous information security development is necessary to ensure that a versatile use of the systems would be still secure. In this study, ten experts were interviewed. The purpose of the study was to identify factors and methods that would help develop security in software systems acquisition. It was studied what items a customer should take into account to support inclusion of security aspects in the software development process. In addition, requirement specification was discussed inform the security point of view. The collaboration between the customer, developers, and security experts became an important topic, too. According to the study, the customer should participate in software development process actively. In the requirements specification, environment, usability, and integrated system together with its interfaces should be considered. Also, a threat assessment should be done. The statutory instructions should be communicated to the developers in terms they understand.

Termiluettelo

CC	Common Criteria for Information Technology Security Evaluation, kansainvälinen tietoturvastandardi ISO/IEC 15408.
CBS	COTS -Based System, valmiisiin tuotteisiin perustuva järjestelmäkokonaisuus.
COTS	Commercial Off – the – Shelf, valmiina ostettava tuote.
MLU	Mid – Life Update, tuotteen suunnitellun käyttöajan varmistamiseksi tehtävä päivitys. Tehdään noin puolessa välissä tuotteen elinkaarta.

Kuviot

Kuvio 1. Järjestelmätekniikan vuorovaikutukset 14
Kuvio 2. Vesiputousmalli 16
Kuvio 3. Ketterä kehitysmalli 21
Kuvio 4. Scrum-menetelmän vaiheet 23
Kuvio 5. Sidosryhmien osallistuminen vaatimusmäärittelyyn 28
Kuvio 6. Ei-toiminnallisten vaatimusten vaikutukset vaatimusmäärittelyyn 33
Kuvio 7. Vaatimusten ja uhkien vuorovaikutus 36
Kuvio 8. Tietovuokaavio järjestelmästä 39
Kuvio 9. Ohjelmistokehitystiimin kokoonpano 41
Kuvio 10. Tiedot haastatteluun osallistuneista henkilöistä taulukoituna 52
Kuvio 11. Tutkimustuloksiin perustuva kehitysprosessimalli 100

Sisältö

1 JOHDANTO 1
1.1 Tutkimuksen tausta 1
1.2 Tutkimuksen tavoite ja aiheen merkitys 5
1.3 Tutkimustehtävä ja rajaus 5
1.4 Tutkimuksen rakenne 8

2 MÄÄRITELMIÄ 9
2.1 Auditointi 9
2.2 Katakri 9
2.3 Sidosryhmä 11
2.4 VAHTI-ohjeet 12
2.5 Suojaustaso 12

3 PROSESSIMALLIT 14
3.1 Vesiputousmalli 16
3.2 Inkrementaalinen malli 17
3.3 Evoluutiomalli 18
3.4 Spiraalimalli 18
3.5 Ketterä kehitys 20
3.5.1 Lean-ajattelu 21
3.5.2 Scrum 23

4 VAATIMUSMÄÄRITTELY 27
4.1 Vaatimukset yleisesti 27
4.2 Tietoturvavaatimukset 34
4.3 Uhka-arviot 35
4.4 Sidosryhmien osallistuminen 40

5 TUTKIMUKSEN TOTEUTUS 43
5.1 Tutkimusongelma 43
5.2 Tutkimuskysymykset 45
5.3 Taustaa tutkimusmenetelmän valinnalle 47
5.4 Aineiston hankinta 51
5.5 Aineiston analyysi 54

6 KOKEMUKSET TIETOTURVAN TOTEUTTAMISESTA 56
6.1 Kehitysprosessi 57
6.1.1 Vaatimusmäärittelyn aloitus 58
6.1.2 Asiakkaan osallistuminen 61
6.1.3 Auditoinnit 64
6.1.4 Tietoturvan todentaminen ja testaus 68
6.1.5 Muutokset tietoturvavaatimuksissa 70

6.1.6 Tietoturva ja tuotteen elinkaari ... 70
6.2 Vaatimusmäärittely ... 72
6.2.1 Perusteet tietoturvavaatimuksille ... 73
6.2.2 Tietoturva ja kokonaisuus ... 75
6.2.3 Uhka-arviot ... 78
6.2.4 Katakri ja VAHTI-ohjeet ... 80
6.2.5 Käytettävyys, toiminnalliset vaatimukset ja tietoturva ... 82
6.2.6 Suojaustason määritys ... 85
6.2.7 Käyttöympäristö ja järjestelmien integraatio ... 86
6.2.8 Vaatimusten tarkkuudesta ... 89
6.3 Kollaboraatio ja asiantuntijoiden käyttö ... 91
6.3.1 Asiantuntija-apu ... 91
6.3.2 Yhteistyö ... 93
6.3.3 Osaamisen kehittäminen ... 94
6.4 Kehitysideoita ... 95

7 JOHTOPÄÄTÖKSET ... 97
7.1 Hankintaprosessin kehittäminen ... 97
7.2 Tietoturvavaatimusmäärittely ... 100
7.3 Yhteistyö sidosryhmien välillä ... 102

8 YHTEENVETO JA POHDINTA ... 104

LÄHTEET ... 107

LIITTEET ... 116
A Käsittelyvaatimuksia osoittavat suojaustasot ... 116
B Haastattelupyyntö ... 117
C Koodiluettelo ja koodien toistuvuus ... 118

1 Johdanto

Digitalisaatio ja järjestelmien verkottuminen lisäävät kyberuhkien mahdollisuutta puolustusvoimien ohjelmistojärjestelmissä. Digitalisoinnin avulla järjestelmiin saadaan lisää ominaisuuksia, joten niiden käyttö monipuolistuu. Samalla järjestelmien integrointimahdollisuudet lisääntyvät ja ominaisuuksia voidaan liittää osaksi toisia järjestelmiä. Laajasti integroitu, monipuolinen järjestelmä lisää väärinkäytösten mahdollisuutta. Jotta järjestelmien tietoturvallisuus pysyisi vaaditulla tasolla, järjestelmien tietoturvaominaisuuksien tulisi kehittyä järjestelmien kehityksen mukana. Vahvan tietoturvan kehittäminen tulisi ottaa huomioon jo ohjelmistojärjestelmien hankinnan ja kehitystyön valmisteluista lähtien.

1.1 Tutkimuksen tausta

Ohjelmistojärjestelmät hankitaan alan yrityksistä puolustusvoimien hankintaprosessin ohjeistuksen mukaisesti. Kaikkiin hankintoihin on käytössä sama ohjeistus hankinnan koosta, tuotteen ominaisuuksista tai kypsyydestä riippumatta. Samaa ohjeistusta noudatetaan, olipa kyseessä varusmiehille hankittavat kumisaappaat tai laajasti verkottunut puolustusvoimien tarpeisiin kehitettävä ohjelmistojärjestelmä. Hankinta koostuu hankintaohjeiden mukaan toinen toistaan seuraavista vaiheista ja vaiheesta toiseen siirtyminen edellyttää hyväksytysti suoritettua katselmointia. Hankinnan aikana mahdollisesti tarvittavaan järjestelmäkehitykseen hankintaprosessi ei ota kantaa, mutta toimittajan kanssa tehtävät hankintasopimukset kiinnittävät prosessin asiakkaan kannalta vesiputousmallin kaltaiseen menettelyyn. Tärkein yksittäinen tekijä prosessissa on, että hankinnat tehdään pääsääntöisesti kilpailutuksena, ja siten järjestelmää koskevat vaatimukset on tehty ennen toimittajan valintaa ja hankintasopimusta.

Tietoturvavaatimusten tiukka määrittäminen ennen kehitystyön aloitusta voi aiheuttaa ongelmia järjestelmäsuunnittelussa. Koska tietoturva rakennetaan ohjelmistoon muun kehitystyön rinnalla, vaatimuksia joudutaan määrittämään ja muokkaamaan työn edetessä. Lisäksi monilla yrityksillä on käytössä ketterät ohjelmistokehitysmenetelmät, jolloin

vaatimusmäärittely halutaan mahdollisimman joustavaksi ja tietoturvaratkaisuja suunnitellaan kehitystyön aikana. Ketterissä ohjelmistokehitysmenetelmissä vaatimusten toteutus ja todentaminen tehdään toistuvissa sykleissä, iteraatioissa (Schwaber & Sutherland, 2016). Asiakkaan toivotaan osallistuvan iteraatioihin mahdollisimman aktiivisesti, jotta kehitettävän tuotteen ominaisuuksien ja asiakkaan odotusten välinen ero ei kasva liian suureksi. Mikäli kehitystyön aikana todetaan, että tuotteen ominaisuudet eivät ole riittäviä, niitä voidaan joustavasti muuttaa kehitystyön edetessä (Deuff & Cosquer, 2013). Vesiputousmallin mukaisessa menettelyssä tuotteen todentaminen tehdään kokonaisuudessaan vasta tuotteen valmistumisen jälkeen. Tämä aiheuttaa muutostöitä valmiiseen tuotteeseen ja kustannukset kasvavat (Sommerville, 1996).

Kirjallisuudessa tietoturvavaatimukset kuuluvat ei-toiminnallisiin vaatimuksiin, joilla on vaikutuksia järjestelmän laatuattribuutteihin (Wiegers & Beatty, 2013). Suunnittelu tehdään toiminnallisten vaatimusten perusteella, joten käyttäjävaatimukset ja järjestelmän arkkitehtuuri vaikuttavat siihen, miten ei-toiminnallisia vaatimuksia pystytään toteuttamaan (Schmidt, 2013). Tutkimuksissa ei-toiminnallisten vaatimusten määrittäminen tehdään hyvin harvoin, jos koskaan asiakkaan tai käyttäjien toimesta. Kuitenkin vahvaa suojausta vaativissa järjestelmissä tietoturvaratkaisut vaikuttavat järjestelmän muihin ominaisuuksiin ja jopa arkkitehtuurin, mikä täytyy ottaa huomioon alusta lähtien järjestelmän kehityksessä.

Tietoturvan kehittäminen nähdään enimmäkseen ohjelmistosuunnittelijoiden työnä (Souag, Mazo, Salinesi, & Comyn-Wattiau, 2016), eikä ylemmän tason käyttäjä- tai asiakasvaatimuksissa käsitellä tietoturvaa juurikaan. Kun ohjelmistoon tarvitaan vahvaa suojausta, suojaustason määrittäminen tulisi olla jo liiketoiminta- tai käyttäjävaatimuksissa. Tällöin ohjelmistosuunnittelijat tietäisivät halutun suojaustason jo ennen kuin tuotetta lähdetään kehittämään. Tutkimuksissa ohjelmistosuunnittelijoille on kehitetty monia menetelmiä ohjelmistoon kohdistuvien uhkien ja tietoturvaominaisuuksien mallintamiseen. Mallinnusta tehdään ennen kaikkea ohjelmiston sisäisistä vuorovaikutuksista ja ominaisuuksista (Souag ym., 2016). Toisaalta tutkimuksissa on hyvin vähän käsitelty, kuinka suunnittelija pystyy tietoturvaratkaisuissa ottamaan huomioon tuotteen aiotun toiminta- ja käyttöympäristön. Viranomaiskäyttöön kehitettävien järjestelmien

tietoturvavaatimukset perustuvat lakiin ja sitä noudattaviin yleisiin tietoturvaohjeisiin. Tutkimuksissa ei ole juurikaan käsitelty, kuinka näitä yleisiä ohjeita ja lakia tulkitaan yksittäisiin ohjelmistoihin soveltuviksi.

Järjestelmien tietoturvaa on määritetty useissa kansallisissa ja kansainvälisissä ohjeissa. Suomessa viranomaiskäytössä on Kansallinen Tietoturvallisuuden auditointityökalu viranomaisille (Katakri) sekä tietoturvan ohjeistaminen lukuisissa valtionhallinnon tieto- ja kyberturvallisuuden johtoryhmän (VAHTI) tietoturvaohjeissa. Vaikka ohjeistusta on kirjoitettu paljon, niiden tulkinta ja ohjeiden mukainen järjestelmien toteutus eivät ole toteutuneet käytännössä. Tietoturvaohjeet eivät ota kantaa itse järjestelmään, vaan vaatimukset on kirjattu siten, että ne ovat sopivia kaikkiin järjestelmiin. Tästä johtuen vaatimukset eivät siis sovi hyvin mihinkään järjestelmään. Yleisistä ohjeista johdettujen vaatimusten tekeminen on työlästä, koska tulkinnasta on hyvin vähän puolustusvoimien järjestelmiin soveltuvia esimerkkejä. Lisäksi ohjeiden tulkinta vaatii tietoturvaosaamista ja tietoturvan kannalta järjestelmäkokonaisuuden huomioon ottamista. Monesti tulkinnat jäävät yksittäisten henkilöiden hengentuotteiksi, jotka eivät välttämättä vastaa ohjeiden tarkoituksiin.

Tietoturvavaatimukset suunnitellaan usein erillään käyttäjien suunnittelemista toiminnallisista vaatimuksista, joten ne huonontavat järjestelmän käyttöä ja hidastavat käytettävyyttä. Esimerkiksi järjestelmän toimintakuntoon asettaminen pitkittyy käyttäjätunnusten ja salasanojen syötössä tai käyttäjien henkilökohtaiset tunnukset hidastavat useamman henkilön järjestelmän käyttöä. Kun yleisistä ohjeista pyritään tekemään suoraan vaatimukset yksittäiseen järjestelmään, käytettävyys heikkenee, jolloin tietoturvan ja käytettävyyden välillä tehdään kompromisseja suunnittelussa tai viimeistään järjestelmän käytössä.

Puolustusvoimilla on käytössä omia tietoturva-asiantuntijoita. Heidän aika ei kuitenkaan riitä järjestelmien kehitystyöhön, koska tietoturvaosaamista tarvitaan laajasti puolustusvoimien eri tehtävissä. Jotta tuotteiden tietoturvaratkaisut täyttäisivät asetetut vaatimukset, asiantuntija-apua on hankittu tietoturva-alan yrityksistä. Tietoturva-asiantuntijoita on mukana järjestelmien vaatimusmäärittelyissä, tuotteen kehitystyössä

miettimässä sopivia tietoturvaratkaisuja sekä tekemässä auditointeja. Mikäli auditoijalle ei anneta kohdejärjestelmästä kattavaa tietoa, hän tekee auditoinnin yleisten Katakri- ja VAHTI-ohjeiden perusteella. Tällöin auditoinnissa löytyneet haavoittuvuudet eivät välttämättä vastaa järjestelmän käytössä esille tulevia haavoittuvuuksia. Toisaalta kaiken kattava auditointi vie aikaa ja maksaa paljon, joten auditointia tulisi kehittää siten, että siitä saataisiin paremmin hyötyä ohjelmistojen tietoturvakehitykseen.

Koska järjestelmät ovat integroituneet suureksi järjestelmäkokonaisuudeksi, sen hallinta vaatii suunnittelua. Mikäli järjestelmiä kehitetään yksitellen tai ominaisuuksia lisätään pikkuhiljaa, kokonaisuutta pitäisi kehittää osajärjestelmien kehityksen mukaan. Mikäli kehitystyö jää järjestelmätasolle, esimerkiksi yhteiskäyttöisten ominaisuuksien hallinnointi on vaikeaa. Tietoturvaratkaisuissa useat palvelut voitaisiin toteuttaa osana järjestelmäkokonaisuutta, jolloin samoja tietoturvaratkaisuja ei tarvitsisi kehittää jokaiseen järjestelmään erikseen.

Asiakkaan puolelta järjestelmän tietoturva havaitaan puutteelliseksi tavallisesti vasta järjestelmän vastaanottohyväksyntöjen yhteydessä. Silloin ohjelmistoille tehdään kattava testaus ja myös tietoturva-auditointi. Mikäli tietoturvassa todetaan puutteita, järjestelmää ei voida hyväksyä käyttöön. Tämä aiheuttaa tietoturvan räätälöintiä tuotteeseen jälkeenpäin, mikä ei tavallisesti johda parhaisiin mahdollisiin tietoturvaratkaisuihin.

Koska tietoturvan toteutus vaatii monialaista asiantuntemusta, yhteistyö on välttämätöntä. Jotta kaikki tarvittava tieto saataisiin kerättyä kokoon järjestelmän kehitystyötä varten, yhteistyön täytyy toimia saumattomasti ja eri osapuolten tulisi osallistua kehitystyöhön aktiivisesti. Nykyisin käytössä olevat ohjelmistokehitysprosessit, kuten ketterät menetelmät korostavat asiakkaan säännöllistä osallistumista kehitystyön eri vaiheisiin. Toisaalta niissä ei ole otettu kantaa eri sidosryhmien toimintaan, vaan se ikään kuin muodostuu kehitysprojektin ympärille. Kuitenkin monien eri sidosryhmien yhteistoiminta vaatii suunnittelua ja koordinointia, jotta se hyödyttäisi kehitystyötä ja edistäisi tietoturvan toteutumista järjestelmissä.

1.2 Tutkimuksen tavoite ja aiheen merkitys

Tutkimuksen tarkoituksena oli löytää tekijöitä puolustusvoimien ohjelmistojärjestelmien hankintatoiminnan kehittämiseen. Tietoturva on noussut erityisen tärkeäksi aiheeksi ohjelmistojärjestelmien kehitystyössä ja sen huomioiminen jälkeenpäin aiheuttaa lisäkustannuksia ja käyttöönotoissa viiveitä. Tutkimuksen tavoitteena oli saada selville tietoturvan kehittämiseen vaikuttavat tekijät järjestelmätoimittajien näkökulmasta. Tarkoituksena oli, että asiakas kuuntelisi toimittajien ehdotuksia siitä, kuinka hän pystyisi edistämään toimittajien työtä. Puolustusvoimien ohjelmistojärjestelmien tietoturvan kehitykseen tarvitaan laaja-alaista osaamista, joten ei voida olettaa, että toimittaja pystyisi hallitsemaan kaikki erityispiirteet, mitä siinä tulee ottaa huomioon. Näin ollen halutaan pienentää epäonnistumisten riskiä, selvittämällä mahdollisia puutteita asiakkaan toiminnassa ja etsimällä kehityskohteita tietoturvallisten ohjelmistojen kehitykseen.

Mikäli kehitysprojektit saataisiin valmiiksi suunnitelluissa aikatauluissa ja ohjelmistojärjestelmien tietoturva olisi hyväksyttävällä tasolla, lisäkustannuksilta säästyttäisiin. Lisäksi tietoturvan rakentamista erikseen järjestelmien ympärille ei tarvittaisi, jolloin tietoturva-asiantuntijoiden ja kehitystyössä olevien henkilöiden työtä säästyisi huomattavasti. Ohjelmistojärjestelmien hyvin toteutettu vahva tietoturvasuojaus antaisi myös käyttäjille monipuolisemmat mahdollisuudet järjestelmien käyttöön.

1.3 Tutkimustehtävä ja rajaus

Tutkimus tehtiin puolustusvoimille. Tutkimusongelmana oli selvittää, miten puolustusvoimien asiakkaana tulisi osallistua tietoturvan kehittämiseen vahvaa suojausta vaativiin ohjelmistoihin. Tutkimus toteutettiin laadullisena tutkimuksena, jossa aineisto kerättiin haastattelemalla kymmentä ohjelmistosuunnittelun parissa toimivaa henkilöä. Tarkoituksena oli henkilöiden kanssa kasvotusten keskustelemalla saada monipuolisempaa ja syvällisempää tietoa kuin vaikka verkossa täytettävällä kyselytutkimuksella.

Lähtökohtana tutkimukselle on kuvata tutkittava ilmiö totuudenmukaisesti ja kattavasti (Hirsjärvi, Remes, & Sajavaara, 2004). Tietoturvaan vaikuttavia tekijöitä

kehitysprosessissa sekä vaatimusmäärittelyssä tutkittiin. Lisäksi eri sidosryhmien välisen yhteistyön onnistumiseen etsittiin keinoja. Laadullinen tutkimus kohdistuu pieneen joukkoon, jota tutkitaan mahdollisimman kattavasti (Eskola & Suoranta, 1998). Yritykset valitsivat tutkimukseen henkilöt heidän tietoturvaosaamisensa perusteella. Haastattelumenetelmäksi valittiin teemahaastattelu. Vaikka haastatteluissa tuli esille asioita, joita asiakkaan tulisi pohtia toiminnassaan, yritykset olivat pääsääntöisesti tyytyväisiä asiakkaaseen. Tutkimuksessa tuli esille myös epäkohtia. Niiden käsittelyn toivotaan kehittävän yhteistoimintaa, eikä niitä tule käyttää yrityksiä vastaan yhteistyötä tehtäessä. Haastattelutilanteissa korostettiin aitoutta ja rehellisyyttä, jotta haastattelujen avulla voitaisiin löytää hyviä kehityskohteita. Tutkimusongelmaa pohdittiin seuraavien tutkimuskysymysten avulla:

Miten asiakkaan hankintaprosessia tulisi kehittää, jotta se tukisi vahvaa suojausta vaativien ohjelmistojärjestelmien kehitystä?

Puolustusvoimien hankinnoissa asiakkaan vastuulla on kirjoittaa vaatimukset niin hyvin, että toimittaja pystyy toimittamaan ja ohjelmistotuotteiden osalta monesti tekemäänkin tuotteen niiden perusteella. Hankintasopimukseen kuuluu vaatimusmäärittely ja käyttötapaukset. Tuotteen kehityksen aikana asiakkaan rooli on seurata kehitystyön edistymistä. Asiakkaan aktiivinen osallistuminen kehitystyöhön vaihtelee hyvin paljon kehitysprojektien välillä. Siitä ei ole mitään yleistä käytäntöä tai ohjeistusta. Mikäli kehitysprojektin aikana ilmenee tarvetta tehdä sopimusmuutoksia, niitä tehdään perustelluista syystä ja se edellyttää asiakkaan ja toimittajan molemmin puolisen hyväksynnän. Selkeä rooli asiakkaalla on järjestelmän vastaanotoissa. Vastaanotto on voitu jakaa useampaan osaan osatoimituksiksi, joissa järjestelmän ominaisuudet testataan ja hyväksytään. Lopullisessa vastaanotossa kaikki toiminnallisuudet ovat käytössä. Tietoturvan toteutuminen tarkistetaan asiakkaan puolelta vasta tietoturva-auditoinnissa. Tietoturva-auditoinnit voidaan sopia hankintasopimuksessa tai ne voidaan tehdä vasta koko tuotteen toimituksen jälkeen, jolloin toimittaja ei välttämättä tiedä koko auditoinnista mitään. Tietoturvan toteutumista on pyritty parantamaan tekemällä auditointeja yhdessä toimittajan ja asiakkaan kanssa jakamalla auditoinnit useampaan osaan, jolloin suunnittelijat saavat tietoa hyväksyttävistä tietoturvaratkaisuista jo kehitystyön aikana.

Mitä vahvaa suojausta vaativien ohjelmistojärjestelmien tietoturvavaatimuksissa tulisi ottaa huomioon, jotta kehitettävien järjestelmien tietoturva täyttäisi lakisäätäiset vaatimukset?

Vaatimusmäärittelyssä vaatimukset tyypillisesti kootaan listaksi, jossa tietoturvavaatimukset ovat yksi oma kokonaisuus. Tietoturvavaatimukset ovat hyvin yleisellä tasolla ja asiakkaan puolelta ei ole mietitty, kuinka ne vaikuttavat käyttäjien antamiin vaatimuksiin tai käyttötapauksiin. Järjestelmän kehitysvaiheen loppupuolella tehtyjen auditointien perusteella järjestelmiin joudutaan tekemään tietoturvan parantamiseksi päivityksiä jo ennen kuin järjestelmä otetaan käyttöön. Ongelmana on yleensä, että toimittaja ei ole tiennyt kaikkia tietoturvaominaisuuksia, joita järjestelmän suunnittelussa olisi pitänyt ottaa huomioon. Tyypillisesti projekteissa keskitytään järjestelmän toiminnallisuuksiin, jolloin tietoturvaominaisuudet jäävät pienemmälle huomiolle. Jos vaatimukset tulevat mukaan suunnitteluun silloin, kun suunnittelussa on tehty jo teknisiä ratkaisuja, ne eivät välttämättä mahdollista kaikkien tarvittavien tietoturvaominaisuuksien toteuttamista. Huonoimmassa tapauksessa voi olla myös niin, että järjestelmässä käsiteltävän tiedon suojaustasoa ei ole mietitty perusteellisesti ennen järjestelmän suunnittelun aloitusta, jolloin suojaustaso ja sitä kautta tietoturvavaatimukset voivat muuttua olennaisesti järjestelmän kehitystyön aikana. Puolustusvoimien nykyisissä hankintaohjeissa on mainittu tietoturvavaatimukset, mutta ne eivät sisällä ohjeita tietoturvavaatimusten määrittämiseen.

Miten eri sidosryhmien välistä yhteistyötä tulisi kehittää, jotta se edistäisi tietoturvan rakentamista ohjelmistosuunnittelussa?

Puolustusvoimilla on käytössä hankintaprosessi, joka säätelee järjestelmien hankintoja ja lisäksi myös hankittavien järjestelmien kehitystyötä. Toimittaja voi periaatteessa käyttää haluamaansa kehitysprosessia, mutta asiakkaan osallistuminen järjestelmän kehitystyöhön voi vaihdella ja siitä ei välttämättä ole sovittu mitään hankintasopimuksessa. Puolustusvoimat käyttävät ulkopuolisia tietoturva-asiantuntijoita ohjelmistojen auditoinneissa ja myös toimittajien apuna tietoturvaratkaisuja mietittäessä. Toimittajien käyttämät ohjelmistokehitysmallit painottavat kaikkien osapuolten osallistumista tiiviisti

kehitystyöhön, jotta suunnittelijoilla olisi kehitystyön aikana mahdollisuus tarkentaa vaatimuksia tai tehdä vaikkapa vaatimuksesta poikkeavia ratkaisuja. Tämä on hyvin erilainen lähestymistapa verrattuna puolustusvoimien tarkasti ennalta suunniteltuun ja sovittuun lähestymistapaan.

1.4 Tutkimuksen rakenne

Tutkimus jakaantuu kahdeksaan lukuun. Luvussa kaksi esitellään tutkimuksessa käytettävät määritelmät. Luvussa kolme tehdään katsaus ohjelmisto- ja järjestelmäkehitysmalleihin. Prosessimallivaihtoehtoja pyritään tarkastelemaan kattavasti. Myös mallien hyviä ja huonoja puolia sekä tietoturvan ja vaatimusmäärittelyn kannalta rajoittavia tekijöitä tarkastellaan. Luvussa neljä esitellään vaatimusmäärittely yleisesti sekä tietoturvavaatimusmäärittely omana kokonaisuutenaan. Tietoturvaa tarkastellaan uhka-arvioiden avulla, joten uhkien mallintaminen käydään läpi lyhyesti. Laajassa ohjelmistohankinnassa sidosryhmien yhteistyö korostuu, siitä esitellään systemaattinen lähestymistapa.

Luku viisi sisältää tutkimuksen toteutuksen. Tutkimusongelman ja siihen liittyvien tutkimuskysymysten lisäksi perustellaan käytetty tutkimusmenetelmä. Aineiston hankinta ja sen analyysin toteutus esitellään kattavasti. Luvussa kuusi käydään läpi tutkimuskysymysten perusteella saadut tulokset. Luku seitsemän esittelee tutkimuksesta tehdyt johtopäätökset ja tulosten perusteella kehitetty kehitysprosessimalli. Luvussa kahdeksan tehdään yhteenveto tutkimuksesta. Samalla pohdintaan tutkimuksen käytettävyyttä ja mahdollisia jatkokehitysaiheita.

2 Määritelmiä

Vahvaa suojausta vaativien ohjelmistojärjestelmien kehitykseen kuuluu käsitteitä, jotka poikkeavat jonkin verran yleisestä ohjelmistokehityksestä. Tässä luvussa määritellään lyhyesti, mitä nämä käsitteet tarkoittavat. Käsitteillä voi olla erilaisia merkityksiä eri yhteyksissä, joten määrityksissä pyritään tuomaan esille tämän tutkimustyön kannalta olennaiset asiat.

2.1 Auditointi

Tietoturvallisuuden arviointi on lakisääteistä toimintaa silloin, kun tietoturva liittyy Suomea koskeviin kansainvälisiin velvoitteisiin. Arvio tietoturvasta ja tarvittaessa hyväksyntä voidaan kuitenkin hakea esimerkiksi valtionhallinnon alaa koskevien ohjeistusten tai viranomaisen riskien arvioinnin perusteella.

Auditointeja tekevät yritykset voivat hakea Viestintäviraston hyväksynnän toiminnalle. Arviointeja ei tehdä luvan tai ilmoituksenvaraisesti, mutta viranomaiset voivat käyttää ulkoiseen arviointiin vain Viestintävirastoa, ja sen hyväksymiä ulkoisia yrityksiä. Tietoturvapalveluja tarjoavat yritykset voivat tehdä sisäisiä arviointeja valtionhallinnossa sisäiseen toimintaan ja järjestelmiin ilman Viestintäviraston hyväksyntää.

Arviointimenettelystä säädetään laissa (Laki viranomaisten tietojärjestelmien ja tietoliikennejärjestelyjen tietoturvallisuuden arvioinnista 1406/2011). Lakiin liittyvien asetusten mukaisten vaatimusten todentamisessa käytetään VAHTI-ohjeistusta. Arvioinnissa tulee selvittää, täyttyvätkö kohteessa arviointiperusteeksi määritetyt tietoturvallisuuden vaatimukset. (Heiskanen & VM-julkaisutiimi, 2014)

2.2 Katakri

Kansallinen turvallisuusauditointikriteeristö (Katakri) on viranomaisten salassa pidettävän tiedon suojaamiseen tarkoitettu auditointityökalu. Kriteeristö määrittää tarvittavat suojausmekanismit organisaatioihin, joissa käsitellään viranomaisten salassa pidettävää

tietoa. Katakrin vaatimukset perustuvat kansalliseen lainsäädäntöön ja kansainvälisiin tietoturvallisuusvelvoitteisiin, joihin Suomi on sitoutunut. Valtioneuvoston laatima asetus tietoturvallisuudesta valtionhallinnossa (681/2010) on kansalliseen lainsäädäntöön perustuva keskeinen lähde. Sitä noudatetaan Suomessa sekä kansainvälisen että kansallisen salassa pidettävän tiedon suojaamisessa. Kansainväliset velvoitteet tulevat Euroopan unionin neuvoston turvallisuussäännöistä (2013/488/EU), jotka sisältävät EU:n neuvoston turvallisuusluokitellun tiedon suojaamista koskevat perusperiaatteet ja vähimmäisvaatimukset. (Puolustusministeriö, 2015)

Katakri koostuu kolmesta eri osa-alueesta: turvallisuusjohtaminen, fyysinen turvallisuus ja tekninen tietoturvallisuus. Turvallisuusjohtamisen vaatimuksilla organisaatioita pyritään ohjeistamaan riittävän turvallisuuskyvykkyyden ja -valmiuksien hankkimisessa. Turvallisuusjohtamiseen kuuluvat henkilöstön ja hallinnollisen turvallisuuden johtaminen. Fyysisen turvallisuuden vaatimukset kohdistuvat tiloihin ja rakennuksiin, joissa käsitellään salassa pidettävää tietoa. Teknisen tietoturvallisuuden osa-alueessa kuvataan vaatimukset, jotka kohdistuvat tekniseen tietojenkäsittely-ympäristöön eli tietojärjestelmiin ja useiden järjestelmien muodostamiin järjestelmäkokonaisuuksiin. Vaatimukset pyrkivät varmistamaan viranomaisten salassa pidettävän tiedon turvallisuusjärjestelyjen riittävyyden sähköisissä tietojärjestelmissä. Vaatimukset jakautuvat tiedon suojaustarpeen perusteella kolmeen eri suojaustasoon. Suojaustasot on määritetty liitteessä A. Katakrin mukaan vaatimuksilla ei ole tarkoitus lukita toteutustapaa, vaan niihin voi käyttää erilaisia toteutusvaihtoehtoja. Toteutustavoista on kuitenkin esimerkkikuvauksia, jotka perustuvat VAHTI-ohjeisiin sekä EU:n turvallisuussääntöjä täydentäviin suuntaviivoihin ja ohjeasiakirjoihin. (Puolustusministeriö, 2015)

Katakrin ajatuksena on, että tiedon suojaamisessa sekä turvallisuusratkaisujen suunnittelussa ja toteutuksessa saavutettaisiin uhkiin nähden riittävä turvallisuustaso. Kohdeorganisaatioille tehdään auditointeja, joissa organisaation on pystyttävä osoittamaan, että heidän rakentamansa turvallisuustaso vastaa Katakrin vaatimuksia. Aikaisemmista Katakrin versioista poiketen Katakri 2015 -versiossa turvallisuutta rakennetaan riskiarvioinnin avulla. Jotta vaatimusten tulkinta olisi tarkoituksenmukaista, sen tulisi pohjautua organisaation riskiarviointiin. Katakrin mukaan turvallisuusriskien hallinnalla

saadaan turvatoimien yhdistelmä, joka auttaa tasapanon löytämiseen kustannusten, käyttäjien vaatimusten sekä turvallisuuden jäännösriskin välillä. (Puolustusministeriö, 2015)

Katakria käytetään vaatimuksena julkisten hankintojen tietoturvan takaamiseksi, vaikka Katakrissa erityisesti mainitaan, että sitä ei ole tarkoitettu käytettäväksi suoraan julkisissa hankinnoissa. Tietoturvavaatimukset tulee laatia hankintoihin kyseisen hankinnan erityispiirteet ja riskit huomioon ottaen. Katakrin mukaan yksittäiset hankkeet tai hankinnat voivat sisältää muitakin kuin Katakrissa mainittuja suojausvaatimuksia. Ne tulee ottaa mukaan hankkeen sopimukseen, jotta kohdeorganisaatio sitoutuu noudattamaan myös niitä vaatimuksia. Katakrin ulkopuolisten vaatimusten toteutumista ei arvioida Katakrin avulla. (Puolustusministeriö, 2015)

2.3 Sidosryhmä

Vaatimusmäärittelyssä käytetään paljon termiä sidosryhmä (engl. stakeholder). Sillä tarkoitetaan yksittäistä henkilöä, ihmisryhmää, organisaatiota tai muuta kokonaisuutta, jolla on suora tai epäsuora kiinnostus kehitettävään järjestelmään. Kiinnostus järjestelmään voi tulla sen käytöstä, hyöty- tai haittavaikutuksista esimerkiksi kustannusten suhteen, järjestelmään kohdistuvasta vastuusta tai muista järjestelmän aiheuttamista vaikutuksista. Sidosryhmillä on oikeus antaa vaatimuksia (Hull, Jackson, & Dick, 2011, s. 7).

Ohjelmistotekniikassa sidosryhmät määritellään henkilöiksi tai organisaatioiksi, joihin järjestelmä tulee vaikuttamaan. Lisäksi sidosryhmillä on suoria tai epäsuoria vaikutuksia järjestelmävaatimuksiin, ohjelmistoartefakteihin sekä prosessissa tuotettuun, muokattuun tai käytettyyn informaation osaan (Kotonya & Sommerville, 1998). Määritelmä ei yksiselitteisesti kuvaa sidosryhmiä, jotka ovat yhteyksissä itse ohjelmistoprosessimallinnukseen (Bai, Huang, & Zhang, 2010).

Sidosryhmä voidaan määritellä myös toimijoiden merkitysten avulla. Siten sidosryhmä voi olla kuka tahansa henkilö, jonka mielipiteillä, tarpeilla tai mieltymyksillä on merkitystä projektin onnistumiseen. Itsestään selvä esimerkki on asiakas. Jos halutaan jonkun henkilön ostavan tuotteen, hänen mielipiteellään on merkitystä. On kuitenkin tärkeää

erottaa hienoinen ero todellisen maksajan ja ensisijaisen käyttäjän välillä. Sidosryhmä termillä on kolme eri merkitystä riippuen siitä, missä yhteydessä sitä käytetään. Niitä ovat sidosryhmäluokka, yksittäinen henkilö tai sidosryhmän edustaja. Sidosryhmäluokka on ryhmä, luokka tai henkilötyyppi tietyn aiheen käsittelyssä. Yksittäinen nimetty henkilö kuuluu yhteen tai useampaan sidosryhmäluokkaan. Samasta luokasta voidaan tarvita useampia yksittäisiä henkilöitä. Sidosryhmän edustaja on yksittäinen henkilö, joka edustaa sidosryhmäluokkaa projektissa. Joissain tapauksissa sidosryhmän edustaja ei kuulu edustamaansa sidosryhmäluokkaan, mutta toimii valtuutettuna edustajana, koska syystä tai toisesta kukaan luokan jäsenistä ei pysty edustamaan heitä. (Berenbach, Paulish, Kazmeier, & Rudorfer, 2009, ss. 140–141)

2.4 VAHTI-ohjeet

Valtionhallinnon tietoturvallisuuden johtoryhmä (VAHTI) on Valtiovarainministeriön alaisuuteen kuuluva elin, jonka tehtävänä on tietoturvallisuuden hallinnointi valtion tasolla. Tietoturvallisuuden osa-alueet ovat fyysinen, henkilöstö-, käyttö-, laitteisto-, ohjelmisto-, tietoaineisto- ja tietoliikenneturvallisuus sekä hallinnollinen tietoturvallisuus.

VAHTI-ohjeet sisältävät kaikki linjaukset ja ohjeet, mitä tietoturvaan liittyy valtionhallinnossa. VAHTI-ryhmän toiminta on parantanut tietoturvallisuutta valtion tasolla. Lisäksi siitä on positiivisia vaikutuksia yritysmaailmaan ja kansainväliseen toimintaan. (Hilve, Rousku, & Hummelholm, 2016)

2.5 Suojaustaso

Tieto tulisi luokitella organisaatioon vaikuttavan lisäarvon tai tiedon arkaluonteisuuden mukaan. Arkaluonteisuuden taso ja omaisuuden arvo määrittävät, kuinka paljon tietoturvaa täytyy käyttää tiedon suojaukseen. Luokittelun tulisi olla yksinkertainen ja helposti käytettävä. Suojattavaa tietoa pitäisi pystyä luokittelemaan tehokkaasti. Julkinen tieto täytyy luokitella julkiseksi, ja sen tulisi olla kaikkien saataville. Luottamuksellinen tieto jaetaan henkilöille, joilla on hyväksytty tarve tietoon. Tiedon omistaja määrittää, kenellä on oikeus saada luottamuksellista tietoa. (Raggad, 2010, ss. 7–9)

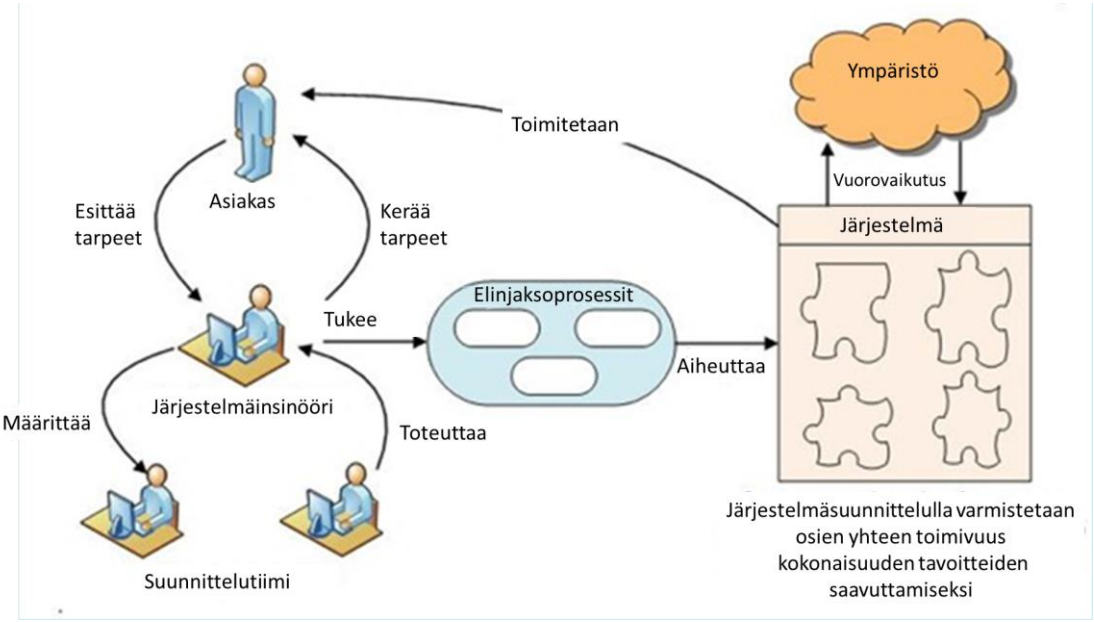
Standardi (ISO/IEC 27002: 2013) määrittelee tiedot seuraaviin suojaustasoihin: julkinen, sisäinen käyttö, tiedon omistusoikeus, erittäin luottamuksellinen ja huippusalainen. Huippusalainen tieto on kaikkein arkaluonteisinta. Sen joutuminen vääriin käsiin voi aiheuttaa katastrofaalisia seurauksia omistajalle. Esimerkki huippusalaiseen suojaustasoon kuuluvista tiedoista on kansallista turvallisuutta koskevat sotilaalliset tiedot. Yritysmaailmassa vastaavan tasoista tietoa ovat strategiset suunnitelmat, sopimustiedot ja tuotekehitysprosessit (Raggad, 2010).

Erittäin luottamuksellinen tieto on kriittistä organisaation toiminnan kannalta ja voisi vaikeuttaa liiketoiminnan jatkuvuutta. Menetykset ovat kovia, ja ne aiheuttavat yritykselle suuria kustannuksia. Tällaisia tietoja ovat kirjanpitotiedot, uudet liiketoimintasuunnitelmat, uudet tuotteet tai teknologiat. Talon sisällä olevat resurssit, laitteet, ohjelmat tai menetelmät ovat omistusoikeus -tietoa. Jos tämä tieto menetetään tai tulee julkiseksi, organisaation resurssitiedot tulevat myös julkiseksi tai ne menetetään. Sisäiseen käyttöön tarkoitettu tieto on edelleen luottamuksellista. Jos tällainen tieto paljastuu, se voi olla kiusallista organisaation johdolle. Riski taloudellisesta menetyksestä on kuitenkin hyvin pieni. Esimerkkejä sisäisestä tiedosta ovat yrityksen sisäinen kirjeenvaihto, muistiot ja toiminnan raportit. Julkinen tieto voidaan pitää julkisena ilman ei-toivottuja seurauksia. Esimerkiksi vapaapääsyisten Internet-sivujen tiedot, vuosiraportit ja mainokset kuuluvat julkiseen tietoon. (Raggad, 2010, ss. 7–9) Liitteessä A on Suomen valtionhallinnon määrittelemät suojaustasot.

3 Prosessimallit

Ohjelmistokehitysprosessi on joukko toimintoja, ja niihin liittyviä ohjelmiston kehittämiseen tarvittavia tietoja. Jokaisella organisaatiolla on oma erityinen ohjelmistoprosessi, mutta nämä yksittäiset menetelmät tavallisesti seuraavat jotain enemmän abstraktia yleistä prosessimallia (Sommerville, 1996).

Riippumatta käytettävästä prosessimallista, järjestelmäsuunnittelun päämäärä on tuottaa hyötyä sidosryhmille ja ennen kaikkea asiakkaalle. Järjestelmäsuunnitteluun osallistuu toimijoita, jotka vaihtavat tietoa keskenään. Suunnittelutyön tarkoitus on kehittää asiakkaalle toimitettava ohjelmisto, järjestelmä tai niiden kokonaisuus. Kuvassa (kuvio 1) on ylätason esitys järjestelmäsuunnittelun eri osa-alueista ja niiden keskinäisestä vuorovaikutuksesta.



Kuvio 1. Kaavio järjestelmäsuunnittelusta (SEBoK authors, 2016)

Järjestelmä- ja ohjelmistosuunnittelua käsitellään useissa kansainvälisissä standardeissa. Yksi niistä on ISO/IEC/IEEE TR 24748 (2016), joka käsittelee järjestelmä- ja ohjelmistosuunnittelua koko elinjaksohallinnan näkökulmasta. Se tuottaa standardien ISO/IEC/IEEE 15288:2015 ja ISO/IEC 12207:2008 kanssa yhtenäisen ja kiinteän ohjeistuksen järjestelmien elinjakson hallintaan. Tarkoituksena on auttaa projekteja

suunnittelemaan elinjaksomallin hallintaprosessia. Standardi (ISO/IEC/IEEE TR 24748: 2016) muodostuu seuraavista osista:

1. Ohjeistus elinjakson hallintaan (engl. Guide for life cycle management)
2. Ohjeistus ISO/IEC 15288 soveltamiseen (engl. Guide to the application of ISO/IEC 15288)
3. Ohjeistus ISO/IEC 12207 soveltamiseen (engl. Guide to the application of ISO/IEC 12207)
4. Järjestelmäkehityksen suunnittelu (engl. System engineering planning)
5. Ohjelmistokehityksen suunnittelu (engl. Software development planning)

Standardissa (ISO/IEC/IEEE 15288: 2015) kuvataan järjestelmäkehitysprosessit, kun taas (ISO/IEC 12207: 2008) standardi keskittyy yksityiskohtaisemmin ohjelmistokehitykseen. Järjestelmäkehityksessä (ISO/IEC/IEEE 15288: 2015) standardia käytetään laajasti. Sen suosio perustuu osaltaan yhteensopivuuteen ISO 9001 standardin kanssa. ISO/IEC/IEEE 15288:2015 esittelee järjestelmäkehityksen yleisen prosessirakenteen kattaen järjestelmän elinkaaren. Elinjakso lähtee ideoinnista jatkaen järjestelmän elinkaaren yli järjestelmästä luopumiseen asti. Standardin tarkoitus on luoda yleinen rakenne parantamaan viestintää sekä yhteistyötä järjestelmien kehityksessä, käytössä ja ylläpidossa. Näin osapuolet pystyvät työskentelemään integroidusti ja koherentisti prosessin eri vaiheissa (ISO/IEC/IEEE 15288: 2015).

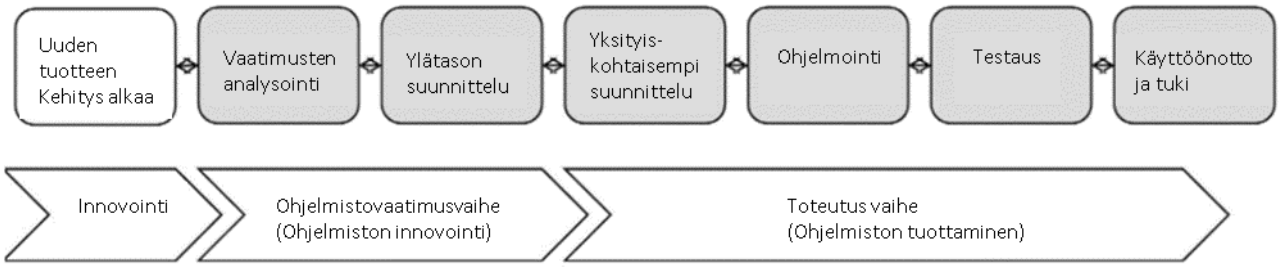
ISO/IEC 12207:2008 määrittelee elinjaksomallit ja niiden eri vaiheet. Vaiheet voivat mennä päällekkäin tai kertautua, mikäli se on tarpeellista projektin päämäärien, koon, monimutkaisuuden, muutostarpeiden tai vaihtoehtojen suhteen. Standardi ei vaadi minkään tietyn elinjaksomallin käyttöä, koska se tulisi jokaisen projektin määrittää itse. Esimerkkivaiheet standardissa ovat: konsepti, kehitys, tuottaminen, käyttö, tuki ja luopuminen. Ohjelmistotuotteen vaiheiksi on määritetty kehitys, operointi ja ylläpito. Standardissa on annettu esimerkkejä erityyppisistä elinjaksomalleista. Niitä ovat vesiputous-, inkrementaalinen, evoluutio- ja spiraalimallit. (ISO/IEC 12207: 2008)

3.1 Vesiputousmalli

Ensimmäinen elinjaksomalli, jota nykyisin kutsutaan vesiputousmalliksi, on kuvattu Roycen artikkelissa vuonna (1970). Vesiputousmallille tunnusomaista on, että ohjelmisto-suunnittelu muodostuu toinen toistaan seuraavista vaiheista. Vesiputousmallista on kehitetty monia eri versioita, mutta alkuperäisen mallin mukaan se koostuu seuraavista vaiheista:

1. Järjestelmävaatimukset (engl. System Requirements)
2. Ohjelmistovaatimukset (engl. Software Requirements)
3. Analyysi (engl. Analysis)
4. Suunnittelu (engl. Program Design)
5. Ohjelmointi (engl. Coding)
6. Testaus (engl. Testing)
7. Käyttöönotto (engl. Operations)

Kuvio 2 esittää vesiputousmallin mukaisia ohjelmistosuunnittelun vaiheita. Kehitys lähtee ohjelmiston innovoinnista sekä etenee vaihe vaiheelta päättyen käyttöönottoon ja järjestelmätukeen.



Kuvio 2. Vesiputousmalli (Lema, 2010, s. 40; Royce, 1970)

Vesiputousmallin perusajatus on, että eteneminen vaiheesta toiseen tapahtuu vasta, kun edellinen vaihe on valmis (Royce, 1970). Mallissa ei ole palautetta tai takaisinkytkentää vaiheiden välillä. Tätä on perusteltu sillä, että suunnittelun edetessä muutoksien tekemisen tulee pienentyä hallittuihin rajoihin. Määrittely-, suunnittelu- ja toteutusongelmat tulevat kuitenkin usein esille vasta toteutuksen aikana. Kun määrittely jäädytetään, sitä on vaikea muuttaa myöhemmin käyttäjän tarpeiden muuttuessa. Kuitenkin sidosryhmät kokevat

todellisten tarpeiden määrittämisen edeltä käsin vaikeaksi, joten sekä organisatoriset että loppukäyttäjän vaatimukset voivat muuttua kehitysprosessin aikana. Käytännössä mallin vaiheiden välillä on aina jonkin verran iterointia, mutta tätä pyritään koko ajan rajoittamaan. Näin ollen toimitettu ohjelmisto ei vastaa asiakkaan todellisiin tarpeisiin (Sommerville, 1996). Jo Royce (1970) ehdotti tähän malliin muutoksia, koska hän näki, että suurissa ohjelmistokehitysprosesseissa tapahtuu väistämättä toistoja eri vaiheiden välillä.

3.2 Inkrementaalinen malli

Inkrementaaliset mallit perustuvat pienien osien kehittämiseen ja niiden todentamiseen kehitettävässä ohjelmistossa. Järjestelmää kartuttavia ohjelmistolisäyksiä tehdään hyvin nopeaan tahtiin. Lisäykset kehitetään ja testataan pienissä riippumattomissa tiimeissä, jotka yhdessä muodostavat suuren projektin. Lisäyksien elinkaaren hallintaan käytetään järjestettyjä spesifikaatioita, mikä tarkoittaa, että tuotteen toiminnallisuudet on jaettu yksittäin kehitettäviin selkeästi sisäkkäisiin alaryhmiin (Selby, Basili, & Baker, 1987).

Järjestelmän integrointi on jatkuvaa, ja uusia toiminnallisuuksia saadaan sitä mukaa, kun onnistuneita lisäyksiä saadaan liitettyä. Vaatimusten täyttyminen osoitetaan virallisesti. Tämä tarkoittaa, että toteutettuja vaatimuksia ei muuteta, mutta seuraavissa lisäyksissä toteutettaviin vaatimuksiin voidaan tehdä muutoksia (Sommerville, 1996). Portaittainen vaatimusten hionta ja muokkaus luovat ohjelmistosuunnitteluun peräkkäiset tasot (Mills, Dyer, & Linger, 1987). Tällä pyritään minimoimaan rajapinta- ja suunnitteluvirheitä sekä auttamaan kehittäjiä säilyttämään kontrollin ohjelmistokehityksessä.

Kehityksessä ei ole yksikkötestausta, ja integrointitestauksen tarkoitus on validoida järjestelmän luotettavuus mieluummin kuin etsiä virheitä (Selby ym., 1987). Ajatus on siis nopeasti kehittää laadukas ja käyttäjien toivoma tuote. Käyttökokemuksista nousevat uudet vaatimukset tehdään sitten seuraavaan ohjelmistoversioon (Linger, 1994). Raporttien mukaan tämä menetelmä ei ole tuottanut kehitettyihin tuotteisiin virheitä juuri lainkaan (Selby ym., 1987).

3.3 Evoluutiomalli

Vesiputousmallit ja inkrementaaliset mallit ovat spesifikaatioihin perustuvia ohjelmisto-kehitysmalleja. Ne ovat siinä mielessä ongelmallisia, että vaatimukset täytyy määrittää etukäteen. Tämä aiheuttaa herkästi vaatimuksien muokkaamista kehitysprosessin aikana. Evoluutiomalleissa kehitys integroi yhteen määrittelyn, suunnittelun ja toteutuksen. Järjestelmävaatimuksista muodostetaan hahmotelma, jonka pitäisi kuvata järjestelmän toiminta ohjelmiston kehittäjälle. Seuraavassa vaiheessa ohjelmisto kehitettään niin nopeasti kuin mahdollista. Viimeisessä vaiheessa järjestelmä evaluoidaan käyttäjän kanssa ja tehdään muokkauksia, kunnes järjestelmän toiminnallisuus vastaa käyttäjän tarpeisiin. Nopeasti kehitettyä ohjelmistoprotoa voidaan käyttää lopullisen järjestelmän määrittelyssä apuna tai se voidaan jatkokehittää käyttäjälle toimitettavaksi järjestelmäksi. (Sommerville, 1996)

Tässä kehitysmallissa on kuitenkin paljon ongelmia. Niitä aiheuttavat esimerkiksi pelkästään loppukäyttäjään keskittyminen, ohjelmistoon tehtävät jatkuvat muutokset sekä prosessin vaiheiden huono näkyvyys. Kun keskitytään pelkästään käyttäjän toivomiin toimintoihin, kriittisiä organisatorisia vaatimuksia, kuten yhteensopivuutta muiden järjestelmien kanssa, ei välttämättä oteta riittävästi huomioon. Lisäksi jatkuvat muutokset heikentävät ohjelmiston rakennetta, joten lopputulokseen on usein vaikeaa ja kallista tehdä muutoksia. Tämän vuoksi myös ylläpito tulee kalliiksi, ja ohjelmisto luultavasti täytyy kirjoittaa kokonaan uudelleen suhteellisen lyhyellä aikavälillä. Kun kehitysprosessissa tehdään melkein kaikki osa-alueet yhtäaikaisesti, työn etenemistä on vaikea arvioida ja hallita. Evoluutiomallia on kehitetty lisäämällä erityisesti laajoissa järjestelmäkehitysprosesseissa tarvittavat projektinhallintavaatimukset. Työn tuloksena on muodostunut evoluutiomallin puutteita korjaava spiraalimalli. (Sommerville, 1996)

3.4 Spiraalimalli

Barry Boehm (1988) esittelee artikkelissa spiraalimallin, joka on kehitetty vastaamaan edellisten kehitysmallien puutteisiin. Spiraalimallin olennainen piirre on minimoida riskiä prototyyppien toistuvalla kehittämisellä. Riskilähtöisyys on mallin suurin etu verrattuna

aikaisempiin dokumentti- ja koodilähtöisiin malleihin. Riskianalyysi tehdään jokaisessa kehityskierroksessa (B. W. Boehm, 1988). Spiraalimallin vaiheet ovat:

- tavoitteiden, vaihtoehtojen ja rajoitusten määrittäminen
- vaihtoehtojen arviointi sekä riskien tunnistaminen ja ratkaiseminen
- seuraavan tason tuotteen kehittäminen ja todentaminen
- seuraavien vaiheiden suunnittelu ja resurssien arviointi

Spiraalimallissa ohjelmisto edistyy jokaisella kierroksella. Kehitystyö aloitetaan spiraalin keskeltä edeten ulospäin. Spiraalin jokaisessa kierroksessa asiakas arvioi työn ja ehdottaa siihen tarvittavat muutokset. Kuten jo aikaisemmin todettiin, jokaisessa spiraalin kierroksessa tehdään riskianalyysi, jonka tuloksena saadaan päätös kehitystyön jatkamisesta (Ruparelia, 2010).

Mikäli riskejä havaitaan kehitykseen liittyen, seuraava askel seuraa inkrementaalisen vesiputousmallin lähestymistapaa. Mikäli riskit kohdistuvat kehitettyyn tuotteeseen, seuraavalla kierroksella käytetään evolutionääristä kehitystä. Tällä taataan, että seuraavalla neljänneksellä pystytään tekemään haluttu prototyyppi. Riskinhallintaa käytetään hallitsemaan kierrosten kustannuksia (Ruparelia, 2010). Mikäli riski arvioidaan liian suureksi, kehitysprojekti lopetetaan. Jokaisen kierroksen lopussa pidettävä katselmointi takaa, että sidosryhmät sitoutuvat seuraavassa kierroksessa tehtävään kehitysvaiheeseen.

Spiraalimalli ei määrittele kehitystyön alkua eikä loppua, joten jo Boehm kehitti täydentävän mallin Mission Opportunity Model (MOM) ottamaan ne huomioon. Spiraalimallista on lisäksi kehitetty lukuisia versioita, kuten esimerkiksi Wheel-and-spoke -malli (Ruparelia, 2010) tai Theory-W -perusteinen spiraalimalli (B. Boehm, Bose, Horowitz, & Lee, 1995). Spiraalimalli vaatii hyvin mukautuvaa johtoa sekä melko joustavaa sopimusmallia sidosryhmien välille ja kierroksista päätettäessä. Se luottaa myös vahvasti suunnittelijoiden kykyyn analysoida riskiä seuraavalle kierrokselle (Ruparelia, 2010).

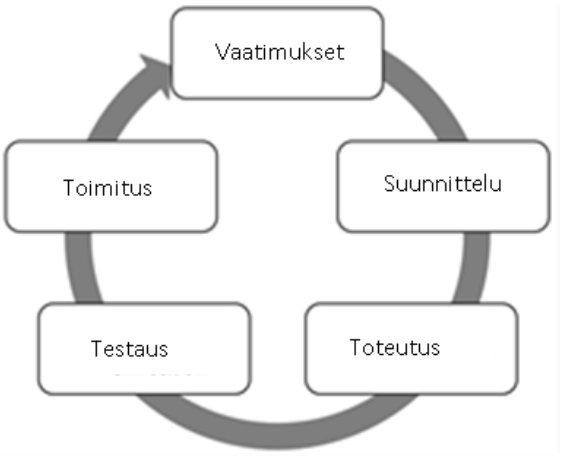
3.5 Ketterä kehitys

Iteratiivisia ja inkrementaalisia ohjelmistonkehitysmenetelmiä on kehitetty monissa tutkimusprojekteissa, mutta ne otettiin laajemmin käyttöön vasta asiantuntijoiden pitämän kokouksen jälkeen vuonna 2001. Kokouksessa 17 asiantuntijan joukko keräsi iteratiiviset ja inkrementaaliset menetelmät agile-menetelmien kokonaisuudeksi ja kirjoitti agile manifestin (Deuff & Cosquer, 2013, ss. 6–7). Manifesti koostuu neljästä ketterän kehityksen menetelmiä kuvaavasta arvosta, jotka korostavat (Agile Alliance, 2001):

- yksilöitä ja vuorovaikutusta enemmän kuin prosesseja ja työkaluja
- toimivaa ohjelmistoa enemmän kuin kattavaa dokumentointia
- yhteistyötä asiakkaan kanssa enemmän kuin sopimusneuvottelua
- muutoksiin reagoimista enemmän kuin suunnitelman seuraamista

Ketterissä kehitysmenetelmissä tiimityöskentely pyritään pitämään toimivana aktiivisella vuorovaikutuksella sekä kiinnittämällä huomiota kehitystyössä mukana oleviin henkilöihin. Tuotteen, vaikka se ei olisi valmiskaan, täytyy aina toimia. Sidosryhmien täytyy olla mukana projektissa jakamassa tietoa kehitystiimille, jolloin tuotteen odotukset ja toteutus vastaavat toisiaan. Alkuperäisen suunnitelman tulee olla mahdollisimman joustava, jotta kehityksen aikana voidaan tehdä muutoksia. Tarkoitus on vastata asiakkaiden tarpeisiin mahdollisimman täydellisesti. Menetelmät mahdollistavat tuotteen säännöllisen evaluoinnin, reaktiivisuuden vaatimusten suhteen sekä joustavuutta tuotteen muokkaamisessa. (Deuff & Cosquer, 2013, s. 6)

Kaikki ketterän kehityksen menetelmät perustuvat samaan filosofiaan ja yhteisiin periaatteisiin, mutta ne eroavat toisistaan siinä, miten ne toteuttavat näitä periaatteita ja filosofiaa. Yleisimpiä ketterän kehityksen menetelmiä ovat Scrum, XP ja Lean. Kaikille ketterien kehitysmallien menetelmille on tärkeää, että kehitys etenee toistamalla kehitysvaiheita useita kertoja. Periaatekuva ketterästä kehitysmallista on oheisessa kuviossa (Kuvio 3).



Kuvio 3. Ketterä kehitysmalli (Houston & Rosemergy, 2016, s. 43)

3.5.1 Lean-ajattelu

Lean-ajattelulla voidaan tarkoittaa joko tuotannon käytäntöjä tai tapaa ajatella. Se sai alkunsa autonvalmistaja Toyotan tuotannosta (Liker & Hoseus, 2008), kun Taiichi Ohno kehitti Kanbanin 1950-luvun taitteessa. Hänen tarkoituksenaan oli kehittää tuotannon kontrolleja prosessien välillä ja toteuttaa Just In Time (JIT) -valmistusmenetelmä autonvalmistustehtaisiin (Gross & McInnis, 2003, s. 1).

Vaikka Lean-ajattelun periaatteet on suunniteltu alun perin autonvalmistusprosessiin, niitä voidaan laajasti soveltaa muille aloille ja myös ohjelmistotuotantoon (Poppendieck & Poppendieck, 2009). Lean-ajattelun yleinen tavoite on tuottaa arvoa ja ennen kaikkea korostaa arvoa lisäävää tuotantoa. Tästä arvosta pitäisi hyötyä loppukäyttäjä, asiakas tai muu tuotteesta kiinnostunut osapuoli. Pystyäkseen tuottamaan arvoa Lean-ajattelussa tärkeimpänä periaatteena pidetään turhan karsimista (Poppendieck & Poppendieck, 2003).

Lean-ajattelussa voidaan tunnistaa seitsemän erityyppistä periaatetta, jotka Poppendieck & Poppendieck (2009) ovat muuttaneet ohjelmistokehitykseen sopiviksi. Niitä ovat turhan eliminointi, oppimisen vahvistaminen, päätöksenteko mahdollisimman myöhään, toimittaminen mahdollisimman nopeasti, työryhmän voimaannuttaminen, eheyden rakentaminen ja kokonaisuuden näkeminen (Poppendieck & Poppendieck, 2009).

Turhan eliminoinnilla pyritään vähentämään kaikkea, mikä asiakkaan näkökulmasta ei lisää arvoa tuotteeseen. Ideana on saada selville, mitä asiakas haluaa ja kehittää haluttu tuote mahdollisimman nopeasti. Ohjelmistokehitys on useista toistoista koostuva oppimisprosessi, jossa isoissa tiimeissä tehdään monimutkaisia tuotteita. *Oppimisen vahvistaminen* on paras keino parantaa ohjelmistonkehitystä. *Myöhäinen päätöksenteko* on tehokasta silloin, kun päätökseen liittyy epävarmuustekijöitä. Muuttuvissa olosuhteissa kehitys-vaihtoehtojen pitäminen avoinna on kannattavampaa kuin päätösten tekeminen aikaisessa vaiheessa epävarmoilla perusteilla. (Poppendieck & Poppendieck, 2009)

Nopealla kehityksellä saadaan asiakkaalle tuote, jota hän tarvitsee parhaillaan, eikä tarve ehdi mennä ohi. Mikäli kehitys on nopeaa, päätöksiä voidaan viivästyttää, jolloin asiakkaat voivat miettiä pidempään, mitä he todella tarvitsevat. Nopealla kehityksellä voidaan saada myös nopeaa ja luotettavaa palautetta. Ohjelmistokehityksessä toistuvista iteraatioista opitaan koko ajan. Mitä nopeampi iteraatio on, sitä enemmän ehditään oppia.

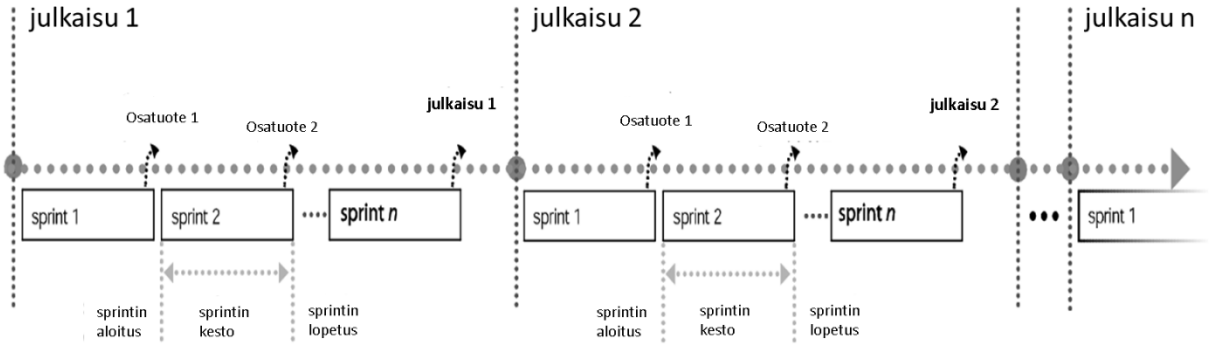
Poppendieckin & Poppendieckin (2009) mukaan varsinaisten kehitystyötä tekevien henkilöiden yhdistämä tieto johtajan ohjaamana tuo parempia päätöksiä ja prosesseja kuin yksittäinen ihminen pystyisi tekemään. *Työntekijät tulee valtuuttaa* tekemään töitään koskevia päätöksiä, koska myöhäinen päätöksenteko ja nopea kehitystyö eivät olisi mahdollisia keskusjohtoisella johtamistavalla. *Ohjelmisto on eheä*, jos sen arkkitehtuuri on koherentti ja käytettävyys hyvä. Lisäksi sen tulee sopia tarkoitukseensa. Eheys saadaan aikaan viisaalla johtamisella, asiantuntemuksella, tehokkaalla viestinnällä sekä terveellä kurinalaisuudella. *Kokonaisuuden näkemisessä* pitäisi keskittyä järjestelmäkokonaisuuden toimintaan, ei niinkään oman erityisen osa-alueen hiomiseen. Kun henkilöitä tai organisaatioita mitataan heidän erityisen osaamisensa, eikä koko toiminnan kannalta, tulokset eivät ole optimaalisia. (Poppendieck & Poppendieck, 2009)

Periaatteiden toteuttamiseksi on kehitetty 22 ajatustyökalua, joiden tarkoituksena on auttaa Lean-periaatteiden soveltamista ketterän kehityksen menetelmien mukaiseen ohjelmistokehitykseen (Poppendieck & Poppendieck, 2009).

3.5.2 Scrum

Scrum sai alkunsa Takeuchin & Nonakan (1986) julkaisemasta artikkelista, jonka mukaan japanilaiset yritykset kykenivät nopeasti tuottamaan maailmaluokan tuotteita. Menestys perustui itseohjautuviin tiimeihin ja johdon aktiiviseen osallistumiseen (Takeuchi & Nonaka, 1986). Artikkeli sisälsi kaikki peruselementit Scrumista, vaikkakin DeGrace & Stahl (1990) keksivät menetelmän nimen myöhemmin viitaten Takeuchin ja Nonakan artikkelissa mainittuun rugby-termiin.

Sutherland (2004) käytti ensimmäisen kerran Scrumia vuonna 1993 kehitysprojektissaan, ja kertoi siitä kokemuksiaan. Myöhemmin Sutherland ja Schwaber määrittelivät perusperiaatteet Scrum-menetelmälle (Schwaber & Sutherland, 2016). Scrum on projektinhallintamenetelmä, joka on suunniteltu erityisesti ohjelmistokehitykseen. Scrum perustuu iteraatioihin eli sprintteihin (engl. sprint), jotka yleensä kestävät kaksi tai neljä viikkoa. Kaikki sprintin aikana tehtävä työ organisoidaan samalla tavalla, ja sillä on sovittu kesto. Jokainen uusi sprintti alkaa heti edellisen jälkeen, ja sprintille määritettyä tavoitetta ei voi muuttaa sen aikana. Tuote kehitetään inkrementaalisesti, mutta sprintin lopussa valmistuva osatuote on aina toimiva, mahdollisesti toimitettava ja käyttökelpoinen (Deuff & Cosquer, 2013 s.12). Projektissa tuotteen toimitusversioita kehitysvaiheiden lopussa kutsutaan julkaisuiksi (engl. release). Julkaisu on useasta sprintistä koostuva lopputuote. Peräkkäiset tuotejulkaisut tapahtuvat toinen toisensa perään (Schwaber & Sutherland, 2016). Kuvio 4 esittää kehitystyön etenemistä osatuotteista julkaisuiksi asti.



Kuvio 4. Scrumin vaiheet (Deuff & Cosquer, 2013, s. 8)

Scrumissa kehitysprosessi perustuu kolmeen erityyppiseen elementtiin: osallistujien rooleihin projektissa, käytettyihin artefakteihin sekä seremonioihin. Kaikilla näillä elementeillä on oma tarkoituksensa, ja ne ovat olennaisia menetelmässä tuoden hyötyä projektiin. Menetelmässä prosessi muodostuu rooleista, jotka pitävät yhteyttä eri elementtien välillä. Scrumin kolme roolia ovat (Deuff & Cosquer, 2013):

1. Tuotteen omistaja (engl. product owner) on tuotteen tilaaja eli asiakas. Hän tekee koko tiimille jaettavan vision tuotteesta. Asiakas määrittää tuotteen sisällön, hallinnoi tuotteen ominaisuuksien priorisointia sekä huolehtii, että kehitystiimi ymmärtää nämä prioriteetit.
2. Scrummaster (engl. scrum master) ei ole projektipäällikkö, vaan hän auttaa tiimiä soveltamaan Scrumia ja adaptoimaan sen kehitystyöhön. Hänen tehtävänään on myös poistaa esteet, jotka voisivat hidastaa tiimin työskentelyä.
3. Kehitystiimi (engl. development team) on vastuussa tuotteen kehityksestä. Se organisoi työnsä, päämääränään työn tuottavuuden, joustavuuden ja luovuuden optimointi. Lisäksi tiimi kehittää taitojaan päästäkseen annettuihin tavoitteisiin.

Scrumin elementteihin kuuluvat artefaktit ovat toiminnan keskeisiä käsitteitä. Niitä ovat tuotteen kehitysjono ja sprintin tehtävälista. *Tuotteen kehitysjono* on prioriteettijärjestyksessä oleva tehtävälista, jossa tärkein tehtävä on listan ylimpänä. Listan tehtävät kuvaavat tuotteen toimintoja. Lista muuttuu ajan mukaan toiminnallisten ominaisuuksien lisäyksillä ja karsinnalla. Myös uudet prioriteetit, kuten käyttäjätarpeiden muutokset, kaupan olosuhteet ja teknologiset evoluutiot, muuttavat sitä (Schwaber & Sutherland, 2016).

Listan tärkeimmät tehtävät on esitetty huomattavasti yksityiskohtaisemmin kuin sen lopussa olevat optionaaliset tehtävät. Kun projekti etenee, seikkaperäisesti esitettyjen tehtävien määrä kasvaa, joten niin kauan kuin tuote on olemassa, tuotteen kehitysjono muuttuu ja elää. *Sprintin tehtävälista* edustaa kehitysjonon osaa, joka on kehitysprosessissa työn alla parhaillaan olevassa iteraatiossa. Tämä tehtävälista on päätetty sprintin alussa, ja tavallisesti sitä ei voi muuttaa sprintin aikana (Schwaber & Sutherland, 2016).

Scrum-prosessi muodostuu viidestä seremoniasta (engl. ceremonies), joita ovat julkaisun ja sprintin suunnittelupalaverit, päiväpalaveri sekä sprintin katselmointi ja retrospektiivi. *Julkaisun suunnittelupalaverissa* tuotteen kehitysjonossa olevat tehtävät priorisoidaan ja jaetaan saman kestoisiin iteraatioihin. Tämä seremonia on ennen kehitystyön aloitusta, ja siksi se pidetään vain kerran julkaisun aikana. (Deuff & Cosquer, 2013)

Sprintin suunnittelupalaverin tarkoitus on jakaa iteraatioon kuuluvat tehtävät osiin lyhyiksi kehitystehtäviksi. Palaveri pidetään sprintin alussa. Tehtävät jaetaan kolmeen eri kategoriaan: kehitystä vaativat tehtävät, kehitysprosessissa jo mukana olevat tehtävät ja valmiit tehtävät. Tehtävien perusteellinen käsittely palaverissa toimii alkusysäyksenä sprintin aloitukselle. (Deuff & Cosquer, 2013)

Kestoltaan 15 minuuttia olevassa *päiväpalaverissa* jokainen tiimin jäsen kertoo lyhyesti, mitä hän teki edellisenä päivänä ja mitä hän aikoo tehdä kyseisenä päivänä. Lisäksi hän kertoo tekemistä hidastavista mahdollisista esteistä (Schwaber & Sutherland, 2012). Nämä lyhyet tapaamiset ovat Scrumissa keskeisiä, koska ne antavat mahdollisuuden arvioida projektin kehitystä ja parantavat yhteydenpitoa tiimin jäsenten välillä. Näin ne myös parantavat ymmärrystä sekä vähentävät palavereiden määrää, kun riskit pyritään tunnistamaan mahdollisimman aikaisessa vaiheessa (Deuff & Cosquer, 2013).

Sprintin katselmointi pidetään sprintin lopussa. Sen tarkoituksena on tarkistaa toiminnallinen osatuote, esitellä projektin tilanne projektin jatkosta päättämisen helpottamiseksi sekä tunnistaa mahdollisia tehtäviä seuraavaan iteraatioon. Tämän palaverin aikana tuotteen omistaja vertaa tuotetta hänen alkuperäisiin tarpeisiinsa ja ehdottaa tarvittaessa muutoksia. Kehitysjonoa korjataan näiden muutostarpeiden mukaan. Sprinttien sisältöihin voi siis tulla muutoksia muutosvaatimuksiin perustuen. *Sprintin retrospektiivi* on sprintin lopussa pidettävä palaveri. Se pidetään sprintin katselmoinnin jälkeen, ja siihen osallistuu vain kehittäjätiimi. Palaverin tarkoitus on tunnistaa hyvin toimineet ja muutoksia vaativat asiat, jotta yhdessä löydettäisiin ratkaisuja työnteon kehittämiseksi. (Schwaber & Sutherland, 2016)

Scrumin kuvauksessa on alkuperin määritetty, että se alkaa julkaisun suunnittelupalaverista. Tämä tarkoittaa, että tuotteen kehitysjonon tehtävät täytyy tunnistaa

ennen kuin Scrum todellisuudessa alkaa. Monissa yhteyksissä tästä valmisteluvaiheesta on käytetty nimitystä sprintti 0, vaikka sillä ei ole mitään tekemistä sprintin kanssa. Toisissa yhteyksissä siitä käytetään nimitystä esi-sprintti (engl. pre-sprint) (Deuff & Cosquer, 2013, s. 11).

Esi-sprintti on erityinen vaihe, koska sen aikana perustetaan tiimi, määritetään visio ja tuotetaan ensimmäinen kehitysjono (Aubry, 2010). Sen lisäksi esi-sprintin aikana täytyy hankkia uutta tietämystä kehitystyön aloittamista varten (Keith, 2010). Tämän sprintin pituutta ei ole määritetty, vaan siihen vaikuttaa aiotun kehitettävän tuotteen kypsyys. Aikaa tietenkin kuluu enemmän uuteen teknologiaan perustuvaan ensimmäiseen julkaisuun kuin olemassa olevan tuotteen uuteen julkaisuun (Aubry, 2010).

Esivaihe voi sisältää myös tuotteen omistajan ehdottamien tehtävien tarkempaa määrittelyä. Vaatimusten tarkkuudesta riippuu, kuinka paljon tehtäviä joudutaan tarkentamaan ja määrittämään kehitysjonoon teknisiä ratkaisuja varten (Cohn, 2004).

Toinen huomioonotettava asia ennen julkaisun aloituspalaveria on, että tuotteen omistaja ei ole yksin, vaan hänen ympärillään voi olla järjestäytynyt tiimi. Tiimissä voi olla käyttäjien tai heidän edustajien lisäksi lukuisa määrä ammattilaisia, kuten markkinoinnin ja liiketoiminnan asiantuntijoita, käytettävyyssuunnittelija ja järjestelmäarkkitehti (Keith, 2010; Rubin, 2012). Schwaber & Sutherland (2016) kuitenkin määrittävät, että tuotteen omistaja ei ole ryhmä vaan henkilö. Vaikka tuotteen omistajalla on ympärillä ihmisiä, joilta hän saa tärkeää tietoa tuotteen määrittämiseen, hän kuitenkin yksittäisenä henkilönä tekee lopulliset päätökset ja kantaa vastuun niistä.

Esi-sprintti on usein kuvattu enemmän tai vähemmän yksityiskohtaisesti vaiheena, jossa pidetään yksi tai useampia kollaboratiivisia työpajoja. Tarkoituksena on kuitenkin tuoda visio etualalle ja tähän perustuen määrittää käyttäjäkertomuksista tehtävälista kehitysjonoon (Deuff & Cosquer, 2013, s. 12).

4 Vaatimusmäärittely

Hyvän ohjelmistokehityksen perustana on huolellinen vaatimusmäärittelytyö. Vaatimusmäärittely lähtee liikkeelle toimeksiannosta, ja siinä on mukana monia sidosryhmiä. Jotta vaatimusmäärittely ja koko ohjelmiston kehitys onnistuisi hyvin, toimijoiden välillä tarvitaan huolellista kommunikaatiota. Kommunikaation helpottamiseksi on kehitetty useita eri työkaluja.

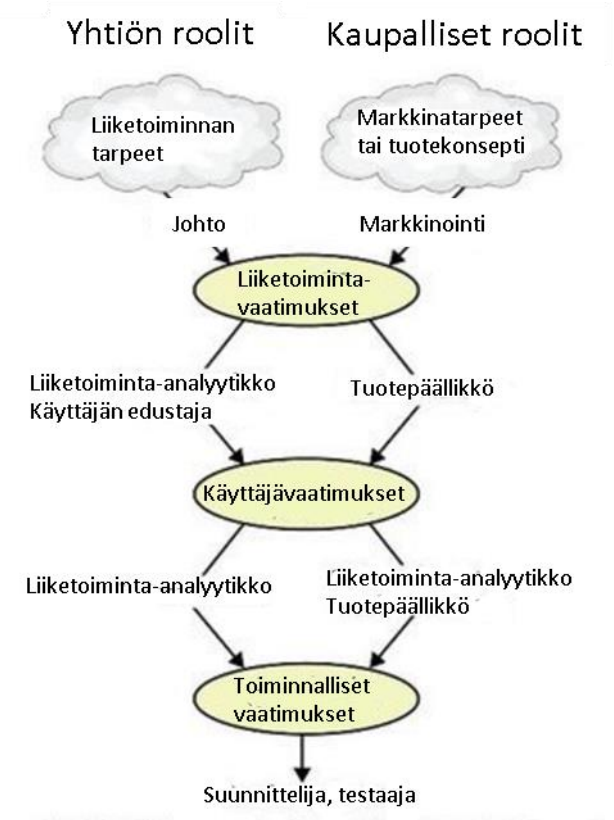
4.1 Vaatimukset yleisesti

Vaatimus -termiä on määritetty monella tapaa. Yhden määritelmän mukaan se on ominaisuus, joka tuotteella täytyy olla antaakseen arvoa sidosryhmille. Tämä määritelmä perustuu ajatukseen, että vaatimukset tulevat sidosryhmiltä ja vaatimuksilla täytyy olla joku merkitys niille. Jos ajatellaan vaatimusten koko kirjoa, kaikkien vaatimusten peilaaminen tätä ajatusta vasten voi olla vaikeaa ja aikaa vievää (Wiegers & Beatty, 2013).

Yksi vaihtoehto määritykselle on, että vaatimus on spesifikaatio siitä, mitä pitäisi toteuttaa (Sommerville & Sawyer, 1997). Spesifikaatio kuvaa, kuinka järjestelmän pitäisi toimia tai se kuvaa järjestelmän ominaisuudet. Vaatimukset voivat olla myös rajoite järjestelmän kehitysprosessissa. Määrityksessä otetaan huomioon, että vaatimukset voivat sisältää erityylisiä tietoja, jotka yhteisesti nimitetään vaatimuksiksi. Tämä määritys kattaa järjestelmän toiminnan käyttäjien näkökulmasta sekä myös sen sisäisten toimintojen määrittämisen kehittäjien näkökulmasta. Vaatimukset siis sisältävät järjestelmän toiminnan tietyissä olosuhteissa tai tilanteissa ja ominaisuudet, jotka tekevät järjestelmän käytön mahdolliseksi sen aiotuille käyttäjille. (Wiegers & Beatty, 2013)

Järjestelmien vaatimusmäärittelyyn kiinnitetään monesta syystä paljon huomiota. Vaatimusmäärittelytyö on monivaiheinen prosessi, ja siihen osallistuvat monet järjestelmän kehitystyössä mukana olevat tahot. Kuviossa 5 vaatimusmäärittely on jaettu karkeasti kolmelle tasolle. Ylimpänä on yrityksen tarpeeseen kehitetyt liiketoiminnan vaatimukset, joiden jälkeen tulee käyttäjävaatimukset. Alimmalla tasolla ovat järjestelmän toiminnalliset vaatimukset. Eri tasoilla tuotetut vaatimukset ovat erilaisia, mutta niillä on

yksi yhteinen tekijä: kehitettävän järjestelmän määrittäminen. Siten ne eivät saisi olla ristiriitaisia keskenään. Kuvassa pyritään myös esittämään eri sidosryhmien osallistuminen eritasoisten vaatimusten määrittämiseen liiketoiminnan ja markkinoinnin tarpeista lähtien. Vaatimusmäärittelyn eri vaiheille voi olla erilaisia nimityksiä eri organisaatioissa. Eroja tulee ennen kaikkea siinä, onko kyseessä organisaation sisäinen kehitystyö, vai valmistaako yritys ohjelmistoa myyntiin. (Wiegers & Beatty, 2013)



Kuvio 5. Sidosryhmien osallistuminen vaatimusmäärittelyyn (Wiegers & Beatty, 2013, s. 12)

Jotta yritysorganisaatio pystyy toimimaan tehokkaasti ja kilpailukykyisesti, yrityksen johdon tai markkinoinnin täytyy pystyä määrittämään liiketoiminnan tarve, markkinatarve tai liiketoimintavaatimukset uudelle tuotekonseptille. Kun tavoite on määritetty, liiketoiminta-analyytikko määrittelee käyttäjien edustajien kanssa käyttäjävaatimukset. Mikäli vaatimusmäärittely kohdistuu myytävään tuotteeseen, tuotteelle nimetään tuotepäällikkö määrittämään tarkemmat vaatimukset ja ominaisuudet. Käyttäjävaatimusten

tai tuotteen ominaisuuksien täytyy vastata liiketoiminnan asettamiin vaatimuksiin. Liiketoiminta-analyytikko tai tuotepäällikkö johtaa käyttäjävaatimuksista järjestelmän toiminnallisuudet, joista määritetään toiminnalliset vaatimukset. Järjestelmän suunnittelija miettii tekniset ratkaisut toiminnallisten ja ei-toiminnallisten vaatimusten mukaisesti ottaen huomioon erilaisten rajoitusten asettamat reunaehdot. Testaajat suunnittelevat vaatimusten verifioinnin. (Wiegers & Beatty, 2013, s. 13)

Vaikka kuviossa 5 tieto valuu vaihe vaiheelta ylhäältä alaspäin, vaatimukset kehittyvät vaiheita useita kertoja toistamalla (Wiegers & Beatty, 2013, s. 13). Vaatimukset jaetaan eri tyyppeihin niiden sisältämän informaation mukaan. Vaikka vaatimukset jaetaan karkeasti kolmeen pääryhmään, niitä täydentäviä vaatimuksia on paljon ja niitä tulee monen eri sidosryhmän tuottamana.

Liiketoimintavaatimuksissa keskeisintä on yrityksen liitetoimintatavoitteiden tai tuotetta tilaavan asiakkaan asettamat vaatimukset. Tarkoitus on kuvata liiketoimintaetu, jonka organisaatio pyrkii saamaan toteuttamalla aiotun tuotteen. *Käyttäjävaatimusten* tarkoituksena on kuvata käyttäjien haluamat toiminnot. Käyttäjien tulisi pystyä esittämään, mitä tavoitteita ja tehtäviä tuotteella halutaan toteuttaa. Käyttäjävaatimuksissa tulisi olla mukana myös käyttötyytyväisyyteen vaikuttavia ominaisuuksia ja piirteitä. Käyttötapauksia, tapahtumavastetaulukoita tai käyttäjien kirjoittamia kertomuksia voi käyttää esimerkkeinä käyttäjävaatimuksissa. (Wiegers & Beatty, 2013, s. 8)

Ideaalitapauksessa käyttäjien todelliset edustajat kirjoittaisivat käyttäjävaatimukset, jolloin käyttäjien toiveet saataisiin ilman välikäsiä. Useimmissa projekteissa on useita käyttäjäluokkia ja siten myös sidosryhmiä, joiden tarpeet tulisi kerätä yhteen. Monissa yhteyksissä onkin tarpeellista laajentaa käyttäjävaatimukset sidosryhmien vaatimuksiksi, jolloin käyttäjävaatimuksissa kuvataan paljon muutakin kuin järjestelmän käyttöä koskevia vaatimuksia. (Wiegers & Beatty, 2013, s. 9)

Jokainen sidosryhmä tuo haluttuun ohjelmistotuotteeseen liittyvät vaatimukset esille omasta perspektiivistään. Vaatimusten tulisi keskittyä sidosryhmäalueisiin, ei järjestelmäratkaisuihin. Sidosryhmien vaatimukset tulisi pitää mahdollisimman pieninä ja helppoina ymmärtää, sekä niiden tulisi olla ei-teknisiä ja realistisia (Hull ym., 2011, s.

114). Kerätyt vaatimukset täytyy karsia, jotta tasapaino tarpeita vastaavan välttämättömän työn ja kehitysprojektiin käytettävissä olevien resurssien välillä säilyisi. Tarpeiden ja odotusten keräyksen, ristiriitojen poistamisen, harmonisoinnin sekä priorisoinnin jälkeen hyväksytyt vaatimukset antavat perustan teknisen työsuunnitelman yksityiskohtaisemmalle tekemiselle (Schmidt, 2013, ss. 124–125). Ohjelmistolla on kuitenkin tarkoitus vastata asiakkaan ja sidosryhmien tarpeisiin, joten on tärkeää, että kaikki osalliset hyväksyvät vaatimukset ennen suunnittelutyön aloitusta (Schmidt, 2013, s. 43).

Toiminnallisten vaatimusten tarkoitus on määrittää järjestelmän toiminta halutuissa olosuhteissa ja tiloissa. Toiminnalliset vaatimukset kuvaavat tehtäviä, joita käyttäjän tulisi pystyä toteuttamaan kehitettävällä järjestelmällä. Kun mietitään projektin menestystä, on olennaisen tärkeää, että toiminnalliset vaatimukset ovat linjassa liiketoiminta- ja käyttäjävaatimusten kanssa. Vasta toiminnalliset vaatimukset kirjoitetaan muotoon ”pitää” (Wiegers & Beatty, 2013).

Järjestelmävaatimukset on yhdistetty monista järjestelmäkomponenteista ja alijärjestelmistä (ISO/IEC/IEEE, 2011). Tässä mielessä järjestelmä-termi voi sisältää ohjelmiston lisäksi myös laitteita. Myös prosessit ja ihmiset kuuluvat järjestelmään, joten määrityksen mukaan osa toiminnoista voi olla myös ihmisen toimintaa (Wiegers & Beatty, 2013). Järjestelmävaatimus -termiä käytetään joissain yhteyksissä myös siinä mielessä, että se määrittää ohjelmistojärjestelmän yksityiskohtaiset vaatimukset. Tässä työssä termiä käytetään määrittämään järjestelmää ja sen alijärjestelmien kokonaisuutta.

Liiketoimintasäännöt sisältävät yritysten politiikat, kansalliset säännöt, teollisuusstandardit ja laskennalliset algoritmit. Säännöt eivät itsessään ole ohjelmistovaatimuksia, vaan ne antavat rajoituksia ohjelmistosovellukselle. Kuitenkin säännön noudattaminen tuotteessa voi usein johtaa toiminnallisen vaatimuksen määrittämisen tuotteeseen. Esimerkiksi yrityksen liiketoimintasääntöihin kuuluva tietoturvapolitiikka voi aiheuttaa tiettyjä laatuattribuutteja, jotka sitten toteutetaan tuotteen toiminnallisuuksiin. Tietyn toiminnallisen vaatimuksen synty voidaan siten jäljittää takaisin tiettyyn liiketoimintasääntöön. (Wiegers & Beatty, 2013, s. 10)

Vaatimukset yleisesti jaetaan toiminnallisiin ja *ei-toiminnallisiin* vaatimuksiin. Toiminnalliset vaatimukset kuvaavat järjestelmän näkyvää toimintaa eri olosuhteissa. Ei-toiminnalliset vaatimukset kuvaavat järjestelmän tärkeitä ominaisuuksia, kuten saatavuus, käytettävyys, turvallisuus, suorituskyky ja useat muut piirteet. Ei-toiminnallisia vaatimuksia pidetään synonyymina laatuattribuutille, mutta joissain tapauksissa se on liian rajoittava ilmaus. Esimerkiksi suunnittelu- ja toteutusrajoitukset ovat myös ei-toiminnallisia vaatimuksia, samoin ulkoisten rajapintojen vaatimukset. (Wiegers & Beatty, 2013, s. 10)

Toisaalta laatuattribuutti on järjestelmän tai prosessin ominaisuus, joka ilmaisee laatua millä tahansa laatuun liittyvällä alueella. ISO 9126-1-standardin mukaan järjestelmän laatu on yleisesti sen sopivuutta käyttötarkoitukseen. Standardi määrittää laatumallin neljällä toisiinsa linkittyvällä alueella. Niitä ovat prosessin laatu, sisäinen laatu, ulkoinen laatu ja laatu käytössä (Berenbach ym., 2009, ss. 132–133). Käyttöä koskeva laatu on jaettu kuuteen kategoriaan: toiminnallisuus, käytettävyys, luotettavuus, tehokkuus, ylläpidettävyys ja siirrettävyys. Standardin tarkoitus on antaa kehykset ohjelmiston laadun evaluoinnille. Sen kuvaamaa mallia voidaan käyttää mihin tahansa ohjelmistoon (Abran, Khelifi, Suryn, & Seffah, 2003). Tietoturva on ISO 9126-1-standardissa toiminnallisuuskategorian yksi aliominaisuus.

ISO/IEC 25010:2011-standardissa, joka on paranneltu versio ISO 9126-standardista, tietoturva on nostettu yhdeksi laatua kuvaavaksi ominaisuudeksi. Tietoturvalle on määritetty aliominaisuudet: luottamuksellisuus, eheys, kiistämättömyys, vastuullisuus ja autenttisuus (ISO/IEC 25010: 2011). Tämä määritelmä eroaa hieman laajasti hyväksytystä Baskervillen (1988) määrittämästä kolmesta tietoturvaominaisuudesta, joita ovat eheys, saatavuus ja luottamuksellisuus. Nämä ominaisuudet ovat laajasti käytössä tietoturvavaatimusten määrittelyssä.

Toiminnalliset ja laatuattribuuttivaatimukset tukevat toinen toistaan, eivätkä ne ole riittäviä yksinään. Kun toiminnallisia vaatimuksia eli käyttötapauksia määritetään, niissä tulisi määrittää myös, kuinka niiden tulisi toteutua. Toiminnalliset vaatimukset ovat perusainesta, kuten substantiiveja ja verbejä, kun taas laatuattribuuttivaatimukset ovat

tyypillisesti toiminnallisten vaatimusten määritteitä, kuten adjektiiveja ja adverbeja. Yleisesti laatuattribuutit voivat mennä päällekkäin toisten laatualueiden kanssa ja laatuattribuutti yhdellä alueella voi olla monen laatuattribuutin indikaattori muilla alueilla. (Berenbach ym., 2009, s. 135)

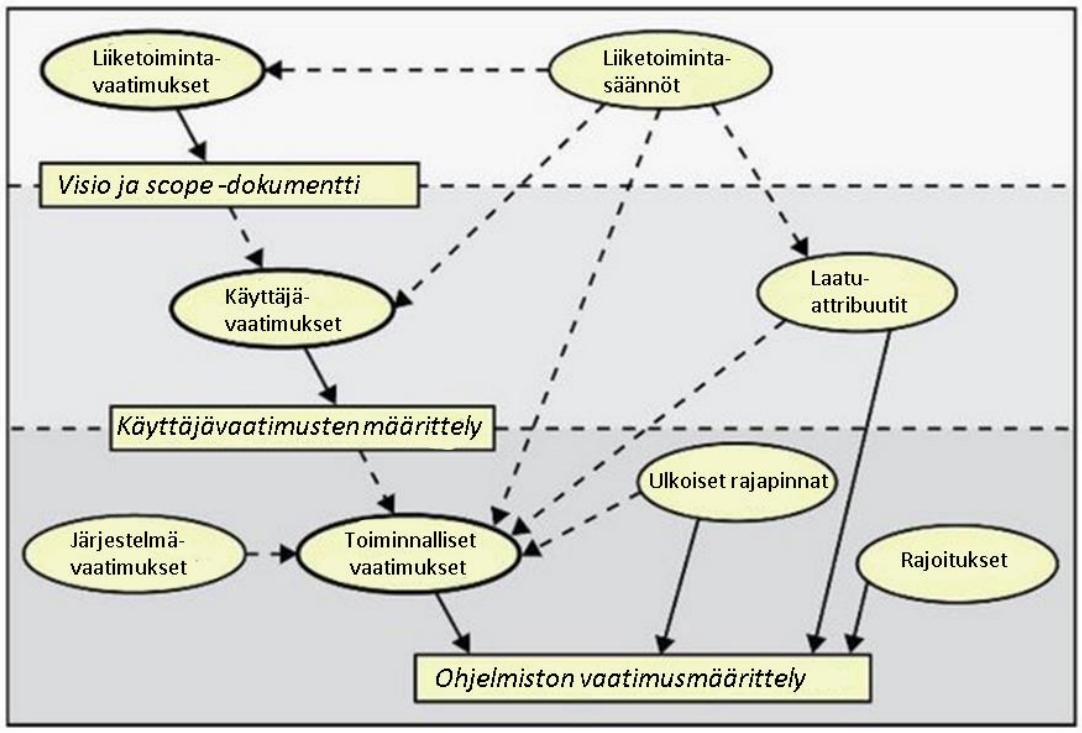
Schmidt (2013) kritisoi kirjassaan ei-toiminnallisten vaatimusten käsitettä. Siinä korostetaan, että ohjelmistotuotteen toimintojen tulee sisältää näkökulmat, kuten informaation tietoturva, virheen tai puutteiden korjaukset ja turvallisuuteen liittyvät toiminnot. Nämä toiminnot on usein virheellisesti liitetty ei-toiminnallisiin vaatimuksiin, vaikka ne pitäisi mieltää erityisteknisiksi valtuuksiksi, jotka tulisi pitää toiminnallisen arkkitehtuurin mukana. Esimerkiksi vian havaitseminen ja vaste- tai korjaustoiminnot vaikuttavat suoraan tuotteen luotettavuuteen. (Schmidt, 2013, s. 43)

Artikkelissaan Cysneiros, do Prado Leite & de Melo Sabat Neto (2001) tuovat esille, että ei-toiminnalliset vaatimukset ovat aina sidoksissa toiminnallisten vaatimusten kanssa. Näin ei-toiminnalliset ominaisuudet voidaan nähdä vaatimuksina, jotka rajoittavat tai asettavat laatuattribuutteja toiminnallisiin vaatimuksiin. Artikkelissa ei-toiminnallisia vaatimuksia pidetään kuitenkin vastakohtana toiminnallisille vaatimuksille, koska ne eivät ilmaise toteutettavaa toiminnallisuutta kehitettävänä olevaan järjestelmään. Sitä vastoin ne ilmaisevat huomioon otettavia toiminnan olosuhteita ja rajoituksia. (Cysneiros ym., 2001)

Vaikka puhutaankin ei-toiminnallisista näkökulmista, ei ole riittävää käsitellä niitä vasta järjestelmän suunnitteluvaiheessa. Ne pitäisi käsitellä vaatimuksina ja ottaa huomioon hyvin aikaisessa vaiheessa järjestelmän kehitysprosessissa. Laatuattribuuttivaatimusten tehokas käsittely on kuitenkin osittain ajoituskysymys. Monet suunnittelutiimin jäsenet eivät ole valmiita keskustelemaan paljonkaan laatuattribuuteista, ennen kuin laaja toiminnallisten vaatimusten joukko on määritetty. (Berenbach ym., 2009)

Laatuattribuuttien suhteellinen tärkeys voi olla erilainen eri sidosryhmillä. Ulkoisilla sidosryhmillä kompromissin tekeminen perustuu ulkoisiin tapahtumiin, joista suunnittelija ei välttämättä ole tietoinen. Laatuattribuuttivaatimukset vaativat järkevän sekoituksen sidosryhmäpäämääriä ja asiantuntijoiden tietämystä, joten niiden määrittäminen vaatii pohdintaa. (Berenbach ym., 2009)

Kuvio 6 osoittaa, kuinka erityyppinen vaatimusinformaatio on suhteessa toisiinsa. Katkoviivanuolet osoittavat, mistä informaatiosta vaatimukset ovat lähtöisin tai ainakin saaneet vaikutuksia. Yhtenäiset nuolet osoittavat, missä eri lähteistä tuleva informaatio säilytetään. Vaikka kuvassa käytetään nimitystä dokumentti, se ei välttämättä tarkoita perinteistä paperista tai sähköistä dokumenttia, vaan paikkaa, johon vaatimusinformaatio on tallennettu. (Wiegers & Beatty, 2013, s. 8)



Kuvio 6. Ei-toiminnallisten vaatimusten vaikutukset vaatimusmäärittelyyn (Wiegers & Beatty, 2013, s. 8)

Kuten jo aikaisemmin mainittiin, ohjelmistovaatimukset voidaan jakaa liiketoiminta-, käyttäjä- ja toiminnallisiin vaatimuksiin. Sen lisäksi kaikissa järjestelmissä on ei-toiminnallisia vaatimuksia, joita ei ole tarkasti eritelty kuvassa. Kuvan tarkoitus on havainnollistaa, miten erityyppisiä vaatimuksia tulisi ajatella. Kuva antaa myös kaavion, jonka avulla pystyy organisoimaan vaatimustietoa, jota tulee esille vaatimusmäärittelytyötä tehtäessä. Tietoa koskevia vaatimuksia ei ole selkeästi esitetty kuvassa. Toiminnalliset vaatimukset muokkaavat tietoa, ja tietoa koskevat vaatimukset voivat olla mukana kaikilla tasoilla vaatimusmäärittelyssä. (Wiegers & Beatty, 2013, s. 7)

4.2 Tietoturvavaatimukset

Useat vaatimukset tulevat järjestelmän käyttäjiltä tai yhteisen pohdinnan tuloksena. Yleinen lähtökohta on käyttää skenaarioita, tarinoita tai käyttötapauksia. Tämä ei oikein toimi tietoturvavaatimuksissa, koska tietoturva on todella harvoin ominaisuuden tavoite. Jos esimerkiksi kuvataan käyttötapausta, siinä harvoin kuvataan tietoturvamielessä käyttöä, kuten miten kirjautuminen tehdään ja mikä pitäisi olla sallittua tai estettyä. Vaikka olisi kyse tietoturvaominaisuudesta, tuotepäällikkö harvemmin määrittelee niistä skenaarioita. (Shostack, 2014, s. 221)

Tietoturvavaatimukset mielletään helposti tietoturva-asiantuntijoiden tekemiksi. Ne tulisi siten toteuttaa järjestelmään siinä muodossa kuin ne asiantuntija sanoo tai ne on standardiin kirjoitettu. Yleisesti ei oleteta, että asiakas olisi mukana määrittämässä tietoturvavaatimuksia tai antaisi hankintasopimuksessa vaatimuksia tietoturvasta. Nykyinen trendi on, että järjestelmän tietoturvaa pohditaan vaatimusten määrittelyvaiheessa. Aikaisemmin ne ajateltiin teknisiksi valinnoiksi, joita tehtiin implementoinnin aikana. Tällöin tietoturvaa yritettiin kehittää liian myöhään, mikä taas lisäsi kustannuksia järjestelmään. (Souag ym., 2016).

Tutkijoiden ja ammatinharjoittajien kasvava kiinnostus on tuottaa tietoturvavaatimusten suunnitteluprosesseja erilaisilla kehyksillä ja menetelmillä. Osa niistä on tavoite-orientoituneita menetelmiä tai niiden laajennuksia kuten Secure tropos (Mouratidis, 2006) ja Tropos/i* -metodologiat (Giorgini, Massacci, & Mylopoulos, 2003) sekä KAOS ja anti-mallit (van Lamsweerde, 2004).

Muita lähestymistapoja ovat objektimallin avulla tehdyt, enimmäkseen UML-pohjaiset mallit, kuten väärinkäyttötapaukset (Matulevicius, Mayer, & Heymans, 2008; McDermott & Fox, 1999; Sindre & Opdahl, 2005), tietoturvakäyttötapaukset (Firesmith, 2003), secureUML (Lodderstedt, Basin, & Doser, 2002) ja UMLsec (Jürjens, 2002). On myös esimerkkejä malleista, joissa otetaan huomioon lakisääteisten sääntöjen noudattaminen esimerkiksi pankkijärjestelmissä (Islam, Mouratidis, & Wagner, 2010; Tran, Holmes, Zdun, & Dustdar, 2011).

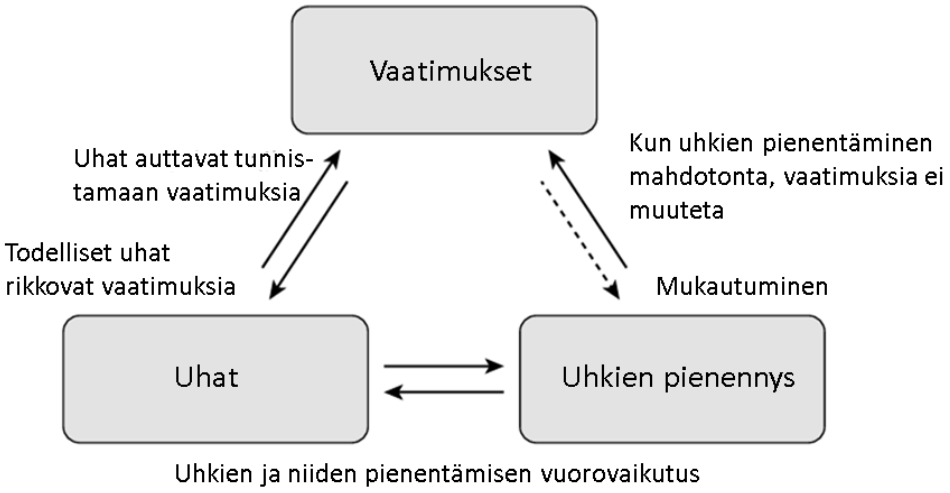
Korkealla tasolla jokaisesta mallista löytyy yhteisiä piirteitä. On samanlaisia perustyyppisiä haavoittuvia omaisuuksia, kuten data, kommunikaatio, palvelut, laitteisto, komponentit ja henkilöt. Haavoittuva ominaisuus aiheuttaa perusteiltaan samantyylisiä tietoturvauhkia, joihin voidaan hyökätä. Sitten ovat hyökkääjät, jotka profiloidaan motivaation, heille tyypillisen asiantuntemuksen ja työkalujen mukaan. (Souag ym., 2016)

4.3 Uhka-arviot

Useissa artikkeleissa tietoturvavaatimuksia lähdetään määrittämään uhkien ja sitä kautta riskien hallinnan näkökulmasta (Yu, Franqueira, Than Tun, Wieringa, & Nuseibeh, 2015). Uhkien mallintaminen voi olla myös osana tietoturvavaatimuksia (Myagmar, Lee, & Yurcik, 2005).

Tietoturvaan uhkamallinnus tuo paljon hyötyä. Se auttaa löytämään tietoturvavirheet aikaisessa vaiheessa, ymmärtämään tietoturvavaatimukset sekä suunnittelemaan ja kehittämään parempia tuotteita. Mikäli uhkamallinnus tehdään ennen ohjelmiston suunnittelun aloitusta, on helpompi löytää suunnittelu- ja tietoturvaongelmat jopa ennen kuin on tehty mitään. Lisäksi voidaan ennakoida uhkia, jotka voivat vaikuttaa myöhemmin ohjelmistoon. (Shostack, 2014)

Mikäli ohjelmistoon kohdistuvia uhkia arvioidaan, niiden vaikutuksilla on yhteyksiä vaatimusmäärittelyyn. Kuviossa 7 on esitetty, miten uhkien tunnistaminen auttaa parantamaan vaatimuksia. Kun uhkat ovat tiedossa, osa niistä voi rikkoa ohjelmistovaatimuksia, mutta samalla ne mahdollistavat tietoturvavaatimusten paremman määrittelyn. Kaikki uhkat eivät välttämättä vaikuta suoraan vaatimuksiin, mutta niiden löytäminen voi auttaa tunnistamaan uusia vaatimuksia. Kun uhkia ja niiden vaikutuksia pohditaan, vaatimuksiin palataan, jotta uhkia saataisiin pienennettyä. Pienentäminen äärimmäisyyksiin vietynä on kuitenkin lähes mahdoton työ. Esimerkiksi jos halutaan väittää, että verkko on täysin luotettava sekä siihen liittyviä uhkia pidetään epäolennaisina, väitteen toteuttaminen on lähes mahdotonta ja vaikeaa osoittaa asiakkaalle. (Shostack, 2014, s. 219)



Kuvio 7. Vaatimusten ja uhkien vuorovaikutus (Shostack, 2014, s. 219)

Tietoturvan parantamiseen tähtäävää uhkien tunnistamista on tehty jo pitkään. Uhkien tunnistaminen lähtee ennen kaikkea tietojärjestelmään kohdistuvien uhkien tunnistamisesta. Vaikka tietojärjestelmä on yleensä laajempi kokonaisuus kuin yksittäinen ohjelmisto tai järjestelmä, uhkamallinuksen kannalta niillä on hyvin samanlaiset ominaisuudet. Näin ollen tässä tutkimuksessa uhkien mallinnusta tarkastellaan tietojärjestelmiin kehitettyjen mallien avulla.

Uhkia voi pyrkiä tunnistamaan käyttämällä historiatietoa järjestelmiin murtautumisista, tietoturvaloukkausraporteista, järjestelmien pääkäyttäjähaastatteluista, palveluneuvojilta ja käyttäjiltä. Tämä voi auttaa tunnistamaan ihmisen aiheuttamat uhkat, joilla on potentiaalia aiheuttaa harmia tietojärjestelmälle ja sen informaatiolle. Sitä kautta voidaan pystyä löytämään järjestelmän haavoittuvuudet. (Stoneburner, Goguen, & Feringa, 2002)

Mahmood, Siponen, Straub, Rao ja Raghu (2010) tuovat artikkelissa esille, että tutkimalla dataa esimerkiksi todellisuudessa tapahtuneista tietoturvauhkista, ymmärretään paremmin tietoturvarikollisten toimintaa. Sisäisiä uhkia arvioitaessa on käytetty lokeja kirjautumisyrityksistä, jolloin on saatu tietoa tietoturvarikoksista (J. Wang, Gupta, & Rao, 2015).

Jo useita vuosia on käytetty listoja yleisten uhkien kokoamiseen (Baskerville, 1993). Uhkaselvitys tai listaus mahdollisista uhkalähteistä täytyy tehdä yksittäiselle

organisaatiolle ja sen toimintaympäristölle erikseen, koska eri ympäristöihin vaikuttavat erilaiset uhkat. Moni valtio tai yksityisen sektorin toimija on tehnyt tiedossa olevien uhkien tunnistamista. Helpotusta työhön tuovat hyökkäyksen havaitsemistyökalut, joten valtiot ja teollisuusorganisaatiot keräävät jatkuvasti tietoa tietoturvatapahtumista parantaen näin kykyä realistisesti arvioida uhkia. Tunnettuja informaatiolähteitä uhkien tunnistamisessa ovat muun muassa (Stoneburner ym., 2002) :

- Tiedustelupalvelut (esimerkiksi the Federal Bureau of Investigation's National Infrastructure Protection Centre)
- Federal Computer Incident Response Centre (FedCIRC)
- Joukkotiedotusvälineet, erityisesti Internet-pohjaiset lähteet, kuten SecurityFocus.com, SecurityWatch.com, SecurityPortal.com, and SANS.org.

Uhka-arvioita ja uhkamalleja voidaan tehdä tietojärjestelmiin hyvin monella eri tavalla. Yksi aikaisemmista uhkamalleista on uhkien jaottelu neliulotteisen uhkamallin mukaan. Mallissa kaikki uhkat on luokiteltu sisäisiin ja ulkoisiin uhkiin, jonka jälkeen ne on vielä luokiteltu kolmen eri jaottelun mukaan päätyen tiettyyn uhkatyyppiin (Loch, Carr, & Warkentin, 1992). Tällä mallinnuksella pyritään parantamaan ymmärrystä tietoturvauhkaluokista yksittäisten tietoturvauhkien sijaan. Tällöin voidaan arvioida uhkaluokan vaikutuksia, eikä pelkästään uhkien vaikutuksia, koska uhkat muuttuvat huomattavasti nopeammin kuin uhkaluokat.

Luokitukseen perustuvalla uhkamallilla ei esitellä pelkästään uhkatekniikoita ja niiden mahdollisia vaikutuksia, vaan siinä yhdistetään kaikki olemassa olevat uhkien tunnuspiirteet (Jouini, Rabai, & Aissa, 2014). Mallin ongelma on siinä, että se ei auta uhkien tunnistamisessa, koska uhka täytyy ensin keksiä, jonka jälkeen sen voi ominaisuuksien mukaan luokitella mallin mukaan.

Jos uhkien mallintaminen pyritään tekemään hyökkääjän näkökulmasta, mallinnus voi johtaa kohdistumaan vääriin uhkiin. Hyökkääjällä on omat motivaationsa, taitonsa, taustansa, näkökulmansa ja ehkäpä organisatoriset tärkeysjärjestyksensä, joten hyökkääjän rooliin asettuminen voi olla vaikeaa (Shostack, 2014, s. 41). Samoin kuin hyökkääjän roolissa, ohjelmistokehittäjillä voi olla vaikea ymmärtää liiketoimintaan ja yrityksen

omaisuuksiin liittyviä uhkia. Mikäli uhkamallinnusstrategia perustuu asetelmaan, jossa pyritään löytämään uhkia toisen ihmisen roolissa, lisätään pieleen menemisen mahdollisuuksia (Shostack, 2014, s. 43).

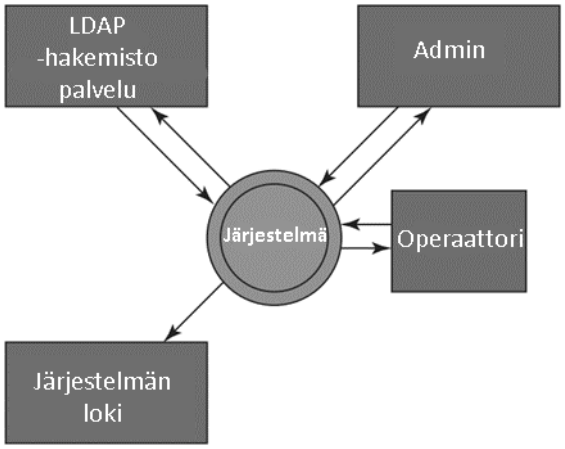
Vaikka uhka-arviot täytyy tehdä yksittäiselle laitteelle erikseen, on tärkeää käyttää apuna tietoa todellisista uhkista, jotta uhka-analyysi vastaisi todellista tilannetta. Yhdessä artikkelissa järjestelmän ominaisuudet oli jaettu tietoturvaominaisuuksien luottamuksellisuuden, saatavuuden ja eheyden mukaan. Tämän jälkeen oli mietitty, miten niitä pystytään vaarantamaan järjestelmässä. Uhkamallista tuli hyvin laaja uhkien kirjo, mikä vaikeutti uhkien todennäköisyyksien arviointia (Javaid, Sun, Devabhaktuni, & Alam, 2012).

Uhkamallinnusta tehdään useissa artikkeleissa järjestelmän kehittäjän näkökulmasta (Torr, 2005) tai kiinteänä osana ohjelmistokehityksen prosessia (Dhillon, 2011). Ohjelmistokeskeisessä lähestymistavassa ohjelmistosuunnittelijoiden oletetaan ymmärtävän kehitettävä ohjelmisto, joten uhkamallinnustyön aloittaminen kehittäjien kanssa ohjelmistosta vaikuttaa ideaaliselta vaihtoehdolta uhkien löytämiseksi (Shostack, 2014).

Uhkamallit suunnitellaan auttamaan tietoturvaominaisuuksien kehittämisessä. Ohjelmistolähtöisiä malleja on useita, kuten STRIDE, hyökkäyspuut, hyökkäyskirjastot ja yksityisyystyökalut. Malleissa käytetään monia kaavioita erilaisia mallinnustoteutuksia varten. Kaavioiden tarkoituksena on jakaa tietoa järjestelmän toiminnasta, että kaikilla uhkamallinnusta tekevillä olisi siitä samalainen käsitys. Paljon käytettyjä kaavioita ovat tietovuo-, UML-, tila- ja vuokaaviot. (Shostack, 2014)

Ohjelmistojärjestelmissä kaikkein yleisimmin käytetään tietovuokaavioita, jonka avulla pystytään tunnistamaan tiedon kulku järjestelmässä. Tämän mallin käyttöä on kuvattu esimerkiksi artikkeleissa (Dhillon, 2011; Myagmar ym., 2005; Torr, 2005). Tietovuokaavio on hyvä uhkamallinnuksessa siinä mielessä, että se esittää visuaalisesti kaikki vuorovaikutuskohdat, joista vihollinen voi hyökätä järjestelmään. Kaavio on hierarkkinen, joten voi valita kuinka yksityiskohtaisella tasolla järjestelmä esitetään.

Taso 0 esittää tietovuokaaviota järjestelmästä sen kontekstissa. Myöskään tiukkoja sääntöjä ei ole siitä, mitä kaavioon sisällytetään, vaan se tavallisesti riippuu työnjaosta ja kaavion luettavuudesta. Kaaviossa vuorovaikutuskohdat ovat keskeisimmässä roolissa, koska järjestelmään, johon ei voi olla vuorovaikutuksessa, ei voi myöskään hyökätä. Tietovuokaaviosta näkee myös selkeästi, kuinka tieto liikkuu järjestelmän läpi (Dhillon, 2011). Kuvio 8 esittää ylimmän tason tietovuokaaviota järjestelmästä sekä sen vuorovaikutuksia muihin järjestelmiin ja toimijoihin.



Kuvio 8. Tietovuokaavio järjestelmästä (Dhillon, 2011, s. 42)

Tietovuokaaviolla pystytään hyvin yksinkertaisesti kuvaamaan järjestelmä. Siinä ympyrät edustavat prosesseja. Kaksoisreunaiset ympyrät kuvaavat komplekseja prosesseja, jotka on tarkemmin kuvattu alemman tason kuvaajassa. Suorakulmiot edustavat käyttäjiä tai järjestelmiä, tyypillisesti ei-luotettuja tietolähteitä. Nämä ovat vuorovaikutuksissa mallinnettavan järjestelmän kanssa ja ovat kehitystiimin kontrollin ulkopuolella. Kaksoisviivat esittävät paikkaa, jossa tietoa on hetkellisesti tai pysyvästi. Tällaisia ovat esimerkiksi tiedosto tai jaettu muisti. Nuolet esittävät tietovuota muiden osien kanssa, kuten verkkoliikennettä, paikallista liikennettä tai vaikka luku- ja kirjoitustoimintaa. Nuolet osoittavat siis järjestelmän tulo- ja poistumiskohtia, joista yhteydenpito myös hyökkääjän kanssa tapahtuu.

Kuviossa 8 esitetty ylimmän tason uhkamallinnus on hyödyllinen, jotta ymmärretään ulkoisiin vuorovaikutuksiin kohdistuvat hyökkäykset. Se ei kuitenkaan tuo esille ulkoisista yksiköistä syvällä ohjelmistossa olevien vuorovaikutusten aukkoihin kohdistuvia

hyökkäyksiä. Tämän takia ohjelmiston uhkamallit vaativat kaavioita, jotka hajottavat järjestelmän riittävän pieniin yksityiskohtiin. Mitä monimutkaisempi järjestelmä on, sitä useampaa tasoa tietovuokaavioissa täytyy käyttää.

4.4 Sidosryhmien osallistuminen

On joitakin tutkimuksia, joissa tutkitaan asiakkaan osallistumisen vaikutuksia tuotekehitysprojektiin (Ilieva, Ivanov, & Stefanova, 2004). On myös kehitetty organisationaalinen tekniikka IPPD (integrated product and process development), joka antaa systemaattisen lähestymistavan tuotteen kehittämiseen sidosryhmien kanssa (Schmidt, 2013). Toisin kuin ketterät kehitysmenetelmät painottavat asiakkaan osallistumista, IPPD keskittyy tarvittavien sidosryhmien oikea-aikaiseen osallistumiseen. IPPD keskittyy parantamaan tuotteen laatua hyvin ajoitetulla yhteystyöllä läpi tuotteen kehityksen, paremmin vastaamaan sidosryhmien tarpeeseen. (Schmidt, 2013)

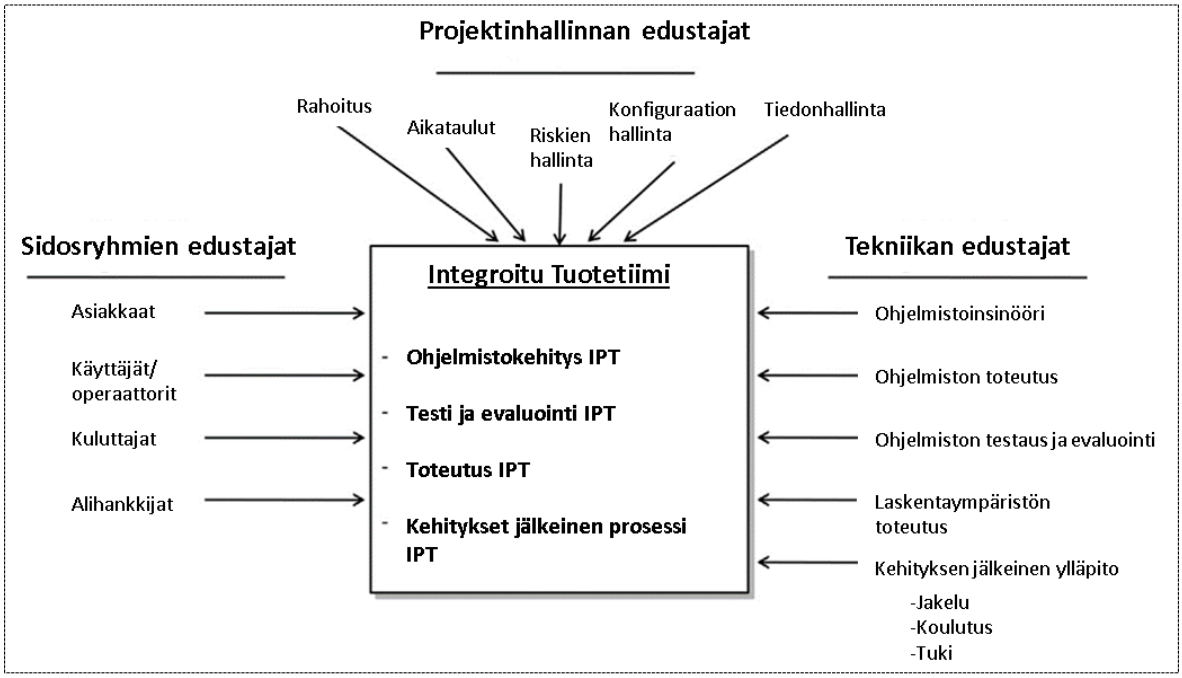
IPPD sisällyttää kaikki tekniset harjoitukset kehitysprosessin alkuun takaamaan, että vaatimukset on kerätty hyvin, ymmärretty ja määritetty. IPPD:n tarkoitus on kannustaa kehittäjiä miettimään tuotteen elinkaaren kaikkien sidosryhmien näkökulmia, jotta taattaisiin, että tuotteen arkkitehtuuri on resilientti muutoksiin teknisissä ja käyttöympäristöissä (Schmidt, 2013).

IPPD -filosofia takaa, että kaikki tekniset ja hallinnolliset organisaatiot ovat edustettuina integroidussa ohjelmistokehityksen tuotetiimissä SWE-IPT (software engineering integrated product team). Vaatimukset kehitetään aluksi yhdessä ohjelmiston tuotetasolla. Sitten vaatimukset hajotetaan alemmilla tasoilla ja katsotaan ohjelmistotuotteen arkkitehtuuria vasten. Tämä on erilainen verrattuna perinteiseen ohjelmiston kehityksen lähtökohtaan, jossa ohjelmistoanalyytikko vastaa vaatimusmäärittelytyöstä välittäen vaatimukset eteenpäin tuotteen suunnitteluun, implementointiin, testaukseen, evaluointiin sekä viimeisenä ylläpitoprosessin insinööreille. Tällainen toimintamalli aiheuttaa tietokatkoja, koska kommunikaatio ei toimi oikea-aikaisesti. (Schmidt, 2013)

Yleinen lähestymistapa IPPD:n toteuttamisessa on muodostaa monitieteinen tiimi kaikille tuote- ja jälkikehitysprosesseille. Tällöin teknisiin ongelmiin, vaatimusten

tasapainottamiseen sekä eri tiimien integrointiin voidaan vaikuttaa tehokkaasti. Teknisten ja hallinnollisten ryhmien osallistujat vaihtuvat tuotteen elinkaaren aikana, kun työt muuttuvat ohjelmiston kehitysvaiheiden mukaan. (Schmidt, 2013, ss. 12–13)

IPT:ia (integrated product teams) käytetään takaamaan, että sidosryhmät, projektin hallinta ja tekniset organisaatiot ovat edustettuna koko ohjelmistokehityksen ajan. SWE-IPT sisältää edustajia eri sidosryhmäyhteisöistä sekä ohjelmistokehityksen teknisistä ja hallinnollisista organisaatioista. Kuvio 9 esittää eri tahoja, jotka ovat tekemisissä useiden ohjelmistokehityksen integroitujen tuotetiimien kanssa (Schmidt, 2013, s. 13).



Kuvio 9. Ohjelmistokehitystiimin kokoonpano (Schmidt, 2013, s. 13)

Integroitu tuotetiimi on vastuussa tuotteen määrittelystä koko tuotteen elinkaaren vaiheiden ajan. Tyypilliseen integroituun tiimiin kuuluu edustajia ohjelmistotekniikasta, implementoinnista, testauksesta ja evalutoinnista. Lisäksi mukana on erikoistekniikan, kuten ylläpidon, turvallisuuden, logistiikan sekä asiakkaan ja hallinnon edustajat.

Integroitu tuotetiimi käyttää kehittyneitä menetelmitä ja työkaluja suunnittelussa, informaation jakamisessa, suunnittelun vertailuanalyysissä, mallintamisessa ja simuloinnissa, mitkä merkittävästi parantavat IPPD:n tehokkuutta. Jokainen IPT pyrkii

ajamaan seuraavia IPPD:n etuja: tuotteen toimitusajan lyhentäminen, järjestelmän ja tuotteen hinnan alentaminen, parempi riskin pienentäminen sekä tuotteen ja prosessin laadun parantaminen (Schmidt, 2013).

Integroituneiden tiimien käyttö tulee välttämättömäksi tietoturvakehityksen yhteydessä. Hankinta- ja tietoturvayhteisö ovatkin yrittäneet tehdä yhteistyötä hyvin laajojen järjestelmien hankinnassa sopivien tietoturvavaatimusten määrittämisessä. Suurin ongelma usein on ollut, että tietoturva-asiantuntijat esittelevät suunnitelmiaan käyttäen omia kaavoja ja kieltään. Kun taas hankkijayksikkö hankkii järjestelmät ja komponentit käyttäen eri malleja ja kieltä. (Abrams, Veoni, & Britton, 2004)

Vain kaikkein korkeimman asteen viittauksissa hankintamallit ja tietoturvayhteisön kieli ovat yhdistyneet hankinnan teon yhteydessä. Huolimatta tästä korkean tason määrityksestä, suurin osa hankintastrategioista on perustunut tuotetason määrityksiin ilman tietoturvaominaisuusien huomioon ottamista järjestelmäkokonaisuudessa. (Abrams ym., 2004)

Yksi lähestymistapa tietoturvaominaisuuksien määrittämisessä on ollut alhaalta ylös-menetelmä. Tällöin komponenttien tietoturvaominaisuudet määrittävät järjestelmän tietoturvaominasuudet (Abrams ym., 2004). Tämä komponenttien yhdistämisellä tehty tietoturvamääritys on ongelmallista, koska hyvin pienillä sisarusominaisuuksien eroilla on kausaalinen syy-yhteys monitoimittaja komponenttien yhdistämiseen kohdistuviin ongelmiin (Gamble & Hemenway, 1995).

Perusperiaatteeltaan tietoturvaominaisuuksien määrityksen ei pitäisi olla erilainen kuin järjestelmän muiden ominaisuuksien määritys. Kompromisseja täytyy tehdä, kun kaikkia määrityksia ja tavoitteita ei voida saavuttaa. Asiakkaan tulisi olla mukana erityisesti niissä päätöksissä, jotka koskevat kompromisseja. Tämä auttaisi integraattoria päättämään, kuinka kehittää määrityksiä ja tavoitteita samoin kuin järjestelmää. (Abrams ym., 2004)

5 Tutkimuksen toteutus

Ohjelmistokehitysprosesseja on tutkittu ja kehitetty paljon. Tutkimuksissa todetaan, että eri kehitysprosessit sopivat eri yhteyksiin eri tavoin (Lane, Boehm, Bolas, Madni, & Turner, 2010). Vahva tietoturva vaikuttaa ohjelmiston kehitystyöhön sen vaatimusmäärittelystä lähtien (H. Wang & Wang, 2003). Asiakkaan oletetaan osallistuvan ohjelmiston kehitystyöhön eri tavoin eri kehitysprosesseissa. Ohjelmistojen tietoturvallisuutta kehitetään ohjelmistosuunnittelijoiden toimesta monissa eri tutkimuksissa. Ohjelmiston tietoturvavaatimusten toteuttamiseen on kehitetty hyvin monenlaisia menetelmiä (Souag ym., 2016).

Vähän on tutkimusta siitä, miten asiakas voisi vaikuttaa tilaamansa järjestelmän tietoturvaan. Tietoturva on korostunut puolustusvoimille hankittavien ohjelmistojärjestelmien vaatimuksissa, ja tietoturvan toteutumista halutaan parantaa mahdollisuuksien mukaan myös asiakkaan puolelta. Tämä luku esittelee tutkimuksen toteutuksen tutkimusongelmineen ja -kysymyksineen, perustelee laadullisen tutkimuksen menetelmän valinnan sekä esittelee aineiston hankinnan ja sen analyysin.

5.1 Tutkimusongelma

Järjestelmien ominaisuudet monipuolistuvat niiden digitalisoitumisen myötä. Samoin järjestelmien verkottuminen laajentaa niiden käyttömahdollisuuksia. Järjestelmien monipuolisuus lisää myös kyberuhkien mahdollisuutta, minkä takia tietoturvaominaisuuksien tulisi kehittyä järjestelmien mukana. Jotta järjestelmälle saataisiin vahva tietoturva, se tulisi ottaa huomioon jo järjestelmän kehityksessä. Tässä työssä on tarkoituksena selvittää tietoturvan parantamiseksi käytettyjä prosessimalleja sekä tietoturvavaatimusmäärittelyä. Tietoturvan toteuttaminen vaatii eri sidosryhmien toimintaa, joten yhteistyöhön etsitään myös parannusehdotuksia.

Puolustusvoimien hankintaprosessi perustuu vesiputousmallin kaltaiseen menettelyyn, jossa hankintaprosessin eri vaiheet seuraavat toinen toistaan. Prosessi ei ota kantaa hankittavan tuotteen kypsyyteen, laajuuteen tai ominaisuuksiin. Hankintaohjeissa ei oteta

kantaa ohjelmistojärjestelmien kehitykseen, mutta käytännössä hankintaprosessissa vaatimukset tulee kirjoittaa ennen hankinnan aloitusta. Hankintojen kilpailutus perustuu hankittavalle tuotteille asetettuihin vaatimuksiin, joten toimittajat arvioivat järjestelmän hinnan vaatimusten perusteella. Tämä menettely johtaa koko järjestelmän kehitysprosessin vesiputousmallin kaltaiseen toimintaan ainakin asiakkaan puolelta. Koska hinta määräytyy vaatimusten perusteella, vaatimuksia ei haluta muuttaa kesken järjestelmän kehitysprosessin.

Puolustusvoimilla käytössä oleva menettely on kuitenkin hyvin erilainen verrattuna nykyisin paljon käytettyihin ohjelmistokehitysmenetelmiin. Niissä vaatimusmäärittely pidetään hyvin keveänä, ja sitä halutaan muuttaa ohjelmiston kehityksen edetessä. Lisäksi ohjelmistoa koskevista ratkaisuista halutaan keskustella asiakkaan kanssa kehitystyön aikana. Tietoturvan osalta tämä vaatii asiakkaalta tietoturvaosaamista, koska tietoturvavaatimukset tulisi alussa pystyä tekemään niin yleisiksi, että ne eivät sitoisi kehitettävää järjestelmää johonkin tiettyyn ratkaisuun. Kuitenkin niiden tulisi olla selkeitä, että toimittaja pystyisi niiden pohjalta kehittämään riittävän tietoturvallisen järjestelmän.

Tietoturvavaatimukset perustuvat kansallisiin lakeihin ja niistä kirjoitettuihin ohjeisiin. Monesti toimittajille annetut tietoturvavaatimukset ovat näitä yleisiä ohjeita, joten niiden tulkinta on jätetty järjestelmätoimittajien vastuulle. Vaatimusten soveltaminen yksittäisiin järjestelmiin aiheuttaa jatkuvasti ongelmia, koska toimittajilla on hyvin harvoin tietoa kehitettävän järjestelmän ympäristöstä tai käytöstä. Väärinkäsityksiä tulee väistämättä, ja se vaikuttaa järjestelmän tietoturvaratkaisuihin.

Tietoturvaominaisuuksia auditoidaan jonkin verran järjestelmän kehitysprosessin aikana, mutta enimmäkseen auditointeja tehdään tuotteelle sen valmistumisen jälkeen. Auditointien perusteella järjestelmien tietoturva osoittautuu usein puutteelliseksi, minkä vuoksi tietoturvaominaisuuksia joudutaan lisäämään järjestelmiin. Tämä aiheuttaa kalliita muutostöitä ja huonontaa järjestelmien käytettävyyttä. Lisäksi aina ei voida taata, että jälkeenpäin lisätyt tietoturvaratkaisut varmistaisivat riittävän tietoturvatason järjestelmiin.

Puolustusvoimat käyttävät ulkopuolisia tietoturva-asiantuntijoita auditointien lisäksi myös toimittajien tukena järjestelmien kehitystyössä. Ulkopuolista asiantuntija-apua tarvitaan,

koska puolustusvoimien tietoturva-asiantuntijat ovat niin kiireisiä, etteivät he ehdi osallistua järjestelmien kehitystyöhön. Hankintaan nimetyissä projektihenkilöissä ei siis ole asiakkaan puolelta tietoturva-asioihin perehtyneitä työntekijöitä. Jotta usean eri sidosryhmän osallistuminen tuotteen kehitykseen toisi haluttuja tuloksia, yhteistyötä tulisi tehdä suunnitellusti. Koska toimittaja on vastuussa tuotteen toteutuksesta ja kehitystyöstä, yhteistyötä eri toimijoiden välillä tulisi kehittää toimittajan näkökulmasta järkevällä tavalla.

5.2 Tutkimuskysymykset

Tutkimusongelman perusteella laadittiin kolme tutkimuskysymystä tutkimuspäämäärien selkeyttämiseksi. Tutkimuskysymykset kohdistuvat tietoturvan kehittämiseen ohjelmistokehitysprosessissa, tietoturvavaatimusten määrittämiseen sekä sidosryhmien välisen yhteistyön kehittämiseen. Tutkimuskysymykset on esitelty alla.

Miten asiakkaan hankintaprosessia tulisi kehittää, jotta se tukisi vahvaa suojausta vaativien ohjelmistojärjestelmien kehitystä?

Ohjelmistonkehityksessä käytetään erilaisia suunnitteluprosesseja, joiden tiedetään sopivan eri tavoin tietoturvan kehittämiseen. Parantaakseen hankittavan ohjelmiston tietoturvaa, asiakas voisi ottaa toimittajan käyttämän suunnitteluprosessin huomioon hankintaprosessin eri vaiheissa. Tällä hetkellä puolustusvoimien hankintamenettely ohjaa asiakasta vesiputousmallin mukaiseen toimintaan, kun taas ohjelmistoyritysten toiminta perustuu suurelta osin ketterien kehitysmenetelmien soveltamiseen kehitystyössä. Ristiriitaisia odotuksia tulee puolin ja toisin, koska asiakas on vaatimusmäärittelyssä laatinut tietoturvavaatimukset valmiiksi. Toimittaja taas haluaa keskustella vaatimuksista ja sopivista tietoturvaratkaisuista koko järjestelmän kehitystyön ajan. Tutkimuksessa halutaan selvittää, miten asiakkaan tulisi toiminnassaan ottaa huomioon toimittajan suunnitteluprosessi sekä kuinka asiakkaan omaa suunnitteluprosessia tulisi kehittää tietoturvan kehitystyön tukemiseksi.

Mitä vahvaa suojausta vaativien ohjelmistojärjestelmien tietoturvavaatimuksissa tulisi ottaa huomioon, jotta kehitettävien järjestelmien tietoturva täyttäisi lakisäätäiset vaatimukset?

Hankintasopimuksissa tietoturvavaatimukset on annettu yleisellä tasolla. Järjestelmän toimittajat eivät pysty tulkitsemaan yleisiä vaatimuksia kulloinkin kehitettävänä olevaan ohjelmistoon. Tällöin suunnittelijat eivät pysty kehittämään ohjelmistoja annettujen vaatimusten mukaisesti tai vaatimusten toteuttaminen tuo järjestelmän kehitystyöhön viiveitä ja lisäkustannuksia. Tietoturvavaatimusmäärittelyä tulisi kehittää, jotta toimittajat saisivat toteutuskelpoiset vaatimukset ja pystyisivät siten paremmin suunnittelemaan sopivat tietoturvaratkaisut kehitettävään ohjelmistoon. Tutkimuksessa yritetään selvittää, kuinka tietoturvavaatimukset tulisi laatia, jotta toimittajat pysyisivät toteuttamaan ohjelmiston tietoturvan halutulle suojaustasolle.

Miten eri sidosryhmien välistä yhteistyötä tulisi kehittää, jotta se edistäisi tietoturvan rakentamista ohjelmistosuunnittelussa?

Toimittajat pyrkivät kehittämään järjestelmiä annettujen vaatimusten perusteella. Kuitenkin toimittajilta tulee paljon kysymyksiä kehitystyön aikana tietoturva-asioista. He haluavat varmistaa, että tietoturvaratkaisut ovat riittäviä halutulle suojaustasolle, tai heidän tulkintansa tietoturvavaatimuksista on oikein. Mikäli tietoturvavaatimukset olisivat selkeät ja toteutuskelpoiset, kysymysten määrä oletettavasti pienenisi. Kuitenkin tietoturvan rakentaminen on monipuolinen työ, ja puolustusvoimat ovat käyttäneet ulkopuolisia tietoturva-asiantuntijoita ohjelmistokehitystä sisältävissä hankinnoissa. Asiantuntijat ovat auttaneet soveltamaan Katakria ja VAHTI-ohjeita sekä ovat tehneet järjestelmiin tietoturva-auditointeja. Tutkimuksessa halutaan selvittää, kuinka yritykset ovat kokeneet ulkopuolisten asiantuntijoiden käytön ja kuinka yhteistyötä tulisi kehittää eri toimijoiden välillä.

5.3 Taustaa tutkimusmenetelmän valinnalle

Tutkimuksen tarkoitus on löytää keinot, joilla tietoturvaominaisuuksia voidaan parantaa puolustusvoimille hankittaviin järjestelmiin kustannustehokkaasti. Yritykset toteuttavat tietoturvaratkaisut ohjelmistoihin tietoturvavaatimusten perusteella. Koska tarkoituksena on saada yritysten näkemyksiä, miten puolustusvoimat voisivat edesauttaa tietoturvan kehittämistä tilaamiinsa ohjelmistoihin, tutkimuksen luonteeseen sopii parhaiten laadullinen tutkimus. Laadullisessa tutkimuksessa perehdytään tutkittavasta aiheesta aikaisemmin tehtyyn tutkimukseen ja niistä johdettuihin teorioihin. Tutkimukseen kuuluu myös aineiston keräys sekä tutkijan omat pohdinnat ja päättelyt, joita hän tekee koko tutkimuksen ajan (Töttö, 2004, ss. 9–20). Laadullista tutkimusta kuvaa hyvin alla oleva listaus (Eskola & Suoranta, 1998, s. 15):

- Aineistonkeruumenetelmä
- tutkittavien näkökulma
- harkinnanvarainen tai teoreettinen otanta
- aineiston laadullis-induktiivinen analyysi
- hypoteesittomuus
- tutkimuksen tyylilaji ja tulosten esitystapa
- tutkijan asema ja
- narratiivisuus

Laadullisessa tutkimuksessa aineistona käytetään usein tekstiä. Tutkija voi hankkia aineiston itse tai saada sen valmiina (Eskola & Suoranta, 1998, s. 15). Kvalitatiiviselle tutkimukselle on tyypillistä, että tutkijan tulisi yllättyä tai oppia uusia asioita tutkimuksen edetessä. Jotta uuden oppiminen olisi mahdollista, tutkimuskohteesta tehdyt ennakko-oletukset tulisi pohtia ja ennen kaikkea tiedostaa tutkimuksen alkuvaiheessa ikään kuin esioletuksina (Eskola & Suoranta, 1998, ss. 19–20). Vaikka laadullisessa tutkimuksessa ei käytetä hypoteeseja, kuitenkin tutkija voi muodostaa työhypoteeseja siitä, mitä analyysissa voisi tulla esille. Tutkimuksen luonteen vuoksi aineiston hankkimisella ja analyysilla ei ole kuitenkaan tarkoitus todistaa hypoteeseja oikeiksi. Odotuksena on, että työhypoteesien avulla pystyttäisiin keksimään uutta tietoa. Laadulliselle tutkimukselle on tyypillistä, että

tutkimusaineisto antaa pikemmin uutta potkua tutkittavan aihealueen ajattelulle ja avaa näkökulmia aiheeseen, kuin että vahvistaa jo tiedettyä tietoa (Saaranen-Kauppinen & Puusniekka, 2006) .

Kun tutkija analysoi ja tulkitsee keräämäänsä aineistoa, hänen tulisi samalla pystyä luomaan teoriaa tutkittavasta aiheesta. Induktiivisessa lähestymistavassa teoria rakennetaan lähtökohtaisesti aineistosta. Esimerkiksi analyysiyksiköitä ei määritetä ennen aineiston keruuta (Eskola & Suoranta, 2008, s. 83). Induktiivisessa lähestymistavassa eteneminen tapahtuu yksittäisistä havainnoista aina yleisempiin teorioihin tai väitteisiin (Eskola & Suoranta, 1998, s. 83). Kun tutkimusta tehdään aineistolähtöisesti, pääpaino on tietenkin aineistossa. Induktiivisen lähestymistavan lähtökohtana ei siis ole teorian tai hypoteesien testaaminen. Tutkija ei myöskään määrää sitä, mikä on tärkeää (Hirsjärvi ym., 2004; Saaranen-Kauppinen & Puusniekka, 2006). Ikään kuin vastakohtana induktiiviselle lähestymistavalle on teorialähtöisyys eli deduktiivisuus. Siinä valmis teoria tai malli testataan vain erilaisessa yhteydessä kuin se aikaisemmin on ollut käytössä. Luonnontieteissä teorialähtöistä tutkimusta on tehty paljon. Vaikka kvalitatiivisessa tutkimuksessa induktiivisuus on hyvin tyypillistä, tutkijalta vaaditaan taitoa ja hyvää itsekuria, jotta tutkimus pohjautuisi pelkästään aineistoon. Tällöin ennakkokäsitykset ja mahdolliset jo olemassa olevat teoriat suljettaisiin pois (Saaranen-Kauppinen & Puusniekka, 2006). Jos pitäydytään vain aineiston analyysissä, tutkimus on täysin irrallinen muista aiemmista saman alan tutkimuksista (Eskola & Suoranta, 2008).

Grounded theory (GT) on myös aineistolähtöinen tutkimusmetodi. Se kuitenkin olettaa, että tutkijalla on tutkittavasta aihealueesta jo kokemuksia sekä aiheeseen liittyvää teoreettista jäsentelyä. GT:n mukaan niillä on vaikutuksia tutkimusaineiston analyysiin. Vaikka tässä tutkimuksessa aineistolähtöisyys on tärkeää, GT:n mukainen lähestymistapa sopii paremmin tutkittavaan aiheeseen. GT-lähtöisessä päättelyssä voidaan puhua abduktiosta eli teoriasidonnaisuudesta, mikä on siis induktiivisen ja deduktiivisen lähestymistavan sekoitus (Eskola & Suoranta, 2008). Vaikka abduktiossa tutkijan pohdinta on tärkeintä, se kuitenkin olettaa, että johtopäätöksiin vaikuttavat aikaisemmat kokemukset ja tutkittavasta aiheesta kerätyt teoreettiset aineistot. Tuloksiin vaikuttavat siis GT:n mukaan konteksti ja tutkijan subjektiivisuus (Saaranen-Kauppinen & Puusniekka, 2006).

Vaikka tutkijan kokemukset vaikuttavat tutkimukseen, siihen tulisi kuitenkin suhtautua mahdollisimman objektiivisesti. Ainakin tulisi selvittää, mitä uskomuksia, asenteita tai arvostuksia tutkittavaan aihealueeseen liittyy. Ne tulisi sulkea pois tutkimuksesta. Tutkimus voidaan siis saada objektiivisemmaksi tunnistamalla omat subjektiivisuudet käsiteltävään aiheeseen (Eskola & Suoranta, 2008).

Koska tietoturva toteutetaan ohjelmistoihin ohjelmistolähtöisesti, tärkeää on saada selville eri ohjelmistotoimittajien näkökulmat. Tutkimuksella on kuitenkin tarkoitus kehittää puolustusvoimien toimintaa, joten tutkittavien valinnat kohdistuvat ohjelmistoyritysten henkilöihin, joilla on käsitys tietoturvan toteuttamisesta puolustusvoimien hankkimiin ohjelmistoihin.

Eri yrityksillä on erilaisia haasteita tietoturvallisten ohjelmistojen toteuttamisessa. Tarkoitus ei ole pelkästään selvittää ongelmien laatua, vaan myös saada selville, miten tietoturvan rakentamista tulisi kehittää yritysten näkökulmasta. Koska tutkimuksessa halutaan asiantuntijoiden mielipiteistä, haastattelututkimus sopii hyvin aineiston keruun menetelmäksi. Siinä henkilö voi kertoa omin sanoin kokemuksiaan ja näkemyksiään asiaan. Toisaalta työntekijät pääsääntöisesti ovat niin kiireisiä, että paneutuminen asioiden pohdintaan voi olla parempaa, jos haastattelija on läsnä ja haastatteluun on ennalta sovittu kiinteä aika.

Vaikka suurella osalla yrityksistä liiketoiminta-ala on informaatioteknologia, ohjelmistokehitys ja tietoturvan toteuttaminen tuotteisiin voi olla hyvinkin erilaista. Lisäksi puolustusvoimien hankintaorganisaatio on laaja, joten yhteisestä hankintaohjeesta huolimatta hankintasopimuksia ja toimintatapoja on monenlaisia. Oli siis vaikea tietää edeltä käsin, mihin asioihin haastattelut tulisivat painottumaan. Lähtökohtaisesti ohjelmistojen tietoturvan toteuttaminen perustuu tietoturvavaatimuksiin, mutta kehitysprosessissa voisi olla myös muita vaikeuksia kuin tietoturvavaatimuksiin liittyvät ongelmat. Ennen haastatteluja ei myöskään ollut tietoa, miten yritysten edustajat suhtautuisivat haastatteluihin yleensä. Kehuisivatko he tutkijalle omaa tietoturva-alan osaamistaan vai pitäisivätkö he haastattelua enemmän ajanhukkana? Tällöin suostuminen olisi tapahtunut vain sen takia, että yritykset haluavat säilyttää hyvät suhteet asiakkaaseen.

Näistä lähtökohdista katsottuna teemahaastattelu vaikutti sopivan väljältä haastattelumenetelmältä. Siinä haastattelu koostuu yksityiskohtaisten kysymysten sijaan ennalta sovituista aihealueista eli teemoista. Menetelmä ei ota kantaa, kuinka syvällisesti tai kattavasti aihealueista keskustellaan. Haastattelu etenee teemojen avulla, mutta tutkittaville tärkeiden painotusten mukaan, jolloin tutkijan näkökulma haastattelussa pienenee (Hirsjärvi & Hurme, 2008, s. 48) Haastattelijalla on tukilista käsiteltävistä teema-alueista, jotta kaikissa haastatteluissa tulisi käsiteltyä samat teemat. Tästä syystä teemahaastattelua voidaan pitää puolistrukturoituna haastatteluna. Toisaalta muissa puolistrukturoiduissa haastatteluissa kysymykset ja jopa niiden muodot ovat samat. (Hirsjärvi & Hurme, 2008, s. 48)

Teemat mietitään aiempien tutkimusten ja aihepiiristä kerätyn tiedon pohjalta, joten haastattelu on strukturoidumpi kuin avoin haastattelu. Vaikka kaikkien haastateltavien kanssa keskustellaan samoista teemoista, niiden käsittely on joustavaa. Esimerkiksi teemoja voidaan käsitellä eri järjestyksessä (Eskola & Suoranta, 1998, ss. 86–87). Teemahaastattelussa pyritään saamaan aikaan ennen kaikkea keskustelua ja puheelle annetaan tilaa. Tällöin haastateltavien tulkinnat ja merkitykset asioista saadaan esille (Puusniekka & Saaranen-Kauppinen, 2006).

Vaikka haastattelu koostuu ennalta sovituista teemoista ja haastattelurungosta, haastattelussa on tarkoitus painottaa haastateltavalle sopivia aiheita. Menetelmän valinalla on tärkeä merkitys tutkimuksen onnistumiseksi. Teemahaastattelulla pyritään saamaan keskustelu, jossa yrityksen edustaja antaa rehellisiä mielipiteitä puolustusvoimien osallistumisesta tietoturvan toteuttamiseen sekä antaa näkemyksiä tietoturvavaatimusten tai tietoturvan suunnittelun ja toteutuksen kehittämiseen. Haastattelurunkoon muodostetaan kysymyksiä, joilla saataisiin selville hankintoihin liittyvän vaatimusmäärittelyn kehityskohteita. Tämän lisäksi kysymyksiä tehdään vaatimusmäärittelyn ulkopuolelta, jotta saataisiin kokonaiskäsitys tietoturvan kehitykseen liittyvistä ongelmista.

GT eroaa muista laadullisen tutkimuksen tavoista vain analyysissa. Tiedonkeruu voi tapahtua monin tavoin, vaikka teorian rakentaminen nimenomaan haastatteluaineistosta onkin yleistä. Aineiston keruussa noudatetaan usein saturaation periaatetta eli aineistoa

kerätään, kunnes aineisto ei enää tarjoa uutta ainesta kehitteillä olevaan teoriaan. (Seale, Gobo, Gubrium, & Silverman, 2004). Vaikka tässä tutkimuksessa ei pystyttäisi luomaan uutta teoriaa, haastatteluaineistosta pyritään löytämään tekijöitä tietoturvan kehittämiseen. Lisäksi ohjelmistokehitysprosessia pyritään mallintamaan puolustusvoimien ja yritysten toimintoihin sopivaksi.

5.4 Aineiston hankinta

Tutkimuksen kohderyhmänä ovat puolustusvoimille ohjelmistoja tekevien yritysten edustajat. Tutkimuksen kannalta olennaista on, että haastateltavat olisivat joko itse olleet kehittämässä tietoturvallisia ohjelmistoja puolustusvoimille tuleviin tuotteisiin tai olisivat kehitystyössä hyvin läheisesti mukana. Lisäksi haastateltavien tulisi tietää asiakkaan kanssa tehtävistä sopimuksista tai ainakin sopimuksissa käsiteltävistä tietoturvaominaisuuksista. Keskustelimme työpaikallani sopivista yrityksistä esimiesten ja työkavereiden kanssa. He antoivat sopivien yritysten yhteyshenkilöiden nimet ja yhteystiedot. Jotkut työkavereista ovat muutenkin lähes päivittäin tekemisissä yritysten kanssa, joten he kertoivat yhteyshenkilölle, että tällaiseen tutkimukseen liittyvä haastattelupyyntö olisi tulossa.

Sähköpostiviestit lähetettiin valittujen yritysten yhteyshenkilöille. Sähköpostiviestissä pyydettiin yritystä suostumaan haastatteluun ja valitsemaan sopiva tai sopivia haastateltavia. Koska haastatteluaihe oli aika erityinen, yritysten annettiin itse miettiä sopivat haastateltavat. Todettiin, että yrityksen sisällä parhaiten tiedetään, kenelle aihealue olisi tuttu ja kuka olisi sopiva haastateltava. Sähköpostiin lisättiin haastattelun runko, jotta haastateltavan valinta olisi helpompaa. Näin haastattelija tiesi ennen suostumustaan, mitä aiheita haastattelu tulisi koskemaan. Sähköposti on liitteessä B.

Homanin (1991) mukaan haastateltava suostuu haastateltavaksi haastattelua koskevan asianmukaisen tiedon pohjalta. Lisäksi tutkimuksen tarkoituksesta ja luonteesta tulee antaa riittävä tieto, että mukana olevat ihmiset voivat päättää, osallistuvatko siihen vai kieltäytyvätkö siitä. Sähköposti lähetettiin yhdeksään yritykseen, ja mikäli vastausta ei kuulunut muutaman päivän sisällä, yhteyshenkilölle soitettiin. Kahdeksan yritystä lupautui

osallistumaan haastatteluun. Mikäli postin vastaanottaja ei itse osallistunut haastatteluun, hän välitti lähetetyn postin haastateltavalle tai antoi hänen yhteystiedot, jotta haastattelusta voitiin sopia henkilön kanssa suoraan. Yhdestä yrityksestä oli kaksi haastateltavaa, toisessa yrityksessä tehtiin yhtä aikaa haastattelu kahdelle henkilölle. Lopuissa yrityksissä haastateltiin vain yhtä yrityksen edustajaa. Kuvioon 10 on kerätty tiedot haastatelluista henkilöistä.

	Päivämäärä	**Yritys**	**Tehtävänimike**	**Yhteydenotot**
X1	11.10.2016	Airbus Defence and Space Oy	Chief Operating Officer	puhelu, sähköposti
X2	25.10.2016	Atea Oy	Network Architect	sähköposti, soitto, viesti
X3	10.11.2016	Insta Group Oy	Tuotepäällikkö	sähköposti
X4	15.11.2016	Patria Oyj	Laatupäällikkö	sähköposti, soitto
X5	8.11.2016	Accenture Oy	Technology Consultant Senior Manager	sähköposti
X6	9.11.2016	Combitech Oy	Project Manager	sähköposti
X7	9.11.2016	Combitech Oy	Unit Manager	-
X8	9.11.2016	Nixu Oyj	Principal Security Consultant	sähköposti
X9	21.11.2016	Bittium Oyj	Director, Corporate Security	sähköposti
X10	21.11.2016	Bittium Oyj	Senior Design Engineer	-

Kuvio 10. Tiedot haastatteluun osallistuneista henkilöistä taulukoituna

Haastattelut sovittiin siis joko sähköpostilla tai soittamalla, kun yhteyshenkilö oli ilmoittanut yhteystiedot. Kaikille haastateltaville ehdotettiin, että haastattelu voitaisiin tehdä heidän tiloissaan, mutta kahdessa tapauksessa haastattelu oli sujuvampaa järjestää puolustusvoimien tiloissa. Haastattelut tehtiin Helsingissä, Tampereella, Oulussa ja Jyväskylässä. Helsingissä tehtiin yksi haastattelu, muilla paikkakunnilla kaksi.

Kaikki haastattelut pidettiin kasvotusten, jotta keskusteluissa mahdolliset suojaustasojen ylitykset eivät aiheuttaisi ongelmia. Kuitenkin haastattelujen alussa sovittiin, että asioista tulisi keskustella julkisella tasolla. Jotta keskusteluista ei tulisi väkinäistä ja liian

harkitsevaa, haastattelun alussa todettiin, että keskustelua pystyisi muokkaamaan vielä haastattelun jälkeen. Esimerkiksi, jos haastateltava tulisi puhuneeksi yritykselle luottamuksellista tietoa, se voitaisiin poista haastattelusta. Lisäksi ilmoitettiin, että litteroinnin yhteydessä haastattelumateriaalista poistetaan tiedot, jotka ylittävät puolustusvoimien kannalta julkisen tiedon rajat. Haastattelua ennen myös kerrottiin, että jos joku asia ei tule haastattelussa mieleen, mutta pitää sitä tärkeänä aihealueeseen liittyen, haastattelijaan voisi ottaa yhteyttä ja kertoa asiasta myöhemmin. Pyrittiin siis siihen, että vaikka haastattelu olikin kertaluontoinen, haastateltavat eivät kokisi painetta asioiden muistamisesta tai keskustelun suojaustason ylityksestä. Kaikissa haastatteluissa puhe tallennettiin MP3-tallentimella. Suostumus tallentamiseen kysyttiin aina haastattelun alussa.

Ensimmäinen haastattelu oli ikään kuin esihaastattelu. Haastateltavalle kerrottiin, että tämä on ensimmäinen haastattelu ja haastattelun sisältöjä voidaan ehkä joutua muokkaamaan haastattelun jälkeen seuraavia haastatteluja varten. Haastattelu pidettiin kuitenkin aivan oikealle haastateltavalle. Haastattelun jälkeen haastattelurunkoon tehtiin pieniä muokkauksia, ennen kuin se lähetettiin muille haastateltaville. Muokkauksista huolimatta haastattelurungon sisältö ja haastatteluissa käyty keskustelu poikkesivat hieman toisistaan.

Haastatteluun oli varattu aikaa yksi tunti, mutta osa haastatteluista kesti enemmän kuin tunnin verran. Kun tietty määrä aineistoa on kerätty, Saaranen-Kauppisen & Puusniekan (2006) mukaan aiheet alkavat toistua ja voidaan muodostaa peruskuvio tutkitusta kohteesta. Tätä kutsutaan saturaatioksi. Samoja asioita haastatteluissa alkoi nousta esille heti ensimmäisten haastattelujen kohdalla, mutta saturaatiota alkoi tapahtua vasta puolen välin jälkeen. Haastattelurunogossa painottui ennen kaikkea vaatimusmäärittelyn kehittäminen. Kun haastatteluja kertyi enemmän, haastattelukysymyksissä mainitut kehitysprosessi ja yhteistyö alkoivat nousta isoina kokonaisuuksina myös esille.

5.5 Aineiston analyysi

Analyysia tulee miettiä jo siinä vaiheessa, kun aineistoa kerätään. Mikäli analyysitapaa on mietitty ennen aineiston keruun aloittamista, tulevan analyysin voi ottaa huomioon ja pitää ohjenuorana aineiston keräämisen aikana. Analyysitapa kannattaa pitää mielessä myös suunniteltaessa aineiston purkamista (Hirsjärvi & Hurme, 2008, s. 135).

Hirsjärvi & Hurme (2008) toteavat myös, että aineiston analyysi on työlästä ja aikaa vievää. Pohdintaa täytyy tehdä, että aineiston ymmärtää oikein. Lisäksi aineistoa pitäisi pystyä jäsentämään ja löytämään siitä merkityksiä. Kuitenkin analyysivaihe on myös mielenkiintoinen, koska aihealueesta nousee monenlaisia ajatuksia ja erilaisia vaihtoehtoja tutkimusongelmaan.

Kvale (1996) esittelee erilaisia lähestymistapoja analyysiin. Niistä kahta mielenkiintoista tapaa käytettiin haastatteluissa. Ensimmäisessä lähestymistavassa haastateltavat itse havaitsevat uusia yhteyksiä asioiden välillä tai huomaavat merkityksiä. Tässä lähestymistavassa tutkija ei tee mitään tulkintaa. Toisessa tavassa haastattelija tiivistää ja tulkitsee haastateltavan kuvausta jo haastattelun aikana. Tällöin haastattelija toteaa oman käsityksensä haastateltavan puheesta haastattelutilanteessa, jolloin haastateltava voi korjata ajatusta. Mikäli tulkinta on pielessä, keskustelu jatkuu niin pitkään, että haastattelija on saanut oikean käsityksen asiasta. Tällaista tyyliä haastattelussa voidaan kutsua itseään korjaavaksi menettelyksi. (Kvale, 1996)

Kun haastattelut olivat valmiit, ne kirjoitettiin puhtaaksi eli litteroitiin. Litterointi tehtiin ELAN Linguistic Annotator -ohjelmalla. Tässä analyysissä ei ollut tarvetta yksityiskohtaiselle litteroinnille, joten sitä ei toteutettu. Kuitenkin litterointi tehtiin sanasta sanaan, jättäen pois tauot, yskähdykset ja merkityksettömät täytesanat. Lisäksi asiakasta ja toimittajaa koskeva luottamuksellinen tieto poistettiin. Tiedot koskivat tuotteiden, organisaatioiden ja yritysten nimiä sekä henkilöiden nimiä ja ammattiarvoja. Ohjelmistojen, laitteiden, laiterakenteiden ja tekniikoiden yksityiskohtaiset tiedot poistettiin myös. Lisäksi tarkat lukuarvot ja arkaluontoinen toimintaan kohdistuva tieto poistettiin. Litteroinneissa haastateltavan ja haastattelijan tekstien eteen laitettiin merkit erottamaan ne toisistaan. Ensimmäinen litterointi tehtiin esihaastattelulle, ja sillä testattiin

myös koodausohjelman käyttöä. Kun se vaikutti onnistuvan, haastattelujen litterointia jatkettiin. Litteroinnin aikana alkoi hahmottua usein toistuvia aihealueita ja miten eri aiheet liittyivät toisiinsa.

Koska ei oltu aivan vakuuttuneita, vastasiko haastattelumateriaalin aiheet tutkimuskysymyksiin, muutamat litteroinnit luettiin läpi kysymyksiä miettien. Kun jokin aihe alkoi toistua tekstissä useita kertoja, siitä muodostettiin yläkategoria eli teema. Näin pikkuhiljaa aineistosta tuli esille kuusi tietoturvaan vaikuttavaa tekijää. Hankintaprosessin kehittämisen kohdasta tuli esille, että toimittajien käyttämät prosessit poikkesivat huomattavasti puolustusvoimien hankintaprosessista. Kehitysehdotuksia tuli prosessin eri vaiheisiin. Tietoturvavaatimusten kehittämisen näkökulmasta nousivat selkeästi esille tietoturvavaatimusten perusteet sekä järjestelmän tekninen ja käyttöympäristö tietoturvan kannalta. Yhteistyötä koskevasta tutkimuskysymyksestä nousi esille asiantuntija- ja kollaboraatioavun sekä tietoturvan validoinnin kehittäminen.

Vaikka analyysi pyrittiin toteuttamaan induktiivisesti ja haastattelijoiden annettiin tuoda esille näkemyksensä, subjektiivisuus eli tutkijan omat näkemykset sekä aiheesta kirjoitettu teoria jotenkin hämmensivät merkityksien ja eri ilmiöiden yhteyksien jäsentelyä. Vasta useamman henkilön kanssa käyty keskustelu vakuutti, että analyysissä esiin tulleet asiat voivat olla tosiasiallisia ja ne voidaan kirjoittaa niin kuin haastattelijat ovat ne kertoneet.

Aineiston purku aloitettiin hahmoteltujen teemojen avulla. Työkaluna käytettiin atlas.ti-ohjelmaa. Kun joku yksittäinen havainto alkoi toistua aineistossa useita kertoja, sille annettiin koodisana. Kun useammalla koodisanalla oli yhteisiä merkityksiä, niistä muodostettiin lopullisesti teema. Näin teemat ja niihin liittyvät havainnot alkoivat jäsentyä koodien avulla selkeiksi kokonaisuuksiksi tulosten käsittelyä varten. Koodiluettelo ja koodien toistuvuus on esitetty liitteessä C.

6 Kokemukset tietoturvan toteuttamisesta

Tutkimuksessa oli mukana kymmenen haastateltavaa kahdeksasta eri yrityksessä. Vastaukset painottuivat sen mukaan, missä tehtävässä haastateltava oli ollut tietoturvan kehitystyössä. Tietoturva-asiat olivat kaikille hyvin tuttuja ja tietoturvaan panostaminen nähtiin tärkeänä painopistealueena yrityksissä.

Haastateltavat olivat kokeneita ohjelmistosuunnittelijoita, tietoturva-asiantuntijoita, projektipäälliköitä ja puolustusvoimien kanssa sopimuksia neuvottelevia yritysjohtajia. Tietoturvan kehittäminen oli siten tullut tutuksi joko ohjelmiston kehitystyössä, kehitysprojektin vetäjänä tai sitten vaatimusmäärittelytyössä. Useimmilla haastateltavilla oli ainakin kymmenen vuoden kokemus puolustusvoimien eri organisaatioiden kanssa tehdystä yhteistyöstä. Joukossa oli myös haastateltavia, jotka olivat tehneet puolustusvoimien kanssa yhteistyötä esimerkiksi vain yhden järjestelmäkokonaisuuden kehittämisen verran.

Kaikki haastateltavat näkivät tietoturvaan panostamisen tärkeänä, ja yrityksissä oli sisäisesti mietitty keinoja tietoturvan kehittämiseen. Haastateltavat myös pitivät tietoturvaa yhtenä tärkeänä osa-alueena tulevaisuudessa.

Tämäki on kehittyvä prosessi, että jatkuvasti pyritään enemmän hakemaan tietoa talon sisään ja näin pois päin. Varmasti on yks painopistealue kaikissa IT-alan yrityksissä tulevaisuutta ajatellen, jotka varsinki tekee jotain verkottuvaa softaa ja näin pois päin.

Yrityksissä oli sisäisesti mietitty, kuinka tietoturvaominaisuudet ja tietoturvaosaaminen olisivat mukana ohjelmistonkehitystyössä. Tietoturva nähtiin kiinteänä osana kehitystyötä.

Ja tietysti meillä on porukkaa käyny kaikenlaisia tietoturvasertifikaatteja suorittamassa ja ympärimaailmaa käyny näis kaikenlaisissa koulutuksissa. Et meiltä on valittu ne ihmiset, jotka on eniten innostunu projektin sisällä, siis eniten innostunu aiheesta ja jotka sitte haluaa oikeesti perehtyä asioihin… Et me on myöski todettu se, että projektin sisällä pitää olla TITU-ymmärtäväisiä ihmisiä. Koska me aluks kuviteltiin, että se riittää, että mehän ollaan myöskin tietoturvatalo, eli meillä on oma yksikkö, jossa on kaikki ihmiset tietoturva-asiantuntijoita. Mut ei se niin toimi, että me käyään kysymässä joka toinen päivä tai kerran

viikossa joku juttu sieltä. Et kyllähän sen ihmisen pitää olla siinä projektissa, päivittäisessä työssä mukana, jotta se on silmät auki koko ajan. Ja sit se tajuaa, kun näkee jotain, että toi muuten ei nyt ollukaan hyvä juttu. Ni, et tää on ollu yks oleellinen kasvaminen tässä, että...

Yritysten edustajat olivat hyvin motivoituneita tutkimuksen aiheeseen, ja haastatteluihin osallistuttiin aktiivisesti. Tutkimuksen tulokset esitellään teemoittain tutkimuskysymysten perusteella. Teemoihin ja havaintoihin saadaan tukea myös kirjallisuudesta. Järjestelmien kehitysprosessissa on toimittajien antamia näkemyksiä prosessiin ja sen eri vaiheisiin. Vaatimusmäärittelyssä käsitellään tietoturvaominaisuuksia eri näkökulmista. Viimeisimpänä teemana esitellään yhteistyö ja asiantuntija-apu. Aivan lopuksi tuloksista tehdään johtopäätökset.

6.1 Kehitysprosessi

Toimittajilla on hyvin erilaiset ohjelmistokehitysprosessit verrattuna asiakkaan vesiputousmalli -lähtöiseen hankintaprosessiin. Yritykset käyttivät ketterän kehityksen menetelmiä kehitystyön eri vaiheissa ja toivoivat asiakkaan osallistumista prosessiin.

Että kylmästi laitetaan ne vaatimukset ja tarkastetaan lopputulos, kun se on valmis, ni mun mielestä se ei…. mieluusti haluttas käyttää sitä iteroivampaa mallia ja että pystyttäs ymmärtään ne vaatimukset ja kehittään niitten pohjalta riittävää tietoturvaa

…Mikä taas näissä ketteris menetelmissä korostuu, tavallaan sen ketterän tiimin pitäis olla poikkitieteellinen tiimi. Siihen pitäis ottaa mukaan myös riittävä määrä tietoturvaosaamista tai tietoturvaihmisiä, jotta tavallaan ne määrittelyt pystyis myös elää ja niitä pystyttäs koko aika testaa sitä toteutusta varten…. Mää näkisin, et se ois hirveen tärkeetä.

Ketterässä kehityksessä puolustusvoimien työmäärä jakautuisi eri tavalla kuin nykyisessä hankintamallissa.

Sitten monesti kuulen väitteen, että se (ketterä kehitys) vaatii teiltä työaikaa ja enemmän tekemistä. Mä en sitä täysin osta, koska se vähentää teiän työtä sieltä alkupäästä, vaatimusmäärittelyvaihetta, sen ei tarvitse olla niin tiukkaan lukittu vaatimusten, vaikka

tulevalle järjestelmälle, koska sitä suunnitellaan koko aika ja se elää. Ja samoin se vähentää sitä vaihetta, kun teille tulee hyväksyntätestiin tai vastaanottotestiin järjestelmä ja sitä aletaan iteroimaan ja tulee näitä CR:iä ja tätä kaupallista muutoshallintaa. Se vähentää sitä työmäärää. Ja veikkaan, että ku käyttäjä ois koko aika mukana siellä, niin myös lopputulos olis laadukkaampi. Ja vastaa paremmin käyttäjien tarpeeseen.

Haastateltavat toivat esille toimintatapoja, joita iteratiiviset ohjelmistokehitysmenetelmät tukevat. Lisäksi he pohtivat tekijöitä, joilla asiakkaan tietoturvavaatimukset saataisiin toteutettua ohjelmistoissa siten, että ohjelmisto saisi hyväksynnän tietoturva-auditoinnista sekä pysyisi koko elinkaaren ajan tietoturvallisena.

6.1.1 Vaatimusmäärittelyn aloitus

Tietoturvan kehityksen tulisi alkaa vaatimusmäärittelystä. Siinä tulisi pohtia, mitä kehitettävältä ohjelmistolta tullaan tietoturvamielessä vaatimaan. Berenbach ym. (2009) tuovat esille ongelmatilanteita, kun vaatimuksia ei ole otettu projektin alussa huomioon. Tuotteen valmistumisen lähellä havaitaan, että vaatimukset ovat välttämättömiä testitapausten muodostamiseksi. Voidaan myös havaita, että vaatimuskatselmointi on välttämätöntä asiakkaan hyväksymiselle ja maksulle. Siinä vaiheessa vaatimusten tekeminen voi olla vaikeaa lähes valmiille jo järjestelmätesteissä olevalle järjestelmälle. (Berenbach ym., 2009)

Kun vaatimukset määritetään tuotteeseen puhtaasti sopimuksellisista syistä, valmis järjestelmä ei välttämättä vastaa asiakkaan tarpeisiin ja hän ei hyväksy sitä. Jos ei-toiminnallisten vaatimusten määrittäminen on viivästynyt projektin loppuun, järjestelmä ehkä osoittautuu riittämättömäksi tarkoitettuun käyttöön. Se ei välttämättä vastaa tarvittavaan suorituskykyyn, luotettavuuteen tai tietoturvatavoitteisiin (Berenbach ym., 2009, s. 41) . Kaikissa haastatteluissa korostettiin tietoturvavaatimusten määrittelyn aloitusta heti ohjelmistokehityksen alkuvaiheessa. Haastateltavien yhteinen ajatus oli, että tietoturva-asiat tulisi ottaa huomioon mahdollisimman aikaisessa vaiheessa ja tietoturvan toteutumista seurattaisiin myös asiakkaan puolelta koko ohjelmiston kehitystyön tai jopa ohjelmiston elinkaaren ajan.

Toimittajilla oli paljon esimerkkejä ei-toivotuista tilanteista, kun tietoturvavaatimukset oli asiakkaan puolelta tuotu esille liian myöhäisessä vaiheessa. Tietoturvaominaisuus oli esimerkiksi jätetty kokonaan toteuttamatta.

Tulihan siinä seki raja vastaan, että tähän nykyseen tuotteeseen ei riitä prosessointikapasiteettikaan toteuttaa ja muistit on täynnä ja prossuteho ei riitä siihen, että lähettäs toteuttaan jotain salausalgoritmeja. Että tämäki raja tuli tietysti vastaan, se huomattiin siinä. Siinäki mielessä ne vaatimukset pitäis olla ennen ku lähetään oikeastaan toteuttaan tuotetta ollenkaan. Että meillä on se arkkitehtuuri vielä auki ja myöski tää prosessointi-kapasiteetti ja kaikki, mitä se tietoturva tuoki mm lisää piirejä ja jottai muuta. Että nää pystytään huomioimaan oikeasti siinä designissa. Jälkikäteen niitten tuominen, siinä tulee nämä rajoitteet yhtenä. Tietenki se, että softassa ei oo ja toteutuksessa mitenkää huomioitu, että voi tulla kapasiteettiraja vastaan, niinku tässä esimerkissä on käyny, jos lähtökohta on ollu alun perin, että ei oo mitään suojaustaso-vaatimusta.

Tietoturvasta oli tullut puutteellista, koska sitä oli jouduttu ikään kuin liimaamaan päälle. Toteutus ei ollut vastannut tarkoitusta ja auditoinnissa oli tullut ei-toivottuja yllätyksiä.

Monessa ohjelmistokehitystyössä tietoturvavaatimukset olivat tarkentuneet vasta auditoinnin jälkeen. Auditoinnissa tarkastus oli tehty halutun suojaustason vaatimuksia vasten. Havaitut tietoturvapuutteet toimittaja oli saanut puutelistana, jotka olivat sitten lopulliset tietoturvavaatimukset. Kun suojaustasoa ei oltu määritetty selkeästi ennen kehitystyön aloittamista, haluttuun suojaustasoon vaaditut ratkaisut olivat osoittautuneet mahdottomiksi toteuttaa. Laajassa järjestelmänkehityksessä osa toiminnallisuuksista oli jouduttu jättämään kokonaan pois.

Noissa esimerkeissä on tiettyjä, jos on ollu laaja järjestelmä, paljon toiminnallisuuksii, paljon prosesseja, tavallaan jotain haaroja lopetettu kesken ja lopputuloksen laajuus ei oo ollu ihan se, mikä on alun perin tavoitteeks asetettu. Johtuen näistä tietoturvaseikoista.

Haastateltavien mukaan myös järjestelmien integrointiin liittyvät tietoturvavaatimukset tulisi ottaa esille jo sopimusten kirjoitusvaiheessa, jotta integrointi ei aiheuttaisi yllätyksiä.

Tutkimusten mukaan turvallisuuteen liittyvien vikojen korjaaminen on paljon kalliimpaa, kun se tehdään järjestelmäkehityksen myöhemmässä vaiheessa, eikä kehityksen alkuvaiheessa (Mokos, Meditskos, Katsaros, Bassiliades, & Vasiliades, 2010). Haastateltavien mukaan tietoturvavaatimukset toivat lisähintaa ja lisäsivät työmäärää tuotteeseen. Asiantuntijoiden mielestä hinta olisi kuitenkin halvempi, mikäli tietoturvavaatimukset olisivat kehitystyön alussa tiedossa, kuin jos ne tulevat vasta kehitystyön aikana esille. Mitä syvemmälle tietoturvavaatimus vaikuttaa ohjelmistossa, sitä tärkeämpänä pidettiin, että vaatimus olisi kehitystyön alussa tiedossa.

Kun vaatimusmäärittely edeltää järjestelmäsuunnittelua ja turvallisuusasiat ovat mukana vaatimusdokumentissa, ne voidaan ottaa huomioon sekä käyttää lähtökohtana järjestelmäarkkitehtuurin suunnittelussa (Schmidt, 2013). Myös toimittajien mukaan tietoturvavaatimukset pystyttäisiin toteuttamaan, mikäli ne olisivat tiedossa heti kehitysprosessin alussa. Lisäksi osa haastateltavista koki, että ne voitaisiin hinnoitella samoin kuin muidenkin vaatimusten toteuttaminen.

Yksi toimittaja oli sopinut kehitysprojektin alkuun erilliseksi kokonaisuudeksi tietoturvaominaisuuksien selvitystyön. Työ oli tilattu erikseen, koska se oli koettu niin isoksi ja se vaati paljon keskustelua asiakkaan tietoturvaihmisten kanssa. Työn tuloksena oli saatu vahvistettua vaatimukset ohjelmiston tietoturvaominaisuuksiin.

Ehdotettiin, että vaatimuksia määritettäisiin iteratiivisesti sekä yhteistyössä asiakkaan ja toimittajan kanssa. Nykyisten ohjelmistokehitysmallien mukaan pitäisi katsoa karkea ylätason visio siitä mitä tavoiteltaisiin. Lisäksi pitäisi katsoa tarvittavat toiminnot, ja kuvata ne vaatimuksina. Vesiputousmallin mukaiseen vaatimusmäärittelyyn liittyviä kysymyspattereita on tietoturvan osalta vaikea määrittää etukäteen, joten määrittelyä pitäisi tehdä iteratiivisesti aloittaen karkean tason vaatimuksilla ja edeten tarkempaan määrittelyyn.

Toivottiin, että asiakas ei tekisi liian yksityiskohtaisia vaatimuksia. Ohjelmiston kehitystä pidettiin niin monisäikeisenä ja haasteellisena työnä, että kaikkia ratkaisuja ei pystyisi selvittämään tai määrittämään etukäteen. Alkuvaiheessa kaikkia vaatimuksia ei voisi edes

kirjoittaa, koska ei ole välttämättä tehty esimerkiksi teknologiavalintoja, jotka saattaisivat vaikuttaa suojausten rakentamiseen.

Mikäli pidettäisiin kiinni siitä, että kaikki määritykset tehtäisiin tuotekehityksen alussa, määritystyöhön menisi niin paljon aikaa, että tuotetta ei enää tarvittaisi.

6.1.2 Asiakkaan osallistuminen

Asiakkaan osallistumista ohjelmistokehityksen eri vaiheisiin pidettiin hyvin tärkeänä. Osallistumista toivottiin ketterän kehityksen menetelmien mukaisesti vaatimusten analysoinnista aina valmiin tuotteen testaukseen asti.

Koettiin, että järjestelmätoimittajan voi olla vaikea ymmärtää vaatimuksia ja hahmottaa, miten ne pitäisi tehdä järjestelmään. Asiakkaan kanssa pitäisi neuvotella sopimusneuvotteluvaiheessa, mitä vaatimukset tarkoittavat. Koska toimittaja joutuu pilkkomaan asiakkaan vaatimukset teknisiksi vaatimuksiksi tai Scrum -maailmassa tehtäviksi, pidettiin hyvänä, että asiakas tai auditoija katsoisi toimittajan näkemyksen vaatimuksista. Asiakas pystyisi arvioimaan, onko järjestelmän toimittaja ymmärtänyt vaatimukset oikein ja vaikuttavatko toteutusratkaisut hyviltä.

Ehdotettiin, että käytäisiin vaatimukset läpi yhdessä ja purettaisiin ne auki ohjelmistoon tarvittavien toimintojen kautta. Yhdessä mietittäisiin esimerkiksi lokitus -toimintoa; mitkä vaatimukset kuuluvat siihen toimintoon. Jos asiakkaan toimintaympäristö ei tue kaikkia vaatimuksista toteutettavia ratkaisuja, sellaiset vaatimukset pitäisi pystyä poistamaan vaatimuksista. Ihmeteltiin, miksi pitäisi tehdä ratkaisuja, joita ei pysty hyödyntämään, kun niillä on kuitenkin eurovaikutus kaikilla. Toivottiin, että tehtäisiin vain ne toiminnot, joita todellisuudessa tarvitaan sekä jätettäisiin pois ne vaatimukset, joilla ei ole vaikutuksia operatiiviseen toimintakykyyn.

Ennen tilauksen tai sopimuksen tekemistä olisi tärkeää yhdessä purkaa vaatimukset käyttötapauksiksi. Tällöin jo alussa saataisiin molemmin puolin yhteisymmärrys, missä selviäisi asiakkaan toiveet. Toimittajalle selviäisi tällöin, että kykeneekö hän tekemään oikeanlaisen tuotteen. Tietoturva-arviointeja ja katselmointeja on pyritty saamaan jo

suunnitteluvaiheessa, jossa katsotaan toteuttamiskelpoisuus tietoturvan osalta. Katselmointeja on voitu myös pitää voitetun tarjouskilpailun jälkeen ennen suunnitteluvaihetta, mikäli projektissa on ollut tarvetta tarkentaa vaatimuksia.

Yhdessä yrityksessä tietoturvavaatimusten tarkennus oli osana tuotekehitysprosessia. Asiakkaalta pyydettiin vahvistukset esitettyihin ratkaisuihin, koska haluttiin varmistaa, että ratkaisut olisivat riittävällä tasolla. Ennen kaikkea oli havaittu, että epäselvät tai tulkinnanvaraiset ehdottomat tietoturvavaatimuksen oli parempi selvittää edeltä käsin. Todettiin, että tällainen menettely ei ole ollut toimittajalle edullista kiinteähintaisissa sopimuksissa, mutta toisaalta sitä ei voi kiertääkään, vaan työ oli ollut pakko tehdä.

Toimittaja esitti, että tietoturvavaatimukset käytäisiin tarkemmin läpi työpajoissa, joita pidettäisiin sopimuksen jälkeen noin kuukauden verran. Siinä toimittajalle ja asiakkaalle tarkentuisi ohjelmiston tietoturvaan liittyvät ominaisuudet.

Myös näissä (tietoturva) määrityksissä voisi olla tällaisia työpajoja. Olisi alussa tiedostettu, että nämä käydään toimittajan kanssa läpi ja sovitaan, miten olisi optimaalisin tehdä tietoturva. Tai vaatimuksen tulisi olla niin yleisellä tasolla, että sen voisi joustavasti ottaa käyttöön siihen omaan ympäristöönsä sen vaatimuksen. Olisi mahdollisuus, että käytäis yhteistyössä työpajana tällaiset vaatimukset, että täyttää asiakkaan antaman idean. Hankintavaiheessa sitä ei välttämättä osata kuvata, kun sitä järjestelmää ollaan tilaamassa/ostamassa… Voisi olla hankintaprosessin osana. Että kaikilla on alussa selvänä, että nämä on vaatimukset, jotka tiedetään, mutta sen jälkeen meillä on kuukauden aika, jossa nää vaatimukset sitten katotaan, mitä se tarkottaa.

Vaatimukset kehittyvät työn edetessä iteratiivisessa ohjelmistokehityksessä. Asiakkaan pitäisi siten olla mukana arvioimassa tietoturvan riittävyyttä kehitystyön aikana. Asiakas haluttaisiin pitää mukana, koska tietoturvaan liittyviin rajausten ja ohjauksen tekemiseen tarvitaan asiakkaalta ohjausta missä tahansa projektin vaiheessa. Mikäli tietoturva tehdään todella laadukkaasti, se myös maksaa paljon. Hintaan vaikuttavasta määrityksestä annettiin esimerkkinä loppukäyttäjän konfiguraatiomahdollisuudet. Mitä laajemmat mahdollisuudet käyttäjille annetaan, sitä enemmän tietoturva-asioita pitäisi ottaa huomioon.

Käyttäjien osallistuminen kehitysprosessiin katsottiin myös tarpeelliseksi. Kun käyttäjät olivat olleet mukana aina muutaman viikon välein, he olivat nähneet esittelyä kehittyvästä ohjelmistosta ja sitä kautta oli pystytty vastaamaan paremmin käyttäjän tarpeisiin. Tällä oli pyritty pääsemään myös parempaan laatuun.

Haastateltavat toivat esille, että tietoturva on käytettävyyden kanssa käsi kädessä, sekä käytettävyys liittyy olennaisesti tuotteeseen ja toteutettaviin ratkaisuihin. Tämän vuoksi olisikin hyvä yhdessä sopia tietoturvan ja käytettävyyden kannalta optimaaliset ratkaisut. Yhdessä projektissa kehitystyö oli tehty tiiviisti asiakkaan kanssa ilman kattavaa vaatimusmäärittelyä. Ohjelmistokehitystä tehnyt asiantuntija vaikutti tyytyväiseltä työtapaan.

Joo, että sen (iteratiivinen kehitys) oon kokenu erittäin hyvänä. Ei oikeestaan paremmin ois voinu mennä. Et sitä harjotellaan ensin, missä oon ollu mukana, tämmönen yhteistyöhanke puolustusvoimien kanssa. Sitä on tietosesti yhdessä kehitetty ja yhdessä määritelty ja keksitty lisää ominaisuuksia ja tälleen iteratiivisesti tehty eteenpäin.

Kaikkien asioitten ottaminen huomioon alkuvaiheessa on varmasti mahotonta tai se vaatii äärimmäistä asiantuntevuutta, että pystyy niin tarkasti määrittelemään kaiken. Sitä mukaa, ku asioita tulee vastaan tuotekehityksen aikana, on hyvä, jos niihin pystyy puuttumaan siinä. Tämmösiä kokemuksia mulla on ollu. Ehkä tuo hanke, missä ite oon ollu mukana, on ollu hieman erilainen. Puolustusvoimat on ollu siinä mukana tavallaan kehityskumppanina. Se ei oo sillä tavalla valmiin laitteen hankkimista.

Lisäksi hän toivoi tietoturva-asiantuntijan osallistumista ohjelmistokehitykseen koko tuotekehitysprosessin ajan. Asiantuntijan tulisi tietää, millä tavalla auditointi tehdään ja mitä järjestelmältä siihen vaaditaan. Suunnittelija oli saanut käsityksen, että auditoinnit tehdään vähän tapauskohtaisesti, sekä vaatimukset auditoinnille voivat muuttua kehitettävän ohjelmiston mukaan. Tietoturva-asiantuntijan toivottiin siis osallistuvan pitkäjänteisesti kehitystyöhön. Koettiin, että muutaman päivän mukana olemisesta ei ole juurikaan apua tietoturvan kehittämiseen. Palaute voi jäädä liian vajaaksi, kun hän ei tiedä missä mennään ja missä tietoturva-asioita käydään läpi.

Mikäli tietoturvaa tarkasteltaisiin koko kehitystyön ajan, ero tavoiteltavan ja toteutuneen tietoturvan välillä ei kasvaisi liian suureksi. Työ voitaisiin ohjata takaisin kohti tavoitetilaa, kun olisi enemmän tarkastuspisteitä. Toisaalta koettiin myös, että projektin edistymistä käydään jo nykyään säännöllisesti läpi. Katsotaan yhdessä, miten järjestelmä rakentuu ja missä vaiheessa ollaan menossa. Tällaista edistymistä voitaisiin seurata säännöllisesti myös tietoturvan osalta.

6.1.3 Auditoinnit

Tietoturva-auditoinnit oli pääsääntöisesti teetetty ulkopuolisilla asiantuntijoilla. Tavallisesti auditoinneissa oli tarkastettu ohjelmistokokonaisuuden tietoturvaominaisuudet. Auditoinnit koettiin hyödyllisiksi, koska yritykset olivat saaneet niistä tietoa tietoturvan kehittämiseen.

Kun auditointeja oli tehty ohjelmistokehityksen aikaisemmassa vaiheessa, niissä oli puolin ja toisin opittu tietoturvan kehittämisestä. Oli saatu ikään kuin välitulos, jonka perusteella pystyttiin arvioimaan korjausta vaativat asiat. Auditoinnissa oli myös annettu suuntaviivat tietoturvariskejä aiheuttavien erityisten ongelmien korjaamiseen.

Parhaimmillaan auditointia pidettiin kehittävänä tilaisuutena, jossa auditoijalta saatiin siis vinkkejä tietoturva-asioiden hoitamiseen. Hyvien käytänteiden vahvistamista pidettiin auditoinnin yhtenä tarkoituksena. Toimittaja oli saanut auditoinneissa myös lisätietoa alihankkimiensa tuotteiden haavoittuvuuksista.

Auditoinnit koettiin hyödyllisiksi, koska ohjelmiston tietoturvapuutteet olivat tulleet esille vasta auditoinnissa. Sitä ennen oli ollut hyvin vaikea saada riittävän täsmällisiä vastauksia tietoturvaan liittyviin teknisiin kysymyksiin. Auditoinnit koettiin ainoiksi paikoiksi, joissa oli saatu tietoa suojaustasojen tietoturvakriteereistä. Niissä oli selvitetty tarkasti, mitä asioita ohjelmistokehityksessä täytyisi ottaa huomioon, jotta ohjelmisto täyttäisi tietyn suojaustason kriteerit.

Tietoturva-auditoinnit koettiin myös valaiseviksi tilaisuuksiksi. Siellä oli viimeistään kirkastunut, mitä oli unohtunut. Toisaalta niistä oli saatu hyvää oppia, koska tietoturvaan

oli paneuduttu tilaisuuksissa niin perusteellisesti. Lisäksi oli herännyt koulutustarpeita, jotta tietoturvaratkaisut opittaisiin tekemään oikein. Auditointi oli koettu hyödylliseksi, koska tulokset oli käyty läpi ja vikojen kriittisyys sekä korjaustarpeet oli arvioitu. Auditointiraportti oli käyty läpi asiakkaan kanssa yhdessä, sekä korjaukset oli myös suunniteltu ja aikataulutettu.

Vaikka auditoinnit koettiin tärkeiksi tietoturvan kehittämisen kannalta, haastateltavat antoivat myös kehitysehdotuksia niihin. Koettiin, että arvioinnit pitäisi tehdä jo siinä vaiheessa, kun kehitysprojektille asetetaan tavoitteet. Niiden tekemistä pitäisi vielä jatkaa kehitysprojektin kaikissa vaiheissa. Tietoturvasuunnitelmat pitäisi myös arvioida, ennen kuin niitä lähdetään toteuttamaan.

Jos auditointeja tehtäisiin pitemmällä aikavälillä, voitaisiin keskustella ja miettiä, miten järjestelmää olisi hyvä kehittää oikeaan suuntaan, eikä pelkästään etsiä virheitä ja haavoittuvuuksia. Auditoinneissa olisi syytä miettiä järjestelmäkokonaisuutta, eikä mennä pelkästään tosi syvälle järjestelmään. Osa haastateltavista koki, että tuotetta oli ”hakkeroitu” ilman mitään rajaa.

Auditointeja oli tehty tuotteen eri kehitysvaiheessa. Olisi kuitenkin hyvä, että auditoinnit etenisivät johdonmukaisesti pienistä yksityiskohdista alkaen ja viimeisessä auditoinnissa katsottaisiin koko järjestelmäkokonaisuutta. Auditointi oli aiheuttanut negatiivisia yllätyksiä, koska se oli mennyt huomattavasti syvemmälle kuin Katakrista johdetut vaatimukset. Kokemus olikin, että pelkästään Katakria seuraamalla auditoinnista ei voisi päästä läpi.

Haastateltavat suhtautuivat auditoinneissa tulleisiin yllätyksiin hyvin eri tavoin. Osan mielestä yllätyksiä tulisi joka tapauksessa jonkin verran, kun taas osa oli sitä mieltä, että yllätyksiä ei saisi juurikaan tulla. Kaikilla pitäisi olla yhteinen kuva, mitä on rakennettu. Viimeisessä auditoinnissa katsottaisiin sitten läpi, ettei ole jäänyt mitään unohduksia tai pieniä puutteita. Koettiin, että olisi tosi ikävää, jos järjestelmän käyttöönotossa tulisi viime hetkellä etenemistä estäviä ongelmia ja jouduttaisiin palaamaan taaksepäin.

Auditoijan näkökulmasta yllätysten määrä vähenisi viimeisessä vaiheessa, jos tietoturvasuunnittelu ja tarkastukset olisivat alusta lähtien mukana. Tietoturvavaatimukset

suunniteltaisiin alusta lähtien, ja auditoija pääsisi jo toteutuksen aikana mahdollisesti katsomaan tietoturvan rakentamista ohjelmistoon.

Osa toimijoista oli jo tottunut tietoturva-auditoinneissa esiin tulleisiin yllätyksiin. Kuitenkin yllätysten määrä oli vähentynyt, kun auditointeja oli käyty läpi useita kertoja ja tietoturva-asioihin oli opittu paneutumaan. Yksi toimittaja kertoi, että auditointia oli helpottanut yrityksen tekemä tietoturvadokumentti. Siinä oli avattu ideologiaa suojaustapojen valintoihin ja kerrottu syitä toisten ratkaisujen poisjättämisestä.

Vaikka osa haastateltavista oli käynyt auditointiraportteja asiakkaan kanssa läpi, kuitenkin suurempi osa koki auditointiprosessin jääneen vähän kesken. Auditointeja ja raportteja oli tehty paljon, mutta tuloksia ei oltu arvioitu toimittajan kanssa. Myöskään jatkotoimenpiteitä tai tarvittavia korjauksia ei oltu suunniteltu. Asiakkaan tulisi antaa mahdollisimman kattava tieto auditointiraportista ja -tuloksista toimittajalle, jotta tarvittavat korjaukset ja uudet vaatimukset pystyttäisiin toteuttamaan. Varsinkin kehitysvaiheessa olevan tuotteen tietoturvaominaisuuksiin pystyttäisiin vielä vaikuttamaan.

Lisäksi toimittaja voisi käyttää auditointitulosten tietoa hyväksi järjestelmän jatkosuunnittelussa ja -kehityksessä. Haastateltavien mukaan kaikki auditointiraportin havainnot eivät välttämättä vaatisi korjauksia ohjelmistoon. Osa tuloksista voisi liittyä auditoidun ohjelmiston ulkopuoliseen osioon tai vika voisi ilmetä tietyssä käyttötapauksessa, jolloin esimerkiksi käyttötapaa muuttamalla vika ei tulisi ikinä vastaan.

Kaiken kaikkiaan auditoinnit todettiin kuuluvan tavallisiin tuotekehitysrutiineihin. Poikkeuksena ulkopuolisiin auditointeihin yksi yritys teki auditoinnit itse. Yritys oli osoittanut asiakkaalle toteutetun auditoinnin ja antanut auditoinnista tulokset. Auditoinnit olivat perustuneet sopimusvaiheessa tehtyyn sopimukseen ohjelmiston verifioinnista ja validoinnista. Asiakas oli ohjelmiston toimituksen jälkeen voinut auditoida tuotteen, mutta toimittaja ei ollut ollut siinä mitenkään mukana.

Tuotteisiin tehtyjen auditointien lisäksi haastattelussa tuli myös esimerkki projektihenkilöstön auditoinnista. Kehitysprojektissa auditoinnit olivat edenneet prosessina ja henkilöille oli pidetty kyselyhaastattelu ennen suunnittelutyön aloitusta. Haastattelussa oli muun muassa kysytty ohjelmistokehitykseen liittyvistä tietoturvavaatimuksista.

Tarkoituksena oli, että alussa olevissa auditoinneissa selviteltäisiin, mihin kehitystuotteeseen tehtävät auditoinnit peilattaisiin ja mitkä osa-alueet auditoinneissa tultaisiin koeponnistamaan. Tietoturvavaatimuksista oli tullut kommentteja myös asiakkaan suuntaan, koska ne oli koettu liian ympäripyöreiksi sekä niiden yksiselitteinen todennus oli koettu mahdottomaksi.

Auditoijan mukaan toimittajan kanssa tulisi käydä läpi, mihin auditoinnit pohjautuvat. Auditoinneissa käytettävät valtion hallinnon ohjeet tulisi esitellä, jottei toimittajilla olisi käsitys, että auditoinnit perustuvat johonkin auditointiyritysten näkemyksiin.

Asiantuntijan mukaan tarkastuksia voitaisiin tehdä kahdella tapaa asiakkaan tarpeiden mukaan. Tarkastuksessa lähtökohtainen kysymys voisi olla esimerkiksi: Onko järjestelmä turvallinen? Silloin järjestelmää tarkasteltaisiin monelta eri puolelta ja tietoturvavaatimukset olisivat yksi lähtökulma testitapauksille. Ohjelmistoa voitaisiin katsoa silloin myös turvallisuuden ja laadun näkökulmasta. Pitäisi tarkistaa, onko järjestelmä toteutettu hyvin ja löytyykö haavoittuvuuksia tai vikoja, joita voi esimerkiksi hyväksikäyttää.

Toinen tarkastus olisi vaatimustenmukaisuuksien todentaminen eli auditointi. Tällöin lähtökohtana olisi esimerkiksi selvittää, täyttyvätkö tietyn suojaustason vaatimukset. Katakri ja VAHTI-ohjeet antavat selkeän listan läpikäytävistä sekä tarkastettavista asioista. Ne antavat myös yleisohjeistukset, kuinka järjestelmiä tulisi tietoturvallisesti toteuttaa.

Auditoijan mukaan vaatimusmäärittely toimisi hyvin tarkastuksen lähtökohtana, mikäli vaatimusmäärittelyyn olisi kirjattu tietoturvavaatimuksia heti kehitystyön alusta lähtien. Vaatimusmäärittelyä sitten tuettaisiin sellaisilla asioilla, joita ei ole erikseen kirjattu. Hyvin usein tietoturvavaatimukset olivat kuitenkin puuttuneet tai niitä ei oltu dokumentoitu. Tällöin auditoija oli joutunut tarkastamaan tuotetta ja arvioimaan tietoturvan toteutumista yleisiä vaatimuksia vasten. Auditoija ei voi tietää kohdeympäristölle asetettuja erityisiä vaatimuksia, mikäli niitä ei ollut kirjoitettu mihinkään. Auditoija tekee työn sen tiedon mukaan, mikä hänelle on annettu. Mitä enemmän hänellä on tietoa käytettävissä, sitä paremmin hän pystyy työn tekemään.

Auditoija kertoi muutamia käytännön asioita, joilla auditoijan ja toimittajankin työtä voitaisiin helpottaa auditoinneissa. Tilaisuuden alussa oikeiden henkilöiden tulisi olla paikalla, sekä käytännön järjestelyt, esimerkiksi käyttöoikeuksien luominen, kannattaisi tehdä etukäteen. Näin auditoija pääsisi aloittamaan työnsä mahdollisimman nopeasti. Tarkastuksissa verrattaisiin dokumentaatiota ja toteutusta, mikäli dokumentaatio olisi ajan tasalla. Dokumentaatiosta voitaisiin katsoa lähtötilanne sekä siitä myös näkisi, mikäli tietoturvan kehityksestä oli tehty linjauksia ohjelmistokehityksen aikana. Auditoijan työtä myös selkeyttäisi, jos dokumentaatiosta näkyisi esimerkiksi suojaukseen kuuluvat, mutta toteuttamattomat tietoturvaratkaisut. Lisäksi työtä helpottaisi, jos auditoija saisi tietoturvavaatimusten lisäksi myös järjestelmän toiminnalliset vaatimukset. Niistä hän näkisi, miten järjestelmää pitäisi pystyä käyttämään ja pystyisi muodostamaan käyttäjätarinoita esimerkiksi kirjautumisten osalta. Tämän pohjalta voisi miettiä toimintaketjun hyväksikäyttömahdollisuuksia ja tehdä niistä testitapauksia auditointiin.

6.1.4 Tietoturvan todentaminen ja testaus

Haastateltavat toivat esille, että tietoturvavaatimuksista tulee helposti sellaisia, että niitä on vaikea todentaa. Jos vaatimus on hyvin yleisluontoinen, todentaminen voi olla mahdotonta. Lisäksi tietoturvan todentaminen vaatisi erityistä osaamista, johon asiakkaalle ei ole ollut riittävästi tietotaitoa.

Projektinhallinnan näkökulmasta vaatimusten todentamisen aikataulu pitäisi sopia. Tulisi sopia, todennetaanko vaatimus suunnittelu-, toteutus- vai vasta esimerkiksi vastaanottovaiheessa. Mitä aikaisemmassa vaiheessa pystyttäisiin keskustelemaan todentamisesta, sitä paremmalla pohjalla toimittaja olisi hankkeeseen lähtiessään. Todentamisessa tulisi myös miettiä, riittääkö vaatimukseen suunnittelumateriaalin paperitarkastelu, vai voidaanko vaatimuksen täyttyminen näyttää toteen vasta käyttöönottoympäristössä.

Eräässä yrityksessä oli toimintatapana, että he miettivät vaatimusmäärittelyvaiheessa, miten he osoittivat vaatimusten mukaisuuden. Jos heidän ehdotuksensa kelpasi asiakkaalle, näin toimittiin. Jos todentaminen ei kelvannut, sitä sitten muutettiin. Koettiin, että

varsinkin testitapauksista olisi hyvä keskustalla suunnittelun alussa. Olisi hyvä saada yhteinen näkemys tietoturvavaatimusten verifioinnista, jotta molemmin puolin tiedettäisiin, miten tuote täyttää tarvittavat tietoturvavaatimukset. Toimitusvaiheessa ei tarvitsisi sitten miettiä, onko tietoturvaominaisuus verifioitu oikein.

Myös yksi haastateltava oli tehnyt esityksen, miten vaatimuksia tultaisiin todentamaan. Toimittaja oli kokenut kovana haasteena tietoturvavaatimusten suuren määrän ja niiden yleisluonteisuuden todentamisen suhteen.

Jos sanotaan, että järjestelmä toimii koko ajan luotettavasti ja häiriöttä, siihen on vaikee yhtäkkii luoda jotain kriteeriä, että mites me nyt todistetaan tämä asia.

Ohjelmiston vastaanotto ja testaus olisivat jouhevampia, mikäli niitä tehtäisiin useammassa ohjelmistokehityksen vaiheessa. Joissain yhteyksissä näin oli jo toimittu. Loppukäyttäjiä ja muita asiakkaan edustajia oli pyydetty aina tietyllä periodilla tutustumaan järjestelmän käyttöön liittyviin asioihin tai ohjelmisto oli mahdollisuuksien mukaan asennettu asiakkaan ympäristöön.

Yllätyksiä ja hankaluuksia pystyttiin vähentämään esimerkiksi esittelemällä kehitettävää tuotetta sekä tuomalla asioita näkyväksi. Vaikka asioita pystyttiin edeltä käsin testaamaan, lopullisessa kohdeympäristössä oli esimerkiksi erilainen verkkoratkaisu, mikä vaikutti operaatioiden kestoihin ja viiveisiin.

Toimittaja pystyi pienentämään riskiä, mitä aikaisemmassa vaiheessa hän pääsi tekemään testausta kohdeympäristöön. Se ei ollut aina mahdollista, mutta siihen pyrittiin. Myös kokeilukäyttö asiakkaan laboratorioympäristössä oli todettu ohjelmiston testauksessa hyvin toimivaksi ratkaisuksi. Kokeilukäytön aikana tehdyt havainnot oli pystytty korjaamaan operatiiviseen käyttöön julkaistuun versioon. Haastattelussa tuli ehdotus yhteistyöstä testaustyökalujen käytössä. Yhteiskäyttöisillä työkaluilla voitaisiin säästää kustannuksissa, koska työkalut ovat kalliita ja niitä tarvitaan vain tietyissä vaiheissa projektin aikana.

Hyvänä käytänteenä pidettiin, että auditoinneista sovittaisiin projektin alussa projektiin kuuluvana testinä. Silloin se ei tulisi yllätyksenä projektin jälkeen ja estäisi pahimmassa tapauksessa ohjelmiston käyttöönottoa.

6.1.5 Muutokset tietoturvavaatimuksissa

Haastattelujen perusteella ei saanut aivan tarkkaa käsitystä, kuinka paljon tietoturvavaatimukset olivat muuttuneet ohjelmistokehityksen aikana. Tähän vaikutti ennen kaikkea se, että jopa viisi haastateltavaa totesi tietoturvavaatimusten tarkentuneen vasta loppuvaiheen auditoinneissa tai tietoturvaa ei oikein oltu käsitelty ohjelmiston kehitystyön aikana. Kuitenkin kolme haastateltavaa pohti, mitä tietoturvavaatimusten muuttaminen tarkoitti käytännössä.

Yhdessä tapauksessa vaatimuksiin oli tullut lisäys auditoinnin perusteella, kun oli selvinnyt, että tuote ei saisi sisältää ulkopuolisen tahon kirjoittamaa ohjelmistoa. Tämä tieto oli periaatteessa ollut jo ennen kehitystyön aloitusta, mutta sitä ei oltu kerrottu toimittajalle selkeästi. Toisessa esimerkissä asiakas oli tuonut uusia ideoita tietoturvan toteuttamiseen kehitystyön aikana ja toimittaja oli joutunut tekemään selvittelyn niiden sopivuudesta kehitteillä olevaan järjestelmään. Myös toimittaja oli pyytänyt muutosehdotuksen tietoturvavaatimuksiin, koska oli havainnut järkevämmän vaihtoehdon tietoturvavaatimuksiin ohjelmiston kehitysvaiheessa.

Tietoturvaan tulee jatkuvasti uusia vaatimuksia esimerkiksi Viestintäviraston julkaisemista ohjeista. Niitä puolustusvoimien järjestelmien on noudatettava. Yksi toimittajista oli joutunut selvittämään tuotteen toimituksen jälkeen, miten muuttuneet vaatimukset voisi toteuttaa tuotteeseen, jotta se menisi auditoinneista läpi. Nämä vaatimukset olivat menneet alkuperäisten vaatimusten ohi. Toimittajan mukaan tietoturvaan liittyviä muutoksia tuli noin kolme kertaa vuodessa. Ne eivät välttämättä olleet pelkästään muutoksia ohjeistuksissa, vaan esimerkiksi muutoksia toisissa järjestelmissä, joten niiden vaikutukset täytyi selvittää toimitettuun järjestelmään.

6.1.6 Tietoturva ja tuotteen elinkaari

Haastateltavien mukaan tietoturvan kehitys ei saisi loppua, kun ohjelmisto on todettu valmiiksi. Pitäisi olla suunnitelma, kuinka tietoturvaa voitaisiin kehittää vaikka seuraavissa MLU:issa. Toimittajat tekevät vuosia yhteistyötä laitevalmistajien kanssa, joten he voisivat antaa ennakkoon tietoa hyödyllisistä ominaisuuksista. Laitevalmistajat pystyisivät

selvittämään hyvissä ajoin, olisiko ominaisuus järkevää toteuttaa ja parantaisiko se tuotteen ominaisuuksia. Jatkokehitys koettiin tietoturvan osalta tärkeäksi ja myös kiinnostavaksi.

Haastateltavien mukaan jo tietoturvan määrittelyssä pitäisi miettiä, kuinka kauan ohjelmiston täytyisi olla tietoturvallinen. Tietoturvaan pitäisi panostaa jatkuvasti, jotta se pysyisi ajan tasalla. Esimerkiksi hyökkäysmenetelmät kehittyvät koko ajan, joten puolustuksen tulisi kehittyä myös samaa tahtia. Lisäksi ohjelmistojen tarvitsema muistin määrä kasvaa jatkuvasti, joten kymmenen vuoden päästä laitteisto voisi olla rajoite tietoturvan kehittämiselle. Esimerkiksi salausavaimien pituus kasvaa jatkuvasti, joten järjestelmän muisti ei välttämättä enää riittäisi hyväksyttävien avainten käyttöön.

Kun tietoturvaa ei oltu mietitty kokonaisvaltaisesti, yksittäiset tietoturvaratkaisut olivat tuoneet käyttävyysongelmien lisäksi myös ylläpidolle lisäresurssitarpeita. Järjestelmien tietoturvapäivityksiä pidettiin kuitenkin tärkeänä. Kun järjestelmä oli ollut pitkään käytössä, sen tietoturvavaatimukset olivat muuttuneet. Kuitenkin auditoinnit peilattiin nykyisiin vaatimuksiin.

Turvallisuuskäytänteiden kiristyminen oli aiheuttanut toimittajille ongelmia. Järjestelmien käyttöönoton myötä toimittajat eivät päässeet enää tilaan, jossa ohjelmistoa käytettiin. Jos ohjelmistoon tuli esimerkiksi tekninen ongelma, vian ilmenemisestä olisi saanut paljon enemmän tietoa sen käyttöympäristössä, kuin asiakkaan antaman ehkä ohuen kuvauksen perusteella.

CERT:n tuottamien raporttien seurannasta oli kahdenlaisia käytäntöjä. Tietoturvauhkia seurattiin yrityksissä itsenäisesti ilman asiakkaan kanssa tehtyä sopimusta, tai asiakkaan kanssa oli sovittu, että toimittaja seuraa ja informoi uhkista asiakasta. Kuitenkin selkeä vastuujako puuttui. Järjestelmän sisäisiä tietoturvapäivityksiä kyllä tehtiin, mutta ei oltu sovittu selkeästi, mikä oli asiakkaan ja mikä toimittajan vastuulla. Jos päivityksiä tulisi tehdä järjestelmäkokonaisuuteen, asiakkaalla ei ollut henkilöä, joka osaisi ottaa kantaa tarvittaviin korjauksiin. Yleisen uhkakuvan muodostamista ei projekteissa oltu tehty.

6.2 Vaatimusmäärittely

Asiakkaan määrittämät tietoturvavaatimukset koettiin pääsääntöisesti ympäripyöreiksi ja epäselviksi. Lisäksi vaatimusten tarkkuus oli vaihdellut. Toisaalta osa koki, että tietoturvavaatimukset olivat olleet hyvinkin ymmärrettäviä, mutta vääriä ja vaatimusten toteutettavuus oli mahdotonta. Vaatimusten toteuttamista oli helpottanut, kun tietty vaatimussetti oli tullut vastaan eri kehitysprojekteissa. Kuitenkaan tietoturvaominaisuuksien toteuttamista ei koettu erityisen vaikeaksi. Mikäli määritelmät olisivat selkeät, niiden kanssa voitaisiin toimia.

Abrams ym. (2004) mukaan laajojen järjestelmien hankinnoissa tietoturvan sisällyttäminen ja tietoturvavaatimusten luominen on tehty usein pinnallisesti. Vaatimukset ovat huonoja ja antavat heikon vaatimuskehyksen järjestelmän kehitykseen. Lisäksi vaatimuksiin liittyvissä päätöksenteon perusteissa on puutteita. Usein tietoturva-analyysien pöytäkirjoille ei ole keskitettyä säilytyspaikkaa, vaikka niitä olisi tehty läpi elinkaaren, projektin katselmuksesta ylläpitovaiheeseen asti (Abrams ym., 2004).

Tietoturvavaatimusten ymmärtämistä helpottaisi, jos toimittaja tietäisi, mitä vastaan suojausta pitäisi tehdä ja millä tavoin suojaus pitäisi toteuttaa. Eri suojaustasojen vaatimukset antavat ohjeita myös työskentelytiloista tai viestintäkanavista. Haastateltavien mukaan eri vaatimusten kanssa voitaisiin toimia, mikäli määritykset olisivat tarpeeksi selkeät.

Haastatteluissa tuli esille, että kotimaan viranomaisilla on hyvinkin yhtenevät tietoturvavaatimukset, mikä helpottaa niiden toteuttamista eri viranomaisten ohjelmistoihin. Myös teollisuudessa on käytössä tietoturvavaatimuksia, jotka antavat yleissivistävää oppia tietoturvan tekemisestä. Haastateltavien mukaan siellä on käytössä samat periaatteet ja parhaat käytännöt ohjelmien suunnittelusta ja huomioon otettavista asioista. Kuitenkin asiakkaan antamat tietoturvavaatimukset koetaan vaativammiksi kuin teollisuudessa käytössä olevat ratkaisut. Erityisesti tiukat suojaustasojen vaatimukset koettiin puolustusvoimien erityiseksi piirteeksi.

Haastatteluissa tuli myös esille, että puolustusvoimien sisällä tietoturvavaatimukset olivat olleet erilaisia eri organisaatioilla. Kaikille ei välttämättä kelvannut samanlaiset tietoturvaratkaisut. Tämä tietenkin lisäsi toimittajilla riskiä, kun ei voinut luottaa jo hyväksi havaittuun toimintaan tai ratkaisuun. Jollekin organisaatiolle oli jouduttu tuottamaan lisämateriaalia jonkin yleisessä vaatimusohjeistuksessa olevan suosituksen takia. Mikäli sama henkilö työskenteli kahdessa eri hankkeessa, tietoturvakäytännöt saattoivat olla erilaiset, vaikka lähtökohtaisesti ylätason vaatimuksia pidettiin täsmälleen samoina. Haastateltavan mukaan tämä johtui erilaisista organisaatiokulttuureista.

Puolustushaaroilla koettiin olevan eriävät tavat tehdä tietoturva-asioita. Tämä haastateltavan mukaan aiheuttaa riskin, että heikoimman organisaation toimintapa tiputtaa koko puolustusvoimien tietoturvatasoa. Toimintatavat saattoivat vaihdella organisaation osan, projektin, hankkeen tai järjestelmän mukaan. Yhdeksi syyksi tähän nähtiin yksiselitteisen ohjeistuksen puute. Osalla haastateltavista oli kuitenkin käsitys koko organisaation linjasta, koska he olivat havainneet, että yhdellä hankintaorganisaatiolla oli ollut yleisestä linjasta poikkeava tulkinta tietoturvaan liittyvissä asioissa. Keskusteluja oli jouduttu käymään, että vaaditaanko tässä enemmän kuin oli vaadittu jossain muualla.

Toisaalta osa haastateltavista koki, että samoja tietoturva-auditoinnin läpäisseitä komponentteja pystyi käyttämään turvallisesti muissakin ohjelmistoissa. Koska auditointi oli jo tehty, komponentti ei ollut aiheuttanut tietoturvaongelmaa toisessa järjestelmässä.

6.2.1 Perusteet tietoturvavaatimuksille

On hyödyllistä, että vaatimusmäärittelytiimissä kysytään ”miksi”-kysymyksiä. Kun tarve on tunnistettu kysymällä miksi, tarpeeseen löytyy perusteltu syy tai ehkä saadaan selville, että ominaisuus on jäänne. Jäänne on jotakin, jota on tehty jokaisessa projektissa, mutta sillä ei ole arvoa asiakkaalle ja kukaan ei tiedä, miksi se on mukana. Perusteiden miettimisellä voidaan poistaa tarpeettomat ominaisuudet, jotka muunnettaisiin vaatimuksiksi ja implementoitaisiin, vaikka asiakas ei niitä haluaisi (Berenbach ym., 2009, s. 41).

Toimittajat toivoivat selkeitä perusteluja tietoturvavaatimuksille. Heidän mukaan vaatimuksia oli vaikea ymmärtää, jos niitä ei pystytty perustelemaan. Toimittaja haluaisi esimerkiksi tietää, minkä suojaustason informaatiota ohjelmistossa liikkuu. Tämän perusteella hän voisi ymmärtää, miksi vaaditaan jotain tiettyä suojaustasoa ja mitä kaikkea tietty suojaustaso sitten vaatisi. Osa toimittajista uskoi, että asiakas vain kopioi hyvältä näyttäviä vaatimuksia vaatimuslistaan, eikä ollut miettinyt vaatimusten tarpeellisuutta tai välttämättömyyttä tuotteeseen. Toimittaja toivoi, että ikään kuin tällaiset ylimääräiset vaatimukset voitaisiin poistaa. Lisäksi kaikki varautuminenkin tulisi jättää pois. Turhina vaatimuksina pidettiin esimerkiksi viiden vuoden päästä ajankohtaiseksi tulevia vaatimuksia.

Toimittajat olivat käsitelleet tietoturvavaatimuksia eri rooleissa ohjelmistokehityksessä. Kun he pohtivat, mitä tietoturvavaatimusten määrittämisessä tulisi ottaa huomioon, he kaikki painottivat eri asioita. Koska tietoturva vaikuttaa niin moneen asiaan ohjelmistossa, sen huomioiminen koettiin mutkistavan ja teettävän enemmän töitä ohjelmiston kehityksessä. Ongelmaa oli lisännyt vielä tietoturvavaatimusten epäselvyys. Haastateltavat toivoivatkin, että epäselvyyksien vähentämiseksi tietoturvavaatimuksista tulisi sopia hankintasopimuksissa.

Tietoturvavaatimuksissa tulisi ottaa huomioon ohjelmiston kokonaisuus, johon vaikuttaisi ohjelmiston uhka-arvio, käyttöympäristö ja käyttäjät. Lisäksi tietoturvan toteuttamiseen vaikuttaa merkittävästi ohjelmistolta vaadittava suojaustaso. Vaikka tietoturvaominaisuudet olivat puolustusvoimien järjestelmissä korostuneesti esillä, pitäisi kuitenkin muistaa säilyttää ohjelmiston käytettävyys ja operatiivinen suorituskyky. Näiden huomioimista auttaisi, mikäli toimittaja pystyisi saamaan ohjelmiston käytöstä laajan ymmärryksen.

Ohjelmistokehittäjän näkökulmasta tietoturvan toteuttamisessa tulisi ottaa huomioon kehitettävän tuotteen elinkaari. Lisäksi pitäisi tietää hyvissä ajoin, miten tietoturvan toteutuminen todennetaan ja testataan. Yksi toimittaja pohti vaatimusten prioriteetteja. Kun tulee satoja kriittisiä vaatimuksia, olisi kehityksen kannalta hyvä tietää, mitkä olisivat ne kaikkein tärkeimmät vaatimukset myös tietoturvavaatimuksista. Monesti ohjelmistoa

tilataan osaksi jotain suurempaa ohjelmisto- tai järjestelmäkokonaisuutta. Silloin rajapinnat tulisi määrittää tarkasti myös tietoturvan osalta. Pitäisi selvittää, mitkä osa-alueet olivat tilattavan ohjelmiston vastuulla ja mitä tietoturvaratkaisuja järjestelmäkokonaisuus jo toteutti.

Koettiin, että tietoturvaan liittyy aina yllätyksiä. Niistä ei voisi päästä täysin eroon, koska ominaisuuksien tulisi elää ulkopuolisten, riippumattomien tahojen mukaan. Tietoturvaa voitaisiin miettiä myös laadun näkökulmasta. Kun laadullisesti pyrittäisiin tekemään asioita mahdollisimman hyvin, se näkyisi myös tietoturvan parantumisena. Hyvästä laadusta annettiin esimerkkinä, että ohjelmisto noudattaisi vähimpien oikeuksien periaatetta.

6.2.2 Tietoturva ja kokonaisuus

Haastateltavat korostivat tietoturvan kokonaisvaltaista kehitystä. Yhtenä kokonaisuuteen vaikuttavana tekijänä pidetiin ohjelmistojen kehittämistä eri aikaan eri järjestelmiin, jolloin ei voitu välttyä epäsynkronisilta tilanteilta. Toinen järjestelmä ei välttämättä tukenut jotain tietoturvaominaisuutta, joka oli uudemmalta ohjelmistolta vaadittu. Ohjelmistot eivät välttämättä siten pystyneet kommunikoimaan keskenään.

Toisaalta tietoturvassa ei saisi olla aukkoja missään kohdassa. Todettiin, että jonkun ominaisuuden loppuun asti hiominen ei auta, jos johonkin toiseen kohtaan jää aukko. Tietoturvan kokonaisvaltaisessa kehittämisessä olisi ensin tiedettävä arkkitehtuuri ja järjestelmään kuuluvat moduulit. Tämän jälkeen voitaisiin miettiä tietoturvan kehitys, eli sopivat tietoturvaratkaisut järjestelmään. Esimerkiksi pitäisi tietää, miten liikennöinti tapahtuisi eri moduulien välillä tai mitä informaatiota järjestelmässä tultaisiin käsittelemään. Siitä pystyisi määrittämään, mitä tulisi suojella ja mitä estää. Tämän perusteella voitaisiin kehittää tietoturvaratkaisut. Tietoturvaa mietittäisiin tavallaan paloissa siten, että ensin mietittäisiin isoa kuvaa ja sen jälkeen tarkemmin, mistä tämä kokonaisuus rakentuisi.

Jos ison järjestelmäkokonaisuuden yksittäiset komponentit on hankittu eri toimittajilta, tietoturvanäkökulmasta pitäisi pystyä päättämään, mitkä tietoturvaominaisuudet tulisi

vaatia miltäkin komponentilta. Pitäisi miettiä, tulisiko järjestelmäkomponentin tuottaa itsessään kaikki kokonaisuudelta vaadittavat tietoturvaominaisuudet vai tulisiko sen integroitua ympäristössä ehkä jo toteutettuihin ratkaisuihin. Lisäksi pitäisi suunnitella, miten vaatimusmäärittely kirjoitetaan yksittäiseen hankintaan sopivaksi.

Esimerkiksi järjestelmä itsessään toteuttaa osan ja osa niistä vaatimuksista saattaa toteutua sen ympärillä olevista järjestelmistä ja palveluista. Ne on ehkä hankalempia asioita siinä mielessä, et jos näkyvyys on ainoastaan yhteen järjestelmään ja sitten pitäisi kirjoittaa, että osaa niistä vaatimuksista järjestelmän ei tarvitse toteuttaa, vaan ne toteutetaan jollain toisella kommentilla, niin mitenkä tämä tehään. Ei se mahottomuus ole, mutta se vaatii vaan enemmän keskustelua puolin ja toisin, että mitenkä se toteutus tulis tehä.

Mikäli tietoturvaa ei oteta kokonaisuudessaan huomioon puolustusvoimissa, haastateltavien arvioiden mukaan järjestelmien integraatiot jäävät puutteellisiksi ja päällekkäisiltä tietoturvatoiminteilta ei voi välttyä. Tietoturvan kokonaisuuden kannalta tulisi lisäksi ottaa huomioon, että kaikissa ohjelmistoissa ei edes ole toiminnallisuuksia, mitä esimerkiksi Katakrissa vaaditaan.

No otetaan vaikka vahva tunnistus. Jos ajatellaan vaikka, että siinä pitää olla tämmönen, joku RSA-pohjanen tai toimikortti-tyyppinen ratkasu, niin, että liittyykö se johonki teiän olevaan systeemiin, että se sitä kautta hakee vahvan tunnistuksen vai rakennetaanko tähän tuotteeseen ihan oma. Koska semmosiin me ollaan törmätty, että ei löydy semmosta tahoo, joka ottas kantaa siihen, että liitetäänkö tämä teidän järjestelmään, vai onko tällä oma. Ja silloin me joudutaan päättään se, että koska me ei tiedetä, mihin tämä liitetään, ni tälle järjestelmälle tehdään sitte oma vahvatunnistus ja korttisysteemi. Välttämättähän teidän organisaation tietyt tahot ei siitä tykkää, ku tulee taas yks hallintajärjestelmä lisää, jossa pitää julkasta toimikortteja.

Vaatimuksissa tulisi siis ottaa huomioon tällaiset seikat, eikä vain tehdä vaatimuksia yleisellä tasolla. Kun hankitaan yhtä pientä osa-aluetta, ei pitäisi vaatia Katakrin vaatimusten täyttämistä, koska ne koskevat koko järjestelmäkokonaisuutta. Pitäisi kirjoittaa, miten hankittava osa-alue integroituu muuhun järjestelmään ja miten se pitäisi

toteuttaa, että haluttu tietoturvallisuus saavutettaisiin. Haastateltavat siis toivoivat, että asiakas avaisi ja johtaisi tietoturvavaatimuksia yleisistä ohjeista.

Hankitaan yhtä pientä palikkaa ja siinä kirjotetaan yleisellä tasolla Katakri STIV vaatimukset. Siellä (Katakrissa) saattaa olla niitä vaatimuksia myös, että sen pittää ottaa huomioon vaikkapa käyttäjien hallintaa tai muuta vastaavaa. Mutta itsessään se järjestelmä, pieni komponentti, toiminnallisuuksiltaan siellä ei oo mitään siihen liittyvää. Ni mitenkä ne toteuttaa, että ne vaatimukset toteutuu, mutta siellä ei sitä toimialaisuutta välttämättä ole ollenkaan. Vaan se on itse asiassa siellä isossa järjestelmäkokonaisuudessa se toiminnallisuus ja et sen pittää integroitua sinne.

Lisäksi haastateltavat toivoivat, että asiakkaalla olisi järjestelmäarkkitehti, joka tietäisi järjestelmäkokonaisuuden ja ymmärtäisi tietoturvasta. Tällöin hän pystyisi hallitsemaan tietoturvaa kokonaisvaltaisesti myös järjestelmien käytön aikana.

Näkisin, että olis hirveen helppoo, jos teillä olis systeemiarkkitehti, joka tietää nää liittyvät järjestelmät ja tietoturvan. Ni pystyy heti sanomaan tai ottamaan jopa toimittajaan yhteyttä, että hei, että nyt meiän on syytä vaihtaa vaikka tämä Windows johonki muuhun tai ottaa joku korjauspätsi käyttöön.

Haastateltavat kokivat tietoturva-asioiden huomioimisen hyvin monitahoiseksi työksi, sekä järjestelmäarkkitehdin tarpeellisuutta perusteltiinkin monesta eri näkökulmasta:

- Järjestelmäarkkitehti pystyisi heti sanomaan tai ottamaan toimittajaan yhteyttä, mikäli tarvittaisiin jotain tietoturvapäivityksiä, kuten vaikka käyttöjärjestelmän vaihtaminen tai korjauksien käyttöönotot
- Kun turvallisuusominaisuuksia lisätään järjestelmään, koordinointi tulisi toteuttaa kaikkien järjestelmätoimittajien kesken.
- Järjestelmäarkkitehdin tulisi olla mukana suunnittelemassa järjestelmäkokonaisuutta sekä kirjoittamassa tietoturvavaatimuksia koko järjestelmälle. Hänen pitäisi myös olla tekemässä määrittelyä, miten yksittäisiä järjestelmäkomponentteja voidaan toteuttaa järjestelmään tietoturvallisesti.

- Koko järjestelmän toimintakyky tulisi myös huomioida. Mikäli yksi komponentti lakkaisi toimimasta, se ei saisi vaarantaa koko järjestelmän tietoturvaa. Tällöin järjestelmäarkkitehdin tulisi arvioida, mitä yksittäisen komponentin hajoaminen tarkoittaisi koko järjestelmän osalta.

Osa haastateltavista mietti myös, että auditointi tulisi suunnitella järjestelmäkokonaisuudesta käsin. Pitäisi paneutua siihen, miten järjestelmät ovat liittyneet toisiinsa ja miten eri moduulit toimivat. Nyt auditoinneissa tutkitaan hyvin syvällisesti järjestelmän avainkomponentteja. Niistä tutkitaan hyvin tarkasti tietoturvan toteutuminen ja haavoittuvuudet. Haastateltavien mukaan kokonaisuuden hahmottaminen vie myös auditoijilta hyvin paljon aikaa. Järjestelmät eivät yleensä ole auditoijille tuttuja, eikä asiakas yleensä anna heille kuvaa järjestelmäkokonaisuudesta. Tällöin asiakkaan vastuulle jää miettiä, mitä esimerkiksi yksittäisen komponentin hyväksikäyttömahdollisuus tarkoittaa koko järjestelmän tietoturvan kannalta.

6.2.3 Uhka-arviot

Useissa lähteissä ohjelmiston suunnittelijat tekevät uhka-analyysia ja -mallinnusta ohjelmiston kehitystyön aikana (Dhillon, 2011; Torr, 2005; Yu ym., 2015). Artikkeleissa kehittäjät pyrkivät tunnistamaan kehittämänsä ohjelmiston kohdat, jotka voisivat joutua hyökkäyksen kohteeksi. Koska suunnittelija tuntee suunnittelemansa järjestelmä, hän pystyy arvioimaan, mitkä paikat vaativat eniten suojausta.

Toisin kuin lukuisissa artikkeleissa haastateltavat halusivat tietää ennen järjestelmän kehitystyön aloitusta ohjelmistoon kohdistuvat uhkat. Toimittajat kokivat, että tietoturvavaatimuksille ei ole löytynyt perusteita. Uhka-arviot auttaisivat hahmottamaan, mitä vastaan suojausta eli tietoturvan kehitystä pitäisi tehdä. He myös uskoivat, että uhka-arvion avulla voitaisiin löytää kokonaistaloudellisimmat tietoturvaratkaisut ohjelmistoon.

…mitä me tavotellaan tietoturvalla ja mikä on se uhka, mitä vastaan me halutaan suojautuu. Koska se, että tehdään tietoturvaa tietoturvan tekemisen ilosta, on vähän turhaa. Mut et meidän pitäis tietää, et mitkä on ne uhkat, eli ketä vastaan me halutaan

suojautua. Että halutaanko me suojautua ulkopuolista vai omaa työntekijää vai sekä että, ja millä tavoin halutaan.

Ainoastaan kahdelle toimittajalle asiakas oli jakanut tietoa uhkista. Osa haastateltavista toivoi, että asiakas tekisi itsenäisesti uhka-analyysia hankittavana olevaan järjestelmään. Useampi toimittaja kuitenkin näki, että uhka-analyysi tulisi tehdä yhteistyössä toimittajan ja asiakkaan kanssa.

Artikkelissa (Spears & Barki, 2010) käyttäjät olivat mukana tekemässä tietojärjestelmien tietoturvan riskianalyysia ja hallinnan suunnittelua, kuten riskien priorisointia, suunnittelua, toteutusta, testausta sekä tietoturvakontrollien monitorointia. Artikkelin mukaan käyttäjien osallistuminen kasvattaa organisaation tietoisuutta tietoturvariskeistä ja liiketoimintaprosessien kontrolleista. Puolustusvoimien organisaatiossa hankintoja tekevät henkilöt ja järjestelmien käyttäjät ovat eri toimijoita, mutta tässä yhteydessä molemmat voisivat olla rinnastettavissa käyttäjiin. Artikkelissa todetaan, että käyttäjät pystyvät antamaan liiketoimintatuntemusta tietoturvan hallinnan suunnitteluun ja siten voidaan saavuttaa tehokkaampia tietoturvatoimenpiteitä.

Toimittajat uskoivat, että uhkien tunnistaminen voisi auttaa löytämään tietoturvaan vaihtoehtoisia ja mahdollisesti uusia ratkaisuja. Voisi myös paremmin peilata maailmalla oleviin tietoturvatrendeihin ja mahdollisesti löytää uusia ratkaisuja suojautumiseen. Ratkaisut voisivat olla siten hyvinkin poikkeavia. Artikkelin tutkimuksessa oletetaan, että käyttäjät ovat suurin uhka järjestelmien tietoturvalle. Vaikka haastatteluissa asiakkaita ei nähty erityisenä uhkana, toimittajat kuitenkin selkeästi halusivat, että asiakas olisi mukana tekemässä uhka-analyysia ja uhkista aiheutuvien riskien hallintaa.

Koska uhka-analyysejä ei oltu tehty asiakkaan eikä toimittajan puolelta, käsitykset analyysin sisällöstä vaihtelivat aika paljon haastateltavien välillä. Joidenkin mukaan uhka-analyysissa mietittäisiin, mitä uhkia ympäristö tai käyttäjät aiheuttaisivat, kun taas osa mietti enemmän järjestelmän teknisiä uhkia.

6.2.4 Katakri ja VAHTI-ohjeet

Haastattelujen mukaan tietoturvavaatimusten määrittämisessä puolustusvoimien yleisin tapa oli viitata Katakriin. Kaikissa haastatteluissa tuli kuitenkin esille, että siihen viittaaminen oli liian ylimalkainen lähestymistapa määriteltäessä ohjelmiston tietoturvaa. Katakri ei anna yksiselitteisiä tietoturvavaatimuksia, ja joidenkin mielestä se oli täysin väärä lähestymistapa vaatimusmäärittelyssä. Haastateltavien mukaan Katakri sopii rakennettaessa tiloja sekä määriteltäessä käyttövaltuushallintaa tai tietojärjestelmän käyttöä määrättyihin tiloihin. Lisäksi sitä voidaan käyttää auditointityökaluna.

Mikäli Katakria käytettäisiin tietoturvavaatimusmäärittelyssä, haastateltavat toivoivat asiakkaan tulkitsevan tai tarkentavan muun muassa seuraavat asiat:

- Katakrin mukaan tietoturvasta voidaan poiketa tietyissä asioissa, jos organisaation turvallisuusviranomainen näin päättää. Asiakkaan tulisi ilmoittaa toimittajalle Katakrin mukainen turvallisuusviranomainen.
- Katakrissa on esillä riskien hallintamenetelmät tietoturvapoikkeamista aiheutuviin riskeihin. Toimittajan kanssa pitäisi sopia riskienhallintamenettely, kuka tekee mitäkin, mitä menetelmää käytetään, mitä riskejä siinä käydään läpi, miten ne pitää painottaa ja milloin voidaan ylipäätään poiketa tietoturvavaatimuksista.
- Katakri on valtionhallinnon tietoturvaohje normaaliolojen tilanteisiin. Asiakkaan tulisi kuitenkin varautua poikkeusolojen toimintaan, joten ohjeita haluttaisiin toimintakyvyn säilyttämiseksi myös poikkeusoloissa.
- Mikäli suunniteltavan ohjelmiston tietosisällöllä on eri suojaustasovaatimus kuin sen käyttöympäristöllä, asiakkaan tulisi määrittää, miten ohjelmiston suojaus silloin tulisi toteuttaa.
- Tietoturvavaatimukset pitäisi miettiä myös järjestelmän toimintaympäristön kannalta. Esimerkiksi tulisi selvittää, mitä siellä tehdään ja ketkä voivat päästä käsiksi järjestelmään.

- Suunniteltava tuote voi olla tarkoituksellisesti ominaisuuksiltaan hyvin rajallinen esimerkiksi muistin koon suhteen, eikä Katakrin yleisiä vaatimuksia voida silloin noudattaa. Asiakkaan tulisi antaa Katakrista poikkeavat tuotekohtaiset vaatimukset ja aukaista ne vaatimusmäärittelyssä.
- Asiakkaan ohjelmistoon voidaan tarvita myös Katakrista riippumattomia tietoturvavaatimuksia, joten niiden kuuluisi olla vaatimusmäärittelyssä mukana.
- Katakrista johdetut tietoturvavaatimukset ovat huomattavasti yleisemmällä tasolla, kuin auditoinneissa tehty tietoturvan testaus. Näin ollen Katakrin mukaan tehty tietoturvamäärittely ei takaa, että järjestelmä menisi auditoinnista läpi. Asiakkaan tulisi olla mukana tietoturvan kehitystyössä, jotta riittävä tietoturvataso saavutettaisiin.

Mikäli tietoturvavaatimuksia ei oltu tarkennettu vaatimusmäärittelyssä, toimittajat olivat itse tulkinneet Katakrista sopivat tietoturvavaatimukset ja muodostaneet tietoturvaspesifikaation. Asiakas oli kuitenkin antanut niukasti tietoa tulkintojen riittävyydestä. Katakria pidettiin hyvänä taustamateriaalina tietoturvan kehittämisessä. Jos vaatimusmäärittelyssä oli annettu tietoturvavaatimus sekä viitattu Katakriin tai VAHTI-ohjeisiin, niistä sai lisätietoa vaatimuksen kontekstiin. Näin sai ehkä paremman kuvan vaatimuksen tarkoituksesta.

VAHTI-ohjeiden sisältö ei ollut kovin tuttu kaikille haastateltaville. Asiakas ei ollut esitellyt niitä toimittajille, vaan niihin oli viitattu yleisesti tietoturvavaatimuksissa. Joissain tapauksissa asiakas oli pyytänyt poimimaan VAHTI-ohjeista vaatimuksiin soveltuvat asiat. Koska VAHTI-ohjeistukset ovat laaja kokoelma dokumentteja, vaatimukset pystyttiin tulkitsemaan hyvin monella tavalla. Yhden toimittaja mukaan VAHTI-ohjeet eivät lisäksi ota huomioon tuotteen todellista käyttöympäristöä, joten niitä olisi vaikea tulkita operatiiviseen tuotteeseen. Katakrin vaatimuksissa olevia VAHTI-ohjeiden viittauksia oli jonkin verran luettu. Kun viitaukset tehtiin yksittäisiin VAHTI-ohjeisiin, toimittajat kokivat VAHTI-ohjeiden käytön hyödyllisenä lisätietona.

Yksi toimittaja kertoi yrityksen toimintatavasta vahvaa suojausta vaativassa ohjelmistokehityksessä. He olivat käyneet läpi kaikki VAHTI-ohjeet ja poimineet ne

kohdat, jotka koskisivat heidän mielestään kehitettävää ohjelmistoa. Ohjeet oli koottu taulukkoon sekä viereen oli kuvattu heidän tulkinta, oletukset ja rajaukset kyseisestä ohjeesta. Kun taulukko oli saatu valmiiksi, he olivat keskustelleet tulkinnasta asiakkaan kanssa sekä pyytäneet varmistuksen, että tulkinta olisi riittävä tietoturva-auditoinnin kannalta. Koska asiakkaalta ei oltu saatu riittävän tarkkaa ohjeistusta, he olivat halunneet selvittää ennen kehitystyön aloitusta valitsemiensa teknisten ratkaisujen riittävyyden haluttuun suojaustasoon.

6.2.5 Käytettävyys, toiminnalliset vaatimukset ja tietoturva

Tutkimuksen mukaan vähintään puolet tietojärjestelmien tietoturvarikkomuksista tulee käyttäjien huolimattomasta tietoturvakäyttäytymisestä. Erityisen suuren osan rikkomuksista aiheuttaa järjestelmän käyttö ilman asianmukaista tunnistautumista (Gordon, Loeb, Lucyshyn, & Richardson, 2005).

Osa haastateltavista mietti käytettävyyttä ja tietoturvaa ennen kaikkea käytettävyyden näkökulmasta. Ensin pitäisi suunnitella käytettävyys, jonka jälkeen tietoturva suunniteltaisiin niin taitavasti, että käytettävyys pysyisi kohtuullisena.

Ne pitäis sillä tavalla, että ensin se käytettävyys, sit tietoturvavaatimus. Vaikka että loggaudutaan johonki järjestelmään sisään, ni se pitäis olla sillä tavalla, että se käyttäjä tekee sen kerran tai mahdollisesti automaattisesti, ku hän aloittaa työvuoronsa ja tällä tavalla. Se tulis niin ku mahollisimman sille loppukäyttäjälle, että se ei oo sillä tavalla, että hänen tarvii aina jostaki tietoteknisestä syystä tehdä ylimäärästä työtä tunnistautumiseen tai johki muuhun liittyen.

Toisen näkökulman mukaan käytettävyyden ja tietoturvan välillä täytyisi päättää kumpaa halutaan painottaa enemmän. Kuitenkaan ei pitäisi vaatia sellaisia tietoturvaominaisuuksia, että käytettävyys kuolee ja operatiivisesti vaadittu tekninen suorituskyky heikkenee. Äärimmäisyyksiin vietynä tietoturva aiheuttaisi järjestelmän käytössä niin paljon ongelmia, että tietoturvakin kärsisi.

Käytettävyyttä pohdittaessa mietittiin ennen kaikkea pääsynhallintaa ja käyttöoikeuksia. Haastattelujen mukaan moneen suuntaan linkittyneiden järjestelmien käytettävyyttä parantaisi esimerkiksi yhteinen sisäänkirjautuminen. Käyttöoikeuksien hallinnalla pystyttäisiin käyttäjille määrittämään järjestelmiin pääsy ja käytettävät resurssit. Paremmalla käytettävyydellä heidän ei tarvitsisi etsiä oikoreittejä järjestelmän käytössä tietoturvan pysyessä kuitenkin kohtuullisena.

…käytettävyyden kanssa käsikädessä. Esimerkiksi tunnistautuminen tulisi miettiä, miten se käytännössä tehdään. Eli kokonaisuus olisi mietitty. Ei ole puuli-korttia, joka on koko ajan siellä, vaan olisi sen verran joustava, että jokainen tekisi sen oikein, eikä tulisi tarvetta keksiä tapoja, miten tätä voidaan käyttää helpommin.

Käyttöoikeuksien hallinnassa pohdittiin luottamusta. Mikäli käyttäjiin ei ole mitään luottoa, käyttövaltuuksista tulee hyvin segmentoituneita. Tämä aiheuttaa hyvin monimutkaisia yhdistelmiä, joiden hallinta aiheuttaa tietoturvariskin jo sinällään. Sen sijaan, että pyrittäisiin minimoimaan oman työntekijän uhkaa sekä tekemään oma käyttöoikeus jokaiselle työntekijälle, käyttöoikeusmalli voisi olla yksinkertaisempi. Monimutkaisten tietoturvaratkaisujen sijaan valvontaa, monitorointia ja analytiikkaa pitäisi kehittää. Tällöin pystyttäisiin mahdollisimman nopeasti reagoimaan työntekijän toimiessa oman toimintatapansa ulkopuolella.

Venkatesh & Bala (2008) esittelevät artikkelissaan tutkimustuloksia käytettävyydestä. Artikkelin mukaan käyttäjien osallistuminen tietoteknisten järjestelmien kehitykseen parantaa käyttäjien asenteita kehitettävää järjestelmää kohtaan. Käyttäjät saavat paremman ymmärryksen järjestelmän ominaisuuksista, organisaation resursseista sekä antavat myös asiaankuuluvan tukensa järjestelmän käyttöön. Näin järjestelmään kohdistuvia ennakkoluuloja pystytään vähentämään sekä parantamaan käytettävyyttä tasapuolisesti. Myös myönteiset näkemykset ulkoisiin kontrolleihin lisääntyvät (Venkatesh & Bala, 2008).

Kaikki haastateltavat pitivät käytettävyyden kannalta tärkeänä loppukäyttäjien mukana oloa ohjelmiston kehitystyössä. Käyttäjien edustajat osallistuivat jo vaatimusmäärittelyyn, ja heille esiteltiin kehitettävää ohjelmistoa mahdollisimman aikaisessa vaiheessa.

Järjestelmän käyttöä oli ollut vaikea kuvata kirjallisesti, joten keskustelu ja havainnointi olivat auttaneet käyttötapausten ymmärtämisessä. Ohjelmiston testausta oli pyritty tekemään käyttäjien kanssa jo kehitysvaiheen aikana, jotta sovellus olisi heille tuttu hyväksyntätestejä varten. Vaikka käyttäjät olivat olleet mukana kehitystyössä antamassa kommentteja käytettävyydestä, tietoturvaan liittyviin asioihin käyttäjät harvemmin pystyivät kommentoimaan.

Yhden haastateltavan näkemyksen mukaan käytettävyys voi olla hyvinkin pienistä ominaisuuksista kiinni, joten teknisillä tietoturvaratkaisuilla voisi jopa helpottaa käyttäjän toimintaa. Ratkaisuja olisi helpompi miettiä, jos suunnittelija pystyisi perehtymään tarkemmin järjestelmän käyttöympäristöön ja sitä kautta ymmärtämään paremmin järjestelmän käyttöä.

Haastattelussa tuli esille yrityksen tapa tehdä kehityksen aikana listausta käytettävyyttä kohtuuttomasti haittaavista ratkaisuista. Näihin oli pyritty keksimään vaatimukset täyttäviä vaihtoehtoja, joihin sitten oli pyydetty auditoijan näkemyksiä auditointien yhteydessä. Haastateltavien mukaan kaikessa tietoturvakehityksessä tulisi kulkea rinnalla muistutus, että tietoturvaratkaisut eivät saisi sabotoida järjestelmän käytettävyyttä.

Käyttötapaukset ja tavoitetilakuvaukset olivat auttaneet toimittajaa hahmottamaan halutun korkeamman tason toiminnan ja käyttöympäristön. Haastateltavan mukaan asiakkaan olisikin tärkeintä kuvata käyttötapauksien kautta ohjelmiston toiminnallisuudet yksittäisten tietoturvavaatimusten sijaan. Tulisi kirjoittaa visio ja toimintojen kautta niin sanotut sateenkaarivaatimukset. Siitä olisi helpompi hahmottaa, mitä tilattavalla tuotteella oikeasti tavoitellaan. Myös tietoturvavaatimuksista tulisi kirjoittaa käyttötapauksia, joista selviäisi järjestelmän tietoturvatoiminnot ennen kaikkea silloin, kun se olisi liittyneenä isompaan järjestelmäkokonaisuuteen. Kun toimittaja oli ymmärtänyt järjestelmän aiotun käytön, oli ollut helpompi miettiä, mitä tietoturvaratkaisuja järjestelmään kannattaisi toteuttaa.

Vaikka osa haastateltavista toivoi tietoturvan määrittämistä toiminnallisuuslähtöisesti, haluttiin kuitenkin toiminnallisten vaatimusten lisäksi tietoturvavaatimuksia. Osa haastateltavista uskoi, että toimittajan olisi selkeämpi toteuttaa tietoturvaratkaisuja, kun olisi käytössä yleisistä ohjeista johdetut tietoturvavaatimukset.

6.2.6 Suojaustason määritys

Katakrissa (2015) mainitaan, että se ei pysty ottamaan huomioon tietoturvamäärityksissä kaikkia erikoistapauksia ja ympäristöjä. Sieltä ei siis pysty saamaan suoria vaatimuksia tai toteutusesimerkkejä yksittäisiin järjestelmiin. Salassa pidettävän tiedon suojaus aiheuttaa kustannuksia ohjelmiston kehityksessä, joten tieto tulisi luokitella selkeästi, sekä sen käyttö tulisi rajata mahdollisimman pienelle alueelle (Puolustusministeriö, 2015).

Usean haastateltavan mukaan suojaustasoa ei oltu määritetty yksiselitteisesti vaatimusmäärittelyssä, vaan oli esimerkiksi vain puhuttu, että ohjelmiston tulisi olla tietoturvallinen. Myöskään kehitysvaiheen alussa ei oltu kerrottu selkeästi, minkä tasoista tietoa järjestelmässä tultaisiin käsittelemään. Toimittajat toivoivat, että tieto luokiteltaisiin sille tasolle, mille se aidosti kuului ja luokittelun tulisi olla yksiselitteinen. Haastattelujen mukaan tietty suojaustaso oli voitu määrittää tavoitteeksi. Vaatimuksista ei kuitenkaan ollut selvinnyt, mitä se tarkoitti tietoturvamielessä kehitettävän laitteen osalta.

Toimittajat olivat kohdanneet ristiriitaisia tilanteita suojaustasojen monimutkaisten määritysten vuoksi ja toivoivat, että asiakas ottaisi nämä huomioon tietoturvavaatimuksissa. Esimerkiksi tuotteella oli voinut olla eri suojaustaso kuin sen käyttöympäristö oli pystynyt tukemaan. Lisäksi yksittäisen tiedon suojaustaso oli ollut jotain tiettyä, mutta yhdistettynä muuhun tietoon, koko tietomassan suojaustaso oli voinut nousta. Suojaustasonmuutos olisi pitänyt pystyä määrittämään selkeämmin. Suojaustasoja mietittäessä tiedon ajallinen perspektiivi pitäisi myös ottaa huomioon. Haastateltavien mukaan tieto saattoi vaatia korkeaa suojausta tiettynä ajanhetkenä, mutta jonkun ajan kuluttua suojauksen tarve oli pienentynyt.

Tiedon suojaustasot olivat vaikuttaneet myös ohjelmiston kehitysympäristöön. Haastateltavat toivoivat, että ohjelmisto voisi olla suurelta osin täysin julkista ohjelmistoa ja suojausta vaativa osa tehtäisiin sitten erikseen suojatuissa tiloissa. Mikäli kehitystyössä käytettävä materiaali oli leimattu varalta vaativampaan suojaustasoon, se oli ohjautunut suoraan suljettuun verkkoon. Tämä oli aiheuttanut ylimääräistä työtä toimittajalle. Haastateltava toivoikin materiaalin suodatusta, jotta suojaustasovaatimuksia saataisiin helpotettua. Vaikka mikään järjestelmä ei pysty huomioimaan suojaustason muutosta

automaattisesti, yksi toimittaja kertoi yrityksen kehittämästä sovelluksia, joissa voitaisiin käyttää teknisiä suodattimia tiedon riisumiseen ja karsimiseen alemman tason tiedon tuottamiseksi.

Tiedon luokittelu oli myös auditoijan kannalta olennaista. Auditoija oli tarkastanut ohjelmistot niitä vaatimuksia vasten, mihin tasoon ne oli kirjoitettu. Jos ohjelmistossa oli käytössä useampi suojaustaso, se oli otettu tarkastuksessa huomioon.

6.2.7 Käyttöympäristö ja järjestelmien integraatio

Jotta tietoturvan toteuttaminen järjestelmään onnistuisi, järjestelmän käyttöympäristö täytyisi ottaa huomioon jo ohjelmiston suunnitteluvaiheessa. Haastateltavat totesivat, että puolustusvoimilla on järjestelmiä monenlaisissa ympäristöissä. Osa järjestelmistä tehtiin fyysisesti hyvin suojattuihin tiloihin, kun taas osa järjestelmistä tuli ulkokäyttöön, suojauksen ulkopuolelle. Ympäristöstä huolimatta, kaikissa tiloissa piti pystyä käsittelemään eri suojaustasoa vaativaa tietoa.

Koska on järjestelmiä, joita käytetään tuolla kuusen juurelle ja niitä, jotka on syvällä luolassa, ni niillä on niin erilainen se ympäristö. Että kun niitä ajatellaan ikään ku sen Common Criterian mukaisen sipulin näkökulmasta, ni sehän on ihan eri ne elementit, mitkä on siinä sipulissa sit sen järjestelmän ulkopuolella. Et ne vaatii ihan erilaisen turvan ne tuol ulkona käytettävät softat. Eikä siihen riitä se, että sanotaan, että otetaan mies rynkyn kanssa. Ei se oo eihän se mihinkään riitä, et se ei oo vastaus. Et kyllähän siinä puhutaan jo täysin toisenlaisista, vielä tiukemmista vaatimuksista.

Todettiin myös, että järjestelmien laajuus ja verkottuneisuus voisi aiheuttaa lisävaatimuksia järjestelmän tietoturvaan. Esimerkiksi maanlaajuisesti käytössä ollut järjestelmä vaati erilaista suojaa ulkoisilta uhkilta kuin paikallisesti käytetty järjestelmä. Kun tilan suojaustaso oli heikompi, eikä antanut riittävää suojausta kehitettävällä ohjelmistolle, vaatimusten oletettiin korvaavan ympäristön puutteita.

Tämä on taas tätä, että pitää tietää siinä vaiheessa, ku lähetään soveltaan, että se käyttöympäristö voi olla muutakin kuin kyseisen tason tila. Taas palataan siihen

vaatimukseen ja käyttöympäristöön. Ei voia olettaa, että ois kakkosen tila, jos on kakkosen laitteesta kyse tai kolmosen tila kolmosen laitteelle. Vaan se voi olla tuolla maastossa esimerkiksi, että se kovennus pitää tehä sillon muilla keinoin.

Toimittajat antoivat myös esimerkkejä, joissa osa ohjelmistoon vaadittavista tietoturvaominaisuuksista olisi pystytty korvaamaan erillisillä tietoturvalaitteilla tai tilan suojausominaisuusilla.

Koska ettei taas tehdä liikaa tietoturvaratkasuja itse sovellukseen, ku jo ympäristö ratkasee ne tietoturvaongelmat. Turha tehdä, joskus heittäny ajatuksen, et turha tehdä kahta kassakaappia ja pistää ne vielä sisäkkäin. Tämmösiä välillä, ku lukkee niitä vaatimuksia, ni välillä tulee semmonenki fiilis, että niin sanottua double check-tehdään siellä. Että varmasti on turvallinen, että toimitilaratkasulla voi jo monta asiaa ratkasta ja vielä järjestelmänki pitäis ratkasta samat asiat.

Suunnittelijat eivät välttämättä olleet nähneet kohdeympäristöä ja olivat joutuneet tekemään oletuksia ympäristöä koskevista toiminnoista. Heillä oli omat kohdeympäristöä jäljittelevät testausympäristöt ohjelmistojen testausta varten, mutta viimeisin testaus oli yleensä tehty käyttäjien toimesta kohdeympäristössä. Asiakkaan tekninen ympäristö oli voinut muuttua ohjelmiston kehitystyön aikana, joten toimittajilla ei välttämättä ollut testausympäristössään sama konfiguraatio, jota asiakas käytti.

Yksi haastateltavista ehdotti rajapintojen ja ympäristön jäädytystä viimeistään testausvaiheessa, jolloin testaus tehtäisiin samalla laitteistokokoonpanolla sekä samoilla asetuksilla kuin ohjelmistoa tultaisiin oikeasti käyttämään. Tällöin toimittajat pystyisivät selvittämään ohjelmiston käyttäytymisen mahdollisimman tarkasti omien testien perusteella. Jälkikäteen tehtävien korjausten määrää voitaisiin näin huomattavasti vähentää.

Yksittäisen tuotteen osalta toimittaja oli hakenut asiantuntijalta edeltä käsin varmistuksen, että ohjelmiston tietoturvaratkaisut sopisivat asiakkaan ympäristöön, eikä niistä aiheutuisi mitään ongelmia asennuksen jälkeen. Käyttöympäristön huomioiminen tuli esille myös tietoturvatarkastuksissa. Auditoijan mukaan tiedon suojaustason määrittämisessä ei välttämättä oltu otettu huomioon käyttöympäristön suojaustasoa. Katakrin mukaan tämä

olisi poikkeuksellista, koska käyttöympäristön suojaustason tulisi tukea järjestelmässä käytetyn tiedon suojaustasoa.

Nykyiset ohjelmistot ovat riippuvaisia moderneista käyttöjärjestelmistä, organisaatiossa jo käytössä olevista tietokannoista, vanhoista järjestelmistä sekä myös nopeista verkkoratkaisuista. Tietotekninen ympäristö on vahvasti integroitunut. Näin ollen ohjelmiston tietoturvasta on tulossa suunnittelun, kehittämisen ja ohjelmiston käytön keskipiste. Yhdenkin osan vaarantuminen järjestelmäkokonaisuudessa voi vaikuttaa ohjelmiston toiminnallisuuksiin. Tästä johtuen ohjelmiston tietoturva täytyy ottaa huomioon hyvin aikaisessa vaiheessa ohjelmistokehitysprosessia. Kaikki tietoturvariippuvuudet täytyy arvioida riskiarvio, laatu ja arkkitehtuuri huomioon ottaen (H. Wang & Wang, 2003).

Myös haastateltavien mukaan tietoturva pitäisi pystyä määrittämään koko järjestelmäkokonaisuudelle, jotta tietoturva toteutuisi yhtenäisesti. Yksittäisten järjestelmien rajapinnat määräytyvät vieressä olevista järjestelmistä. Ne tulisi selkeästi määrittää kaikille järjestelmille. Haastateltavien mukaan samat tietoturvavaatimukset eivät käyneet kaikille järjestelmille, vaan vaatimuksissa tulisi tarkastella järjestelmäkokonaisuutta ja siitä johdettuja yksittäisten järjestelmien tietoturvavaatimuksia. Jotta toimittaja pystyisi ottamaan rajapintoihin liittyvät tietoturva-asiat huomioon, asiakkaan tulisi pystyä kuvaamaan tietoturvaominaisuudet ja integrointivaatimukset toimittajille.

Kun aletaan toteuttaa järjestelmää, jossa on tietoturvavaatimukset, niin monastihan nämä järjestelmät on kytkeytyneenä muihin järjestelmiin ja silloin kokonaisuuden hahmottaminen on tärkeetä ja siinä voi tulla haasteita, että se kokonaisuus on tavallaan niissä tietoturvamäärityksissä otettu huomioon ja siinä voi tulla kysymyksiä myös asiakkaalle, että miten nämä muut järjestelmät sitten käyttäytyy, kun tähän laitetaan tämmönen vaikka autentikointi. Eli yhteensopivuus näissä järjestelmissä ja että ne tietoturvavaatimukset on vielä siinä mukana.

Järjestelmien integrointi koettiin itsessään vaativaksi kokonaisuudeksi. Vaativuutta oli lisännyt, kun järjestelmään oli haluttu vahva suojaus tietoturvaratkaisuilla. Suojaus oli

täytynyt ottaa erityisesti huomioon järjestelmien integraatiossa, jotta tietoturva oli pysynyt yhtenäisenä, eikä ollut jäänyt aukkoja. Integraatioon liittyvät tietoturvavaatimukset olivat tulleet monesti ympärillä olevista järjestelmistä.

Tietoturvaa oli kehitetty eri aikaan järjestelmiin, joten tietoturvan toteuttamisessa olisi pitänyt ottaa huomioon muiden järjestelmien tietoturvakehitys. Pidettiin järjettömänä vaatia yhdeltä järjestelmältä tietoturvaominaisuutta, jota integroituva järjestelmä ei tukisi. Rajapintojen määrittäminen heti projektin alussa katsottiin selkeyttävän tietoturvan toteuttamista. Jos toimittajat eivät olleet tienneet järjestelmäkokonaisuutta, he olivat lähteneet kehittämään järjestelmää sen omista tietoturvavaatimuksista käsin. Siksi toivottiin, että rajapinnat ja niihin liittyvät tietoturvaominaisuudet dokumentoitaisiin huolellisesti. Lisäksi dokumentissa tulisi erottaa selkeästi asiakasympäristön ja toimitettavan ohjelmiston vastuulle kuuluvat tietoturvaominaisuudet.

Rajapintamäärittelyn lisäksi toimittaja haluaisi laitteistot, joita vasten kehitettävää ohjelmistoa pystyisi testaamaan. Hankintasopimuksessa tulisi selkeästi sopia, mitkä laitteet asiakas toimittaa testausympäristöön. Toimittaja lisäksi toivoi, että asiakas olisi velvollinen toimittamaan kaikki tarvittavat laitteet, koska laitteiden tulisi olla testeissä juuri oikeaa versiota. Asiakkaan laitteet eivät välttämättä olleet aivan uusinta laitekantaa, joten niitä oli ollut vaikea saada kaupasta tai mistään muualtakaan.

6.2.8 Vaatimusten tarkkuudesta

Kaikissa haastatteluissa tuli esille, että asiakas oli antanut tietoturvavaatimukset liian yleisellä tasolla. Niitä ei oltu kirjoitettu selkeästi vaatimusmäärittelyyn, joten toimittaja oli joutunut tulkitsemaan vaatimuksia Katakrista ja VAHTI-ohjeista.

Niistä (tietoturvavaatimuksista) osalla on luonne tosi yleisluontonen. Ja se on tosi laaja. Mä poimin muutaman esimerkin, että ymmärtää, mikä se laajuus on. Meillä ne oleellisimmat vaatimukset on kuulunu näin: "Järjestelmien tulee mahdollistaa liityntöjen toteutus eri suojaustason järjestelmiin lakien, asetusten ja normien ohjeistamalla tavalla." Ja toinen esimerkki: "Järjestelmän pitää täyttää valtioneuvoston asetuksessa Tietoturvallisuus valtionhallinnossa -määritellyt tietoturvavaatimukset."

Kun pohdittiin sopivaa vaatimusten tarkkuutta, vastauksissa oli aika paljon vaihtelua. Yhtenä ehdotuksena oli, että tietoturvavaatimusten tulisi olla niin selkeitä ja yksiselitteisiä, että ne pystyttäisiin todentamaan. Vaatimuksen toteutuminen voitaisiin selkeästi testata tai mitata.

Monessa keskustelussa tuli myös esille, että vaatimuksia tulisi kuvailla kattavammin kuin pelkästään mainitsemalla tietoturvaominaisuus. Tulisi määrittää, mitä vaatimuksella tarkoitetaan sekä mitä sillä järjestelmän ominaisuutena tarkoitettaisiin. Esimerkiksi jos tuotteeseen haluttaisiin kirjautuminen, tulisi määrittää, minkälaisesta kirjautumisesta olisi kyse. Vaatimukseksi ei riittäisi, että ”tuotteeseen pitää pystyä kirjautumaan turvallisesti”. Vaatimusta tulisi avata enemmän, esimerkiksi: ”kirjautuminen pitää tapahtua aina omalla identiteetillä”.

Yksi haastateltava mietti, miksi tietoturvavaatimuksissa ei voitaisi määritellä suoraan toteutustapaa, jos se oli asiakkaalla tiedossa. Esimerkiksi asiakas ei ollut käskenyt käyttämään tiettyä salausta, vaikka hän oli tiennyt sen täyttävän vaatimukset. Toisaalta useassa haastattelussa oltiin sitä mieltä, että vaatimus ei saisi sitoa teknistä ratkaisua. Kunhan tavoite olisi selkeä, toimittajalla tulisi olla valinnanvapaus toteutukseen.

Vastauksissa tuli myös esille, että tietoturva-asioissa ei pitäisi tehdä oletuksia. Asiakkaan ei pitäisi olettaa toimittajan tekevän tai tietävän asioita, joista ei oltu sovittu. Toimittaja katsoi tietoturva-asioita hyvin eri näkökulmasta, eikä voinut tietää asiakkaan oletuksia, mikäli niistä ei oltu sovittu tai edes keskusteltu. Esimerkiksi asiakkaan toimintaympäristöön liittyvät vaatimukset koettiin niin erityisiksi, että toimittajat tai auditoijat eivät olleet pystyneet niitä huomioimaan, jos niitä ei oltu kuvattu heille riittävällä tarkkuudella. Kun tietoturvan tavoitteet oli kuvattu selkeästi sekä ominaisuudet oli kirjoitettu selkeästi vaatimuksiksi, tietoturvaratkaisut oli helpompi toteuttaa oikein ja myös auditoijan oli helpompi tehdä tarkastus.

6.3 Kollaboraatio ja asiantuntijoiden käyttö

Tietoturvan kehittämisessä yhteistyö eri toimijoiden välillä koettiin hyvin tärkeäksi. Toimittajat olivat jonkin verran tehneet yhteistyötä puolustusvoimien hankkimien ulkopuolisten asiantuntijoiden kanssa, mutta kuitenkin yhteistyö myös asiakkaan kanssa koettiin tarpeelliseksi. Haastatteluissa tulikin paljon ehdotuksia, miten yhteistyötä tulisi tiivistää ja kehittää.

6.3.1 Asiantuntija-apu

Puolustusvoimat on käyttänyt ulkopuolisia tietoturva-asiantuntijoita järjestelmien auditoinneissa ja tietoturvan kehittämisessä, joten haastateltavilta kysyttiin kokemuksia asiantuntijoiden käytöstä. Muutamissa yrityksistä asiantuntija-apua oli ollut hyvin niukasti ja kokemukset siitä vaihtelivat. Yhdessä yrityksessä ulkopuolinen asiantuntija oli ollut käytössä jossain vaiheessa kehitystyötä. Yhteistyö ei ollut oikein toiminut, koska oli jäänyt sopimatta, kuinka asiantuntijaa käytettäisiin ja kuka häntä ohjaisi. Muissa yrityksissä asiantuntija-apua oli käytetty ja siitä oli positiivisia kokemuksia

Asiantuntijat olivat käyneet tekemässä esiauditointia ja heidän kanssaan oli suunniteltu myös koulutuksesta. Tukea haluttaisiin jatkossakin ja mielellään samoilta henkilöiltä, koska he tunsivat jo kehitettävän tuotteen ongelmat. Asiantuntijat koettiin hyvänä asiana, koska he pystyivät jakamaan tietoa siitä, miten tietoturvaa pitäisi toteuttaa. Tietoturva-asiantuntija oli ollut mukana myös järjestelmän määrittelytyössä sekä kehitystyön alkuvaiheessa, kun oli tehty isoja tietoturvaan vaikuttavia valintoja. Lisäksi hän oli ollut mukana lukemassa ja kommentoimassa tietoturvadokumenttia.

Asiantuntijan käyttöä pidettiin tarpeellisena koko tuotteen kehitystyön ajan. Olisi hyvä olla mukana henkilö, joka ymmärtäisi, miten auditointeja tehdään. Hän voisi auttaa tietoturvan toteutuksessa, koska tietoturvaratkaisuja tuskin pystyttäisiin etukäteen määrittämään tarpeeksi tarkasti. Auditoija voisi auttaa toimittajaa ymmärtämään tietoturvavaatimuksia, jotta tietoturva miellettäisiin osaksi kokonaishankintaa. Tämä voisi vähentää yllätysten määrä auditoinneissa.

Ulkopuoliset auditoijat koettiin hyödyllisiksi, koska järjestelmään perehdytyksen jälkeen, heiltä sai paljon tietoa. He tietävät viimeisimmät tietoturvaan liittyvät asiat, joten toimittaja koki oppivansa paljon jutellessaan auditoijien kanssa. Mikäli auditoiva taho olisi antamassa aktiivisemmin suosituksia heidän mielestään hyviin ratkaisuihin, uskottiin että, auditoinnit voisivat mennä paremmin. Asiantuntijoita olikin käytetty ohjelmistokehittäjien kouluttamisessa.

Ulkopuolisten asiantuntijoiden käyttö koettiin osittain myös tarpeettomaksi. Yhdellä yrityksellä oli itsellään tietoturva-asiantuntemusta niin paljon, että se ei tarvinnut ulkopuolista apua. Toisessa yrityksessä todettiin oman tietoturvayksikön antaneen apua teknisiin ratkaisuihin ennen kaikkea tuotekehityksen aikana.

Puolustusvoimista on myös annettu asiantuntija-apua yrityksille tietoturvan kehittämiseen. Apu koettaisiin hyödylliseksi, jos se vain olisi säännöllistä koko kehitysprojektin ajan. Muutaman päivän mittaisessa avussa palaute oli voinut jäädä vähän liian vajaaksi. Tietoturva on otettava huomioon kokonaisuutena, joten asiantuntijan käyttöä yksittäiseen erityiseen ongelmaan ei koettu riittäväksi.

Rajauksia ja ohjausta tarvittaisiin missä tahansa projektin vaiheessa. Uskottiin, että puolustusvoimilla olisi korkealla tasolla paras näkemys ympäristöön. Tuli kuitenkin esille, että teknisellä tasolla ulkopuolinen toimija, joka työskentelee juurikin teknisten asioitten kanssa, voisi antaa parhaiten vastauksia teknisiin kysymyksiin. Arveltiin, että tietoturvaihmisten apua ei kaivattaisi niin paljon, jos puolustusvoimat tekisi ohjeita tietoturvasta. Tällöin kaikille toimittajille ei tarvitsisi kertoa samoja asioita erikseen.

Yleisesti toivottiin, että kehityshankkeessa olisi puolustusvoimilta tietoturvavastuullinen mukana koko projektin elinkaaren ajan. Hänen vastuullaan olisi varmistaa, että lopputuote vastaa tarvittaviin kriteereihin. Tuli esille myös esimerkki, jossa tietoturva-asiantuntija oli saatu mukaan osatoimituksiin. Hän oli saanut niin hyvän kokonaiskuvan järjestelmästä, että häneltä oli saatu apua myös toteutusratkaisuihin.

Yksi haastateltava oli kokenut turhauttavana viiveet, mitä puolustusvoimien asiantuntija-avun saannissa oli ollut. Kehitystyö on hektistä ja sykli nopea. Olisi pitänyt olla suora kontakti henkilöön, jotta kysymyksiin olisi saatu vastaukset riittävän nopeasti. Kun

vastauksia oli jouduttu odottamaan, suunnittelijalle tai tiimille ei välttämättä ollut löytynyt muuta tekemistä. Näin ollen suunnittelija oli joutunut miettimään ratkaisut osaamisensa ja kokemuksensa pohjalta. Sitten oli vain toivottu, että valitut ratkaisut olisivat riittävät. Haastateltava toivoikin, että projektin alussa sovittaisiin kontaktihenkilö, jolle voisi soittaa suoraan tietoturvaan liittyvistä asioista.

Lisäksi ehdotettiin tietoturvaihmisten puolustushaarakohtaisia kokoontumisfoorumeja muutaman kerran vuodessa. Siellä asiakkaan tietoturva-asiantuntijat kertoisivat uusimmat tietoturvaan liittyvät tiedot. Näin saataisiin keskusteluyhteys mahdollisimman aktiiviseksi asiakkaan tietoturvaihmisten kiireistä huolimatta. Tiedon vaihtoa toivottiin mahdollisimman paljon, että toimittaja tietäisi, mihin suuntaan asiakas toivoo tietoturva-asioiden kehittyvän. Näin vähennettäisiin tietoturvassa eteen tulevia yllätyksiä.

Yhdellä haastateltavalla oli kokemus, että puolustusvoimien tietoturva-asiantuntija oli käyttänyt tietoturvatyökalua kehitteillä olevan järjestelmän testaamiseen projektin aikana. Tilanne oli muuttunut, kun puolustusvoimat alkoi käyttää ulkopuolisia auditoijia. Haastateltava ehdotti, että asiakas voisi lainata tätä työkalua projekteille, jolloin saataisiin resurssit maksimaaliseen käyttöön.

6.3.2 Yhteistyö

Haastateltavat kokivat yhteistyön asiakkaan kanssa tärkeäksi tietoturva-asioissa. Yhteistyöstä tuli esille monenlaisia kokemuksia, ennen kaikkea sen haluttiin olevan jatkuvaa ja vuorovaikutteista ohjelmiston kehitystyön eri vaiheissa. Asiakkaalta haluttiin päätöksiä ja kannanottoja tietoturva-asioihin.

Useat haastateltavat kokivat, että asiakkaan kanssa yhteistyö oli sujunut hyvin. Oli löydetty yhdessä järkeviä tietoturvaratkaisuja, jotka eivät olleet liian vaativia toteuttaa. Osa haastateltavista toivoi yhteistyötä lisää, jotta jatkossakin löytyisi asiakkaan käyttöön soveltuvia ratkaisuja. Tarpeelliseksi koettiin myös keskustelut eri vaatimusten tarkoituksista. Mitä aikaisemmassa vaiheessa kyettäisiin keskustelemaan myös vaatimusten todentamisesta, sitä paremmalla pohjalla toimittaja olisi hankkeeseen lähtiessään.

Toisaalta asiakas koettiin kankeaksi neuvottelijaksi. Toimittajalla oli ollut vaikeuksia saada ehdotuksia läpi tai löytää sopivia vaihtoehtoja tietoturvaratkaisuihin. Asiakkaan edustaja ei ollut oikein hyväksynyt toimittajan esittämiä ratkaisuja tietoturvaan vedoten. Oli myös koettu, että jos toimittajan ehdotusta oli lähdetty viemään eteenpäin, se oli johtanut hyvin pitkiin keskusteluihin ja pitkään prosessiin. Tällöin projekti ei enää ollut pysynyt sovitussa aikataulussa.

Säädöksissä ja muissa ohjeissa tietoturvavaatimukset eivät ole sillä tasolla, että toimittaja pystyisi niistä päättelemään hyväksyttävät tekniset ratkaisut. Lisäksi koetiin, että ohjeista ei pystynyt päättelemään, mikä ratkaisu tulisi hyväksytyksi asiakaan tietoturvapäällikön toimesta. Tulkintaan tarvittaisiin molemmin puolinen hyväksyntä ennen kuin suunnittelu lukittaisiin. Vaikka tietoturvavaatimuksia ei oltu saatu, tuotteilta vaaditaan tietoturvaominaisuuksia, joita toimittaja oli joutunut itsekseen miettimään ja pähkäilemään. Kokemus oli auttanut toimittajia vaatimaan asiakkaalta riittävän tarkkoja ja konkreettisia vastauksia, jotta he ei olleet joutuneet tekemään omia tulkintoja eikä ottamaan riskiä tulkintojen onnistumisesta. Yhdelle toimittajalle tietoturvahenkilöiden kiire oli näkynyt siten, että heitä ei oltu saatu useista yrityksistä huolimatta samaan palaveriin, eikä tietoturvavaatimuksia oltu siten määritetty järjestelmään.

Yhdessä kehityshankkeessa asiakas ja toimittaja olivat toimineet ketterän kehityksen mallin mukaisesti. Siinä oli yhdessä määritetty, kehitetty ja keksitty lisää ominaisuuksia iteratiivisen toimintatavan mukaisesti. Toimittaja oli kokenut tällaisen menettelytavan erittäin hyvänä.

6.3.3 Osaamisen kehittäminen

Toimittajat olivat opiskelleet tietoturva-asioita työn kautta, ja esimerkiksi Katakria oli tullut tutuksi. Lisäksi auditoinneissa oli opittu paljon uusia asioita tietoturvasta, koska tarkastukset olivat olleet pitkiä ja perusteellisia. Puolustusvoimilta oli saatu jonkin verran tietoturvaan liittyvää koulutusta ja sitä haluttiin lisää.

Puolustusvoimien tietoturva-asiantuntijoiden tai turvallisuusviranomaisten toivottiin kertovan uhkista ja muista tietoturvaan liittyvistä asioista päivitysmielessä toimittajille.

Heidän toivottiin kertovan järjestelmiin kohdistuneet uusimmat uhkat, sekä toivottiin CERT:n tietoturvahaavoittuvuuksien ja ohjelmistopäivitysten läpikäyntiä.

Puolustusvoimien asiantuntijan kanssa oli ollut keskustelua, että hän antaisi koulutusta järjestelmäkehityksen tietoturvaan liittyen, mutta se oli jäänyt keskustelun tasolle. Isompien sopimusten yhteydessä tietoturvavastaava tai -päällikkö oli antanut jollekin yritykselle hankekohtaisia koulutuksia. Näissä oli käyty läpi tietoturvakäytänteitä, ei niinkään tietoturvan kehittämistä järjestelmiin.

Toimittajat olivat kautta linjan hakeneet ulkopuolista koulutusta tietoturvaosaamiseensa. Osa koulutuksista oli pyydetty auditointiyrityksistä, jolloin oli saatu tietyn järjestelmän kehittämiseen tarvittavaa koulutusta. Paljon oli myös käyty koulutuksissa ulkomailla ja osaamista oli kehitetty erilaisilla kursseilla. Tietoturvan kehitystä järjestelmiin oli opittu paljon tekemisen kautta. Ensimmäisen hyvin tehdyn projektin jälkeen oli helpompi lähteä tekemään uutta kehitystyötä. Toivottiin, että suunnittelijat pääsisivät näkemään tai kyselemään riittävästi puolustusvoimien toimintamalleista ja ympäristöstä, koska niiden koettiin muuttuvan koko ajan.

Osa piti tärkeänä tietoturvakoulutusta organisaation laajuisesti, jotta osaamiskulttuuria pystyttäisiin nostamaan. Toimittajilta tuli kommentti, että järjestelmien käyttäjille tulisi myös antaa tietoturvakoulutusta. Näin käyttäjät tietäisivät, miten järjestelmiä tulisi käyttää tietoturvallisesti. Tietoturvan toteutumisen kannalta isona tekijänä pidettiin käyttäjien tietoturvakäyttäytymistä.

6.4 Kehitysideoita

Haastatteluissa tuli esille monenlaisia ideoita asiakkaan tietoturvan, toiminnan ja auditointien kehittämiseksi. Tähän kappaleeseen on pyritty poimimaan sellaiset kokonaisuudet, jotka mahdollisesti voisivat antaa asiakkaalle uusia näkemyksiä tietoturvan kehittämiseen ja siihen liittyvään toimintaan.

Yksi haastateltava toi esille, että nykytrendi tietoturvan kehittämisessä on pyrkiä huomaamaan poikkeamat ja reagoimaan niihin äärimmäisen nopeasti. Lukuisia

väärinkäytöksiä tai hyökkäyksiä ei edes pyritä estämään. Yleisessä käytössä olevien tietoverkkojen osalta tällainen menettely on jo käytössä. Tietoverkot ovat niin monimutkaisia, että hyökkäykset niihin on mahdotonta estää. Täydellisestä estämisestä on siirrytty hyökkääjän tunnistamiseen, jotta siihen vaikuttaminen pystyttäisiin tekemään mahdollisimman nopeasti.

Haastattelussa tuli ehdotus, että tietoturvavaatimuksista tulisi kirjoittaa tietoturvaspesifikaatio. Siinä toimittaja pystyisi puntaroimaan eri tietoturvamekanismeja sekä pystyisi miettimään auditointia varten, miten järjestelmää pyritään suojaamaan. Moneen suojaustoimenpiteeseen on useita toteutusvaihtoehtoja, joten yritys voisi puntaroida asiakkaan kanssa vaihtoehtoja sekä dokumentoida oman näkemyksensä tietoturvan toteutuksesta. Näin saataisiin optimaalinen ratkaisu kustannusten ja ylläpidon kannalta. Kaksi yritystä olivat projekteissaan tehneet tietoturvakuvauksen ja pitivät sitä hyvin tärkeänä dokumenttina. Siitä oli ollut hyötyä myös auditoijille, koska he pystyivät tutustumaan sen avulla auditoitavaan järjestelmään ja saamaan vaatimuksia auditointiin.

Tuli myös ehdotus, että tarjouspyyntövaiheessa toimittajille järjestettäisiin yhteinen infotilaisuus, jossa kerrottaisiin auditoinnin vaiheet ja mitä eri vaiheissa tehdään. Tarjouspyynnön liitteessä voisi myös kertoa, mitä auditointi tarkoittaa ja mitä toimittajan tulee ottaa huomioon sen osalta tarjousta kirjoittaessaan.

Haastattelussa tuotiin myös esimerkki Iso-Britannian hankintatoiminnasta. Siellä on käytössä hankintojen osalta standardimenettely, jossa on esikysely tarjouskilpailuun halukkuutensa ilmoittaneille yrityksille. Esikyselyssä yrityksen tulee täyttää Iso-Britannian turvasäädökset. Kehitystyötä varten tulee olla riittävän turvalliset tilat, toimintatavat, prosessit sekä kaikki muut turvaratkaisut. Ne tulee olla auditoituna hyväksytyn tahon toimesta ja yrityksellä tulee olla siitä sertifikaatti. Vasta sertifikaatin saatuaan yritys pääsee näkemään varsinaiset tietopyyntökysymykset. Haastateltava arvioi, että valtio varmasti säästää tällä omaa työtään ja mietti menettelyn sopivuutta myös Suomessa käytettäväksi.

7 Johtopäätökset

Toimittajat osallistuivat tutkimukseen aktiivisesti. Heillä oli paljon ehdotuksia tietoturvan ja yhteistyön kehittämiseen. Tulosten mukaan toimittajat olivat kiinnostuneita tietoturvaan liittyvistä asioista ja pitivät tietoturvaa yhtenä tärkeänä painopistealueena liiketoiminnassaan. Tutkimuksen tuloksista saadut johtopäätökset käsitellään tutkimuskysymysten perusteella. Hankintaprosessin kehittämisestä saatujen tulosten yhteenveto on ensin, sitä seuraa vaatimusmäärittely sekä yhteistyön kehittämisestä kootut lopputulokset.

7.1 Hankintaprosessin kehittäminen

Ohjelmistokehitysprosessin tarkoituksena on tukea ohjelmiston kehitystä siten, että valmis tuote antaa lisäarvoa asiakkaalle ja myös muille sidosryhmille. Toimittajat käyttivät kehitystyössään eri kehitysprosessimenetelmiä, kuin asiakkaalla oli käytössä. Haastattelujen mukaan tämä aiheuttaa ongelmia yhteistyöhön, mikä on näkynyt myös valmiissa tuotteissa. Toimittajat eivät haluaisi liian tiukkoja vaatimuksia kehitystyön alkuun, vaan vaatimuksista ja toteutusratkaisuista haluttaisiin keskustella asiakkaan kanssa koko kehitystyön ajan. Vaatimusten muuttamisesta toivottiin joustavampaa. Tutkimuskysymyksessä pohdittua asiakkaan hankintaprosessia tulisi siis kehittää enemmän ohjelmistokehitysmallin mukaiseen suuntaan, jotta yhteistyö ja asioista sopiminen sujuisivat paremmin.

Toimittajat toivat esille toimintamalleja, jotka tukevat ketterän ohjelmistokehityksen menetelmiä. Kuitenkin asiakkaan vaatimukset vahvasta tietoturvasta tuovat lisäpiirteitä ketteriin kehitysmenetelmiin, mitkä täytyy ottaa huomioon tietoturvaominaisuuksia kehitettäessä. Esimerkiksi ketterän kehityksen Scrum-malli ei sisällä lainkaan vaatimusmäärittelyä. Malliin on kuitenkin lisätty niin kutsuttu esi-sprintti, jossa vaatimukset kuvataan käyttäjäkertomuksina. Toimittajat toivoivat, että asiakas määrittäisi tietoturvavaatimuksia uhka-arvioiden, lakisääteisten ohjeiden tulkinnan, käyttöympäristön, käytettävyyden, suojaustason sekä rajapintamääritysten kautta. Scrumin esi-sprintin

mukainen käyttäjäkertomusten kirjoittaminen ei siten anna toimittajalle riittävää kuvaa halutusta tietoturvasta.

Toimittajien mukaan tietoturvavaatimuksia pitäisi toteuttaa iteratiivisesti, koska kaikkia tarvittavia tietoturvaratkaisuja ei pystytä suunnittelemaan kehitystyön alussa. Kuitenkin haluttiin, että vaatimusten todentaminen sovittaisiin jo kehitystyön alkuvaiheessa. Keskusteluja ja työpajoja tarvittaisiin, jotta tietoturvalle voitaisiin määrittää järkevät ylätason vaatimukset kehitystyön alkuun. Pitäisi myös sopia, miten tietoturvavaatimusten todentaminen tullaan tekemään. Tehdäänkö se auditoinneilla vai muilla tarkastuksilla, sekä monessako eri vaiheessa tarkastukset tehtäisiin? Iteratiivisuus antaisi myös asiakkaalle mahdollisuuden tarkastella vaatimuksia kehitystyön aikana. Mikäli työ kestää monia vuosia, tietoturvaohjeet ehtivät muuttua asiakkaan puolella useita kertoja, joten vaatimuksiin tulee muutospaineista.

Tietotuvan kehitys nähtiin monisäikeisenä ja monivivahteisena työnä. Asiakkaan olisi syytä osallistua ohjelmiston kehitystyöhön huomattavasti enemmän, kuin mitä nykyinen käytössä oleva prosessimalli ohjaa. Hänen toivottiin ottavan kantaa tietoturvaratkaisuihin, tekemään rajauksia tai hyväksymään muutoksia koko ohjelmiston kehitystyön ajan. Myös loppukäyttäjien mukana olo koettiin tärkeäksi. Haluttiin, että he näkisivät muutaman viikon välein kehitettävän sovelluksen, jotta he pystyisivät ottamaan kantaa toteutettuihin ratkaisuihin jo kehitystyön aikana.

Ohjelmistokehitysprosesseissa mukana olevat auditoinnit koettiin hyödyllisiksi ja myös mielenkiintoisiksi. Joissain yhteyksissä tuli ilmi, että auditointituloksien perusteella ei oltu ryhdytty kuitenkaan toimenpiteisiin. Auditointeja tehdään ja niiden tuloksiakin voidaan pohtia. Kuitenkin auditoinneista saatava hyöty voi jäädä aika laihaksi, jos ohjelmistojen tietoturvaa ei välttämättä korjata tulosten perusteella. Olisi syytä kiinnittää huomiota siihen, miten auditoinnit liitetään osaksi ohjelmiston kehitysprosessia. Tällöin auditoinneilla aidosti parannettaisiin ohjelmistojen tietoturvaa, eikä sitä pidettäisi vain ohjelmistokehityksen viimeisenä tarkastuksena, joka ei oikein johda mihinkään tai ainakaan asiakkaan toivomiin tuloksiin.

Siirtyminen vesiputousmallista iteratiiviseen toimintamalliin olisi asiakkaalle iso muutos. Uskalletaanko ottaa riskiä, että kaikkia vaatimuksia ei kirjoiteta aivan loppuun asti ja tarkasti? Toisaalta tietoturvavaatimusten osalta tätä riskiä on jo otettu, koska vaatimukset ovat voineet puuttua kokonaan tai ne ovat olleet niin ylimalkaisia, että toimittaja ei ole pystynyt niitä ymmärtämään. Muutos, että vaatimuksia muokattaisiin ymmärrettävämmiksi, pitäen ne kuitenkin ylätason vaatimuksina, ei välttämättä olisi kovin iso.

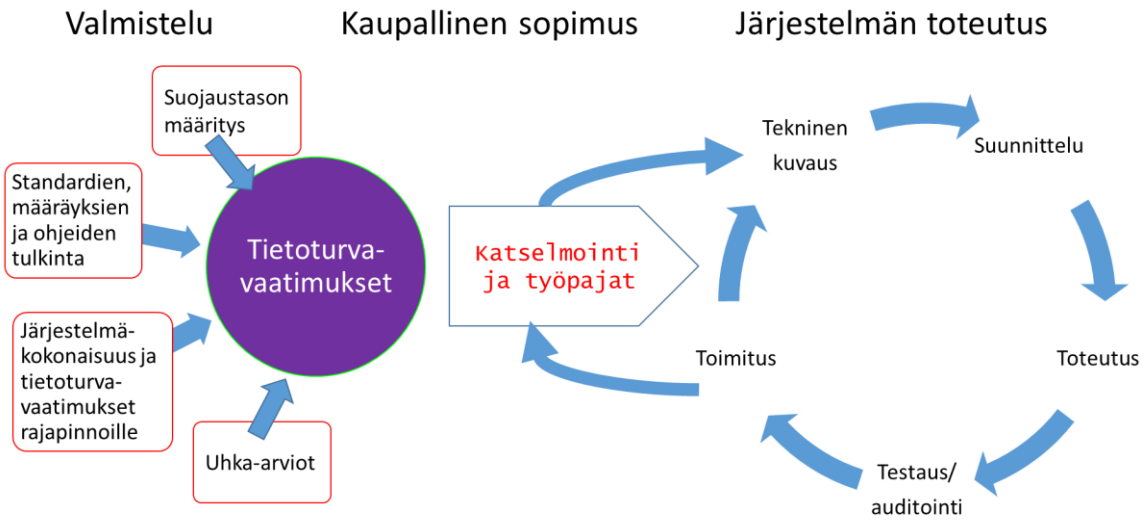
Isompi muutos olisi siinä, että asiakkaan pitäisi antaa koko kehitysprojektin ajan kommentteja tietoturva-asioihin. Tämä vaatii perehtymistä ja kouluttautumista asiakkaan puolella. Asiakkaan tulisi tuntea asettamiensa tietoturvavaatimusten taustaa, jotta hän pystyisi perustelemaan ne toimittajalle. Pitäisi myös osata tulkita valtionhallinnon tietoturvaohjeita ja soveltaa niitä tietoturvavaatimuksiin. Käytäntönä on ollut, että vastuu vaatimusten tulkinnasta siirretään toimittajalle. Tämä ei vaikuta oikein onnistuvan, koska toimittajat eivät tunne esimerkiksi asiakkaan ympäristöä, toimintatapoja, lainsäädäntöä tai järjestelmiin kohdistuvia uhkia.

Tutkimustulosten perusteella muodostettuun kehitysprosessimalliin (Kuvio 11) on poimittu tietoturvan onnistuneeseen kehittämiseen tarvittavat tekijät. Siinä tietoturvavaatimusten määrittäminen koostuu monesta osa-alueesta, kuten suojaustaso, järjestelmän ympäristö, uhka-arviot tai tietoturvaan liittyvien ohjeiden tulkinta. Vaatimuksia käsitellään tarkemmin seuraavassa alaluvussa.

Kaupallista sopimusta neuvoteltaessa ylätason tietoturvavaatimukset tulisi katselmoida ja tarvittaessa sopia työpajoja tietoturvavaatimusten selkeyttämiseksi. Järjestelmän kehitystyön aikana sidosryhmien tulisi osallistua tietyin väliajoin toistuviin toimituksiin. Näissä tilaisuuksissa päätetään, tarvitaanko seuraavaa suunnittelukierrosta varten vaatimusten muokkaamista. Mikäli tarvitaan, palaverissa tulisi sopia vaatimusten katselmointi ja tarvittaessa järjestää työpajoja uudelleen. Sidosryhmien tulisi osallistua myös muuhun järjestelmän kehitystyöhön, mikäli se nähdään tarpeelliseksi.

Tutkimuksessa ei arvioitu nykyisen rahoitusmallin sopivuutta iteratiiviseen kehitykseen siten, että jokaisella iteraatiolla olisi järjestelmän osatoimitus ja rahoitukseen vaikuttavaa

vaatimusmäärittelyä. Kuitenkin kaupallisen sopimuksen katselmointi täytyisi jollain tapaa pitää, jos todetaan, että kehitystyössä tulee rahoitukseen vaikuttavia muutoksia. Toisaalta osatoimituksista sopiminen on hyvinkin tuttua järjestelmähankinnoissa.



Kuvio 11. Tutkimustuloksiin perustuva kehitysprosessimalli

7.2 Tietoturvavaatimusmäärittely

Toimittajilla oli asiakkaan antamista tietoturvavaatimuksista hyvin monenlaisia näkemyksiä. Haastateltavien kokemukset vaatimuksista poikkesivat toisistaan, koska puolustusvoimilla ei ole yhtenäistä ohjeistusta tietoturvavaatimusten määrittämisestä. Tämä näkyy hyvin erilaisina vaatimuksina. Tutkimuksen yksi tärkein osa-alue oli pohtia tietoturvavaatimusten kehittämistä, jotta ne tukisivat toimittajia ohjelmistojen tietoturvan kehityksissä.

Pohdittaessa vaatimuksia toimittajien oli vaikea ymmärtää kaikkien asiakkaan antamien tietoturvavaatimusten tarpeellisuutta. He haluaisivatkin tietää, onko kaikilla annetuilla tietoturvavaatimuksilla käytännössä merkitystä. Ymmärrystä lisäisi, jos toimittajille pystyttäisiin kuvaamaan ohjelmistoon kohdistuvat uhkat ja asiakas arvioisi myös uhkien vaikutuksia.

Tietoturvavaatimusmäärittelyä pohdittaessa toimittajat korostivat ohjelmiston ympäristön huolellista määrittämistä. Tietoturvan toteutus tulisi myös suunnitella kokonaisuutena. Asiakkaalla pitäisi olla järjestelmäarkkitehti, jolla olisi näkemys asiakkaan järjestelmäkokonaisuuden tietoturvasta. Tämä auttaisi järjestelmien integroinnissa ja rajapintojen määrittelyssä. Kun tietoturva suunniteltaisiin kokonaisuutena, yksittäiset ohjelmistot voisivat käyttää järjestelmäkokonaisuudessa jo olevia tietoturvaratkaisuja. Tämä yhdenmukaistaisi tietoturvan hallintaa sekä helpottaisi järjestelmien ylläpitoa. Kun yhdellä henkilöllä olisi tiedossa koko järjestelmäkokonaisuus, hän pystyisi arvioimaan esimerkiksi yksittäisten tietoturvapäivitysten vaikutuksia järjestelmäkokonaisuuden toimintaan.

Hyvin suuria ongelmia tulee, jos tietoturvavaatimuksia ei ole annettu toimittajalle kirjallisesti. Toimittaja on itse joutunut tunnistamaan vaatimuksia ja kokenut sen hyvin haastavana, koska hän ei ole tiennyt, miten tietoturva-asioita arvioidaan myöhemmin. Tietoturvaominaisuuksien toteuttaminen ei ole halpaa, joten täytyisi miettiä, mikä olisi järkevää tehdä. Tietoturvassa korostuu vaatimusten tärkeysjärjestys ja asiakkaan näkemykset. Pitäisi tietää, mitä tietoturvalta halutaan ja mille suojaustasolla ohjelmiston toteutus tulisi tehdä.

Uhka-arvioiden tekeminen nousi tutkimuksessa esille monessa yhteydessä. Mikäli asiakas pystyisi osoittamaan ohjelmistoon kohdistuvat ulkoiset uhkat, se antaisi kehykset sekä ohjelmiston kehittäjille että auditoijille. Tällöin ohjelmistosuunnittelija pystyisi valitsemaan sopivat tietoturvaratkaisut näitä uhkia vastaan ja siten ne olisivat tarkoituksen mukaisia. Myös auditoijat pystyisivät miettimään testitapauksia määritettyjen uhkien näkökulmasta. Tämä toisi mielekkyyttä tietoturvan kehittämiseen, kun työlle olisi perusteet ja tavoitteet.

Katakrin käyttö ylimpänä tietoturvavaatimusten oppaana vaikuttaa harmilliselta jäänteeltä. Siitä tulisi päästä eroon ja paneutua myös asiakkaan puolelta tietoturvan miettimiseen siten, että vaatimukset tukisivat toimittajaa hänen pohtiessaan sopivia tietoturvaratkaisuja. Katakrissakin todetaan, että se ei pysty antamaan riittäviä suojausvaatimuksia kaikkiin erityistapauksiin tai ympäristöihin. Asiakkaan tulisi siis selvittää, mitkä

suojausvaatimukset ovat riittävät kulloinkin hankittavaan ja kehitettävään ohjelmistoon. Nämä kyseiseen ohjelmistoon sovelletut vaatimukset tulisi antaa toimittajalle ja auditoijalle. Lisäksi haastateltavien antamat kommentit Katakrin käytöstä tulisi pohtia tietoturvavaatimuksia mietittäessä.

Tutkimuksessa tuli myös esille, että asiakkaan tulisi pohtia tietoturvaa käytettävyyden näkökulmasta. Toivottiin, että tietoturvavaatimuksilla ei huononnettaisi merkittävästi järjestelmän käytettävyyttä. Esimerkiksi, mikäli käyttäjiin pystyttäisiin luottamaan, käyttöoikeussääntöjä voitaisiin yksinkertaistaa. Näin ne eivät paisuisi niin monimutkaisiksi, että itsessään aiheuttaisivat tietoturvaongelmia.

Kun haastateltavien kanssa pohdittiin tietoturvavaatimusten sopivaa tarkkuutta, vastauksissa oli aika paljon vaihtelua. Ajatukset sopivasta tarkkuudesta peilasivat luultavasti asiakkaalta saatuihin toisistaan poikkeaviin vaatimusmäärittelyihin. Jossain haastattelussa haluttiin asiakkaan kertovan hyväksytyt ratkaisut, kun taas useassa haastattelussa oltiin sitä mieltä, että vaatimus ei saisi sitoa kehitettävän tuotteen teknisiä ratkaisuja. Tässä ehkä tulee esille myös tietoturvaominaisuuksien kirjava luonne, kun osaan tietoturvaominaisuuksista voidaan käyttää monenlaisia hyväksyttyjä ratkaisuja. Mutta esimerkiksi tietoliikenteen salauksessa on hyvin tarkasti määrätty salausavaimen pituus ja käytettävä standardi.

7.3 Yhteistyö sidosryhmien välillä

Asiakasorganisaatiossa on ehkä vallitseva ajatus, että tietoturva-asiat kuuluvat tietoturva-asiantuntijoille, joten projektipäälliköiden ja järjestelmäasiantuntijoiden perehtyminen tietoturvaan on jäänyt hyvin vähäiseksi. Kuitenkin tietoturvan laajeneva tarve lisää koko ajan asiantuntijoiden työtä, joten he eivät ehdi osallistua kaikkeen tietoturvakehitykseen. Työtä on siten ostettu ulkopuolisilta asiantuntijoilta. Tutkimuksen yhtenä tarkoituksena oli pohtia yhteistyön kehittämistä eri toimijoiden välillä. Tietoturvan toteuttaminen on monitahoinen ongelma, joten eri sidosryhmien välisen yhteistyön tulisi olla joustavaa. Lisäksi asiakkaan tulisi panostaa omaan osallistumiseensa yhteistyön edistämiseksi. Koska yhteistyötä on tehty vähän satunnaisesti, toimittajilla oli siitä hyvin erilaisia kokemuksia.

Tutkimuksessa tuli esille, että asiakkaan haluttaisiin osallistuvan aktiivisesti tietoturvan määrittelyyn ja kehitystyöhön. Tilanne ei ole oikein kenellekään edullinen, jos asiakas toimittaa tietoturvavaatimukset yleisellä tasolla ja ulkopuolinen asiantuntija osallistuu projektissa kehitystyöhön. Lisäksi auditointi hankitaan ulkopuolisesta yrityksestä. Tietoturva vaikuttuu täysin ulkoistetulta, eikä siinä ole otettu huomioon asiakkaan ympäristön tai toiminnan erityispiirteitä. Pitäisi ehkä miettiä työnjakoa uudelleen. Tulisiko järjestelmien projektipäälliköillä olla edes alkeellinen tietämys tietoturvasta ja tietoturvan kehittämisestä hankkimiinsa järjestelmiin?

Asiakaan tulisi ymmärtää tietoturvan kehittäminen ennen kaikkea monipuolisena yhteistyökuviona. Osaamista tulisi kehittää siten, että tietoturvan toteutuksessa osattaisiin vaatia asiakkaalle tärkeät asiat. Lisäksi asiakkaalla tulisi pystyä hallitsemaan järjestelmäkokonaisuuden tietoturvaa, jolloin yhteistyö ulottuisi eri järjestelmäntoimittajien välille. Aktiivista yhteistyötä tarvitaan, jotta tieto vaihtuisi kaikkien osapuolten kesken ja tietoturvan kehittäminen saataisiin onnistuneesti toteutettua kaikissa ohjelmistotuotteissa.

Ulkopuolisten asiantuntijoiden käytöstä ja auditoinneista oli hyviä kokemuksia. Toimittajat olivat saaneet asiantuntijoilta tietoa hyvistä käytänteistä sekä soveltuvista tietoturvaratkaisuista eri suojaustasoille käyttöön tuleviin ohjelmistoihin. Auditointeihin suhtauduttiin positiivisesti, mutta niiden toteuttaminen ja tulosten analysointi tulisi sopia paremmin, jotta toimittajat saisivat auditoinneista todellista hyötyä tuotteiden kehitykseen.

8 Yhteenveto ja pohdinta

Tutkimuksen tarkoitus oli selvittää tietoturvan kehittämiseen vaikuttavat tekijät vahvaa suojausta vaativissa ohjelmistoissa. Tutkimuksessa selvitettiin, miten asiakkaan ja muiden sidosryhmien tulisi osallistua kehitysprosessiin sekä mitä asiakkaan tulisi huomioida tietoturvavaatimusmäärittelyssä.

Asiakkaan ja toimittajien kehitysprosessimallit edustivat mallien ääripäitä, joten teoriaosuudessa päädyttiin tekemään yleiskatsaus ohjelmistoprosessien kehityksestä. Tämä laajensi näkemyksiä tutkimusaineiston käsittelyssä. Aineistosta saatujen havaintojen perusteella pystyttiin luomaan prosessimalli. Asiakkaan osallistuminen kehitysprojektiin tuli esille prosessimenetelmissä sekä omana kokonaisuutenaan sidosryhmien oikea-aikaisena yhteistyönä IPPD-mallissa. Säännöllinen yhteistyö eri sidosryhmien välillä korostui kaikissa nykyisin käytössä olevissa ohjelmistokehitysmenetelmissä ja myös tutkimustuloksissa.

Tietoturvaan liittyvää ylätason vaatimusmäärittelyä oli käsitelty aikaisemmissa tutkimuksissa vähän. Tietoturvan kehittäminen nähdään pelkästään ohjelmistosuunnittelijoiden työksi, eikä asiakkaiden oleteta määrittävän tietoturvavaatimuksia. Vaatimusmäärittelyn yhteydessä puhuttiin ainoastaan ei-toiminnallisista vaatimuksista, joihin tietoturvavaatimukset kuuluvat. Tietoturvan merkitys ohjelmistoissa on kuitenkin noussut, ja esimerkiksi standardissa (ISO/IEC 25010: 2011) se on nostettu yhdeksi laatua kuvaavista pääominaisuuksista.

Tutkimus toteutettiin laadullisena tutkimuksena teemahaastattelumenetelmällä. Haastattelu katsottiin sopivaksi menetelmäksi, jotta asiakkaan toimintatapaan saataisiin toimittajien ja auditoijien näkemyksiä. Teemahaastattelu ei ollut aikaisemmin tuttu, joten siihen piti perehtyä kirjallisuuden avulla ennen haastattelujen aloitusta. Haastateltavat toivat esille asioita, joihin he toivoivat asiakkaan aktiivista osallistumista ja työpanosta. Haastatteluissa tuli esille yksittäisten henkilöiden ehdotuksia tai hyviä esimerkkejä yrityksissä käytetystä toimintatavoista. Näitä pyrittiin poimimaan mahdollisimman kattavasti mukaan analyysiin. Yritysten edustajat kertoivat ennen kaikkea omia näkemyksiään tutkimusaiheeseen, joten

niitä ei voi pitää yritysten antamina virallisina kantoina. Tutkimuksessa ei tehty määrällistä tutkimusta esimerkiksi siitä, kuinka usein aiheet toistuivat keskusteluissa.

Tutkimus tehtiin vahvaa suojausta vaativien ohjelmistojärjestelmien tietoturvan kehitystä varten, joten tulokset eivät ole yleistettävissä ohjelmistokehitykseen yleisesti. Kuitenkin tietoturva-asioiden huomioiminen kehitysprosessin alussa asiakkaan kanssa selkeyttäisi varmasti tietoturvan kehitystä missä tahansa ohjelmistoprojektissa.

Kun mietitään vahvaa suojausta vaativia ohjelmistoja julkishallinnossa tai yksityisellä puolella, niiden kehityksessä käytetään varmasti samoja tietoturvamekanismeja, mitä tässä tutkimuksessa on tullut esille. Ohjelmistokehittäjille tuotteen ympäristö tai rajapinnat eivät ole luultavasti entuudestaan tuttuja missään ohjelmistokehityksessä, joten asiakkaan ja muiden sidosryhmien osallistumista kehitysprosessiin varmasti tarvitaan.

Julkisissa hankinnoissa käytössä oleva hankintalaki on Euroopan unionissa yleisesti hyväksytty ohjeistus. Sen määrittelemä tarjouskilpailuperiaate ajaa hankinnat helposti vesiputousmallin kaltaiseen toimintatapaan. Tämä tutkimus auttaa ketterien kehitysmenetelmien soveltamista ohjelmistokehityshankinnoissa yleisesti julkishallinnon organisaatioissa.

Tutkimuksessa tuli esille tietoturvan kokonaisuuden hallinta. Kokonaisuuden hallintaan tulisi olla nimetty järjestelmäarkkitehti ja lisäksi tietoturvan ylläpitäminen tulisi tehdä yhteistyössä ohjelmistotoimittajien kanssa koko järjestelmän elinkaaren ajan. Tutkimuksessa esille tuotu kokonaisuuden hallinta ja säännöllinen yhteistyö olisi hyvä malli mihin tahansa isopien järjestelmäkokonaisuuksien tietoturvan hallintaan ja ylläpitoon. Kokonaisvaltaisella tietoturvan suunnittelulla pystyttäisiin saamaan säästöjä ohjelmistojärjestelmiin niin julkishallinnon kuin yksityiselläkin puolella.

Tutkimus kohdistui etsimään parannusehdotuksia puolustusvoimien toimintaan. Haastatteluihin haluttiin vain toimittajien ja auditoijien näkemyksiä. Erilaisia tuloksia voitaisiin saada, jos haastateltaisiin myös asiakkaan edustajia. Tutkimus haluttiin tehdä yleisesti puolustusvoimien ohjelmistohankintoja koskevaksi. Yksityiskohtaisempia tuloksia olisi varmasti saatu, mikäli tutkimus olisi kohdistunut yksittäiseen puolustushaaraan.

Tutkimustuloksista saatujen kehitysehdotusten toteuttaminen käytännössä on varmasti monen vuoden työ. Olisi kuitenkin mielenkiintoista nähdä, miten ehdotetut toimenpiteet soveltuisivat käytännön työhön. Jatkotutkimuksena olisi kiinnostavaa testata mallin toimivuutta eri kokoisiin ohjelmistokehitysprojekteihin. Koska ketterät menetelmät soveltuvat parhaiten pienempiin kehitysprosesseihin, olisi kiinnostavaa nähdä, mitkä osa-alueet toimisivat laajoissa ohjelmistokehitysprojekteissa.

Puolustusvoimilla on ollut pitkään käytössä hankintaprosessi, jota käytetään yleisesti kaikkiin hankintoihin. Kiinnostavaa olisi tutkia, pystyttäisiinkö kehitysprosessia muuttamalla saamaan vaikutuksia muuhunkin kuin tietoturvan parantamiseen. Ketterät kehitysmenetelmät on suunniteltu parantamaan ennen kaikkea ohjelmistokehityksen tuottavuutta ja joustavuutta. Sen perusteella ohjelmistokehityksessä voitaisiin vähentää kustannuksia, lyhentää kehitykseen käytettävää aikaa sekä toteuttaa paremmin käyttöön soveltuvia ohjelmistoja.

Lähteet

Abrams, M., Veoni, J., & Britton, R. K. (2004). Security in Large System Acquisition. Teoksessa R. Kazman & D. Port (Toim.), *COTS-Based Software Systems: Third International Conference, ICCBSS 2004, Redondo Beach, CA, USA, February 1-4, 2004. Proceedings* (ss. 18–30). Berlin, Heidelberg: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-24645-9_14

Abran, A., Khelifi, A., Suryn, W., & Seffah, A. (2003). Usability Meanings and Interpretations in ISO Standards. *Software Quality Journal*, *11*(4), 325–338. http://doi.org/10.1023/A:1025869312943

Agile Alliance. (2001). Agile Manifesto. Noudettu 12. maaliskuuta 2017, osoitteesta agilemanifesto.org/

Aubry, C. (2010). *SCRUM, Le guide pratique de la méthode agile la plus populaire*. Paris: Dunod.

Bai, X., Huang, L., & Zhang, H. (2010). On Scoping Stakeholders and Artifacts in Software Process. Teoksessa J. Münch, Y. Yang, & W. Schäfer (Toim.), *New Modeling Concepts for Today's Software Processes: International Conference on Software Process, ICSP 2010, Paderborn, Germany, July 8-9, 2010. Proceedings* (ss. 39–51). Berlin, Heidelberg: Springer Berlin Heidelberg.

Baskerville, R. (1988). *Designing Information Systems Security*. New York, NY: John Wiley & Sons, Inc.

Baskerville, R. (1993). Information systems security design methods: implications for information systems development. *ACM Computing Surveys*, *25*(4), 375–414. http://doi.org/10.1145/162124.162127

Berenbach, B., Paulish, D. J., Kazmeier, J., & Rudorfer, A. (2009). *Software & Systems Requirements Engineering : In Practice*. New York: McGraw-Hill.

Boehm, B., Bose, P., Horowitz, E., & Lee, M. J. (1995). Software requirements negotiation and renegotiation aids. Teoksessa *Proceedings of the 17th international conference on Software engineering - ICSE '95* (ss. 243–253). New York, NY: ACM Press. http://doi.org/10.1145/225014.225037

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *21*(5), 61–72. http://doi.org/10.1109/2.59

Cohn, M. (2004). *User stories applied : for agile software development*. Boston, MA: Addison-Wesley.

Cysneiros, L. M., do Prado Leite, J. C. S., & de Melo Sabat Neto, J. (2001). A Framework for Integrating Non-Functional Requirements into Conceptual Models. *Requirements Engineering*, *6*(2), 97–115. http://doi.org/10.1007/s007660170008

DeGrace, P., & Stahl, L. H. (1990). *Wicked problems, righteous solutions: A catalogue of modern software engineering paradigms*. Englewood Cliffs, NJ: Yourdon Press.

Deuff, D., & Cosquer, M. (2013). *User-Centered Agile Method*. Hoboken, NJ: John Wiley and Sons Inc.

Dhillon, D. (2011). Developer-Driven Threat Modeling: Lessons Learned in the Trenches. *IEEE Security & Privacy Magazine*, *9*(4), 41–47. http://doi.org/10.1109/MSP.2011.47

Eskola, J., & Suoranta, J. (1998). *Johdatus laadulliseen tutkimukseen*. Tampere: Vastapaino.

Eskola, J., & Suoranta, J. (2008). *Johdatus laadulliseen tutkimukseen*. Tampere: Vastapaino.

Firesmith, D. G. (2003). Security use cases. *Journal of Object Technology*, *2*(3), 53–64. http://doi.org/10.5381/jot.2003.2.3.c6

Gamble, D., & Hemenway, J. (1995). The Use of Generic Architectures in System Integration. Teoksessa *Proceedings 18th National Information Systems Security Conference* (ss. 431–445).

Giorgini, P., Massacci, F., & Mylopoulos, J. (2003). Requirement Engineering Meets Security: A Case Study on Modelling Secure Electronic Transactions by VISA and Mastercard. Teoksessa I.-Y. Song, S. W. Liddle, T.-W. Ling, & P. Scheuermann (Toim.), *Conceptual Modeling - ER 2003: 22nd International Conference on Conceptual Modeling, Chicago, IL, USA, October 13-16, 2003. Proceedings* (ss. 263–276). Berlin, Heidelberg: Springer Berlin Heidelberg.

Gordon, L. A., Loeb, M. P., Lucyshyn, W., & Richardson, R. (2005). *CSI/FBI Computer Crime and Security Survey*. Computer Security Institute.

Gross, J. M., & McInnis, K. R. (2003). *Kanban made simple : demystifying and applying Toyota's legendary manufacturing process*. New York, NY: Amacom.

Heiskanen, A., & VM-julkaisutiimi (Toim.). (2014). *Tietoturvallisuuden arviointiohje*. Helsinki: Juvenes Print - Suomen Yliopistopaino Oy.

Hilve, A., Rousku, K., & Hummelholm, A. (2016). Valtionhallinnon tieto- ja kyberturvallisuuden ohjausryhmän (VAHTI) ohjesivusto. Noudettu 19. toukokuuta 2017, osoitteesta www.vahtiohje.fi

Hirsjärvi, S., & Hurme, H. (2008). *Tutkimushaastattelu : teemahaastattelun teoria ja käytäntö*. Helsinki: Gaudeamus Helsinki University Press.

Hirsjärvi, S., Remes, P., & Sajavaara, P. (2004). *Tutki ja kirjoita* (10. p.). Helsinki: Tammi.

Homan, R. (1991). *The ethics of social research*. London: Longman.

Houston, D. X., & Rosemergy, S. W. (2016). Assessing Product Development Agility. Teoksessa M. Kuhrmann, J. Münch, I. Richardson, A. Rausch, & H. Zhang (Toim.), *Managing Software Process Evolution: Traditional, Agile and Beyond -- How to Handle Process Change* (ss. 39–60). Cham: Springer International Publishing. http://doi.org/10.1007/978-3-319-31545-4_3

Hull, E., Jackson, K., & Dick, J. (2011). *Requirements Engineering* (Third). London: Springer London. http://doi.org/10.1007/978-1-84996-405-0

Ilieva, S., Ivanov, P., & Stefanova, E. (2004). Analyses of an agile methodology implementation. *Proceedings. 30th Euromicro Conference, 2004.*, 1–8. http://doi.org/10.1109/EURMIC.2004.1333387

Islam, S., Mouratidis, H., & Wagner, S. (2010). Towards a Framework to Elicit and Manage Security and Privacy Requirements from Laws and Regulations. Teoksessa R. Wieringa & A. Persson (Toim.), *Requirements Engineering: Foundation for Software Quality: 16th International Working Conference* (ss. 255–261). Berlin, Heidelberg: Springer. http://doi.org/10.1007/978-3-642-14192-8_23

ISO/IEC. (2008). *Systems and software engineering — Software life cycle processes (ISO/IEC 12207:2008)*. Noudettu osoitteesta https://www.iso.org

ISO/IEC. (2011). *Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models (ISO/IEC 25010:2011)*. Noudettu osoitteesta https://www.iso.org

ISO/IEC. (2013). *Information technology — Security techniques — Code of practice for information security controls (ISO/IEC 27002:2013)*. Noudettu osoitteesta https://www.iso.org

ISO/IEC/IEEE. (2011). *Systems and software engineering — Life cycle processes — Requirements engineering (ISO/IEC/IEEE 29148:2011 E)*. Noudettu osoitteesta https://www.iso.org

ISO/IEC/IEEE. (2015). *Systems and software engineering — System life cycle processes (ISO/IEC/IEEE 15288:2015)*. Noudettu osoitteesta https://www.iso.org

ISO/IEC/IEEE. (2016). *Systems and software engineering — Life cycle management — Part 4: Systems engineering planning (ISO/IEC/IEEE 24748-4:2016)*.

Javaid, A. Y., Sun, W., Devabhaktuni, V. K., & Alam, M. (2012). Cyber security threat analysis and modeling of an unmanned aerial vehicle system. Teoksessa *2012 IEEE Conference on Technologies for Homeland Security (HST)* (ss. 585–590). http://doi.org/10.1109/THS.2012.6459914

Jouini, M., Rabai, L. B. A., & Aissa, A. Ben. (2014). Classification of security threats in information systems. *Procedia Computer Science*, *32*, 489–496. http://doi.org/10.1016/j.procs.2014.05.452

Jürjens, J. (2002). Using UMLsec and goal trees for secure systems development. Teoksessa *Proceedings of the 2002 ACM symposium on Applied computing - SAC '02* (ss. 1026–1030). New York, NY: ACM Press. http://doi.org/10.1145/508791.508990

Keith, C. (2010). *Agile Game Development with Scrum*. Boston, MA: Pearson Education.

Kotonya, G., & Sommerville, I. (1998). *Requirements engineering*. Chichester: Wiley.

Kvale, S. (1996). *InterViews: An Introduction to Qualitative Research Interviewing*. Thousand Oaks, CA: SAGE Publications.

Lane, J. A., Boehm, B., Bolas, M., Madni, A., & Turner, R. (2010). Critical Success Factors for Rapid, Innovative Solutions. Teoksessa J. Münch, Y. Yang, & W. Schäfer (Toim.), *New Modeling Concepts for Today's Software Processes: International Conference on Software Process* (ss. 52–61). Berlin, Heidelberg: Springer. http://doi.org/10.1007/978-3-642-14347-2_6

Lema, R. (2010). Adoption of Open Business Models in the West and Innovation in India's Software Industry. *IDS Research Reports*, *2010*(62), 1–144. http://doi.org/10.1111/j.2040-0217.2010.00062_2.x

Liker, J. K., & Hoseus, M. (2008). *Toyota culture : the heart and soul of the Toyota way*. New York, NY: McGraw-Hill.

Linger, R. C. (1994). Cleanroom process model. *IEEE Software*, *11*(2), 50–58. http://doi.org/10.1109/52.268956

Loch, K. D., Carr, H. H., & Warkentin, M. (1992). Threats to Information Systems: Today' s Reality, Yesterday' s Understanding. *MIS Quarterly*, *16*(2), 173–186.

Lodderstedt, T., Basin, D., & Doser, J. (2002). SecureUML: A UML-Based Modeling Language for Model-Driven Security. Teoksessa J.-M. Jézéquel, H. Hussmann, & S. Cook (Toim.), *In: Jézéquel JM., Hussmann H., Cook S. (eds) ≪UML≫ 2002 — The Unified Modeling Language* (ss. 426–441). Springer Berlin Heidelberg.

Mahmood, M. A., Siponen, M., Straub, D., Rao, H. R., & Raghu, T. S. (2010). MIS Quarterly. *Mis*, *34*(3), 431–433. http://doi.org/130.234.244.120

Matulevicius, R., Mayer, N., & Heymans, P. (2008). Alignment of Misuse Cases with Security Risk Management. Teoksessa *Third International Conference on Availability, Reliability and Security* (ss. 1397–1404). IEEE. http://doi.org/10.1109/ARES.2008.88

McDermott, J., & Fox, C. (1999). Using abuse case models for security requirements analysis. Teoksessa *Computer Security Applications Conference, 1999. (ACSAC '99) Proceedings. 15th Annual* (ss. 55–64). http://doi.org/10.1109/CSAC.1999.816013

Mills, H. D., Dyer, M., & Linger, R. C. (1987). Cleanroom Software Engineering. *IEEE Software*, *4*(5), 19–25.

Mokos, K., Meditskos, G., Katsaros, P., Bassiliades, N., & Vasiliades, V. (2010). Ontology-Based Model Driven Engineering for Safety Verification. Teoksessa *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications* (ss. 47–54). IEEE. http://doi.org/10.1109/SEAA.2010.60

Mouratidis, H. (2006). Analysing Security Requirements of Information Systems using Tropos. Teoksessa *Proceedings 1st Annual Conference on Advances in Computing and Technology (AC&T)* (ss. 55 – 64). Noudettu osoitteesta http://roar.uel.ac.uk/409/

Myagmar, S., Lee, A. J., & Yurcik, W. (2005). Threat Modeling as a Basis for Security Requirements. Teoksessa *Symposium on requirements engineering for information security (SREIS)* (ss. 1–8).

Poppendieck, M., & Poppendieck, T. (2009). *Lean software development : an agile toolkit* (14. p.). Upper Saddle River, NJ: Addison-Wesley: Pearson Education.

Poppendieck, M., & Poppendieck, T. D. (2003). *Lean software development : An agile toolkit*. Boston, MA: Addison-Wesley.

Puolustusministeriö. (2015). Katakri 2015 Tietoturvallisuuden auditointityökalu viranomaisille. (T. Takala, Toim.). Helsinki.

Raggad, B. G. (2010). *Information Security Management : Concepts and Practice.* Boca Raton, Florida: CRC Press.

Royce, W. W. (1970). Managing the Development of Large Software Systems. Teoksessa *Proceedings of the 9th international conference on Software Engineering* (ss. 328–338). IEEE Computer Society.

Rubin, K. S. (2012). *Essential Scrum : a practical guide to the most popular agile process*. Ann Arbor, MI: Addison-Wesley.

Ruparelia, N. B. (2010). Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes*, *35*(3), 8. http://doi.org/10.1145/1764810.1764814

Saaranen-Kauppinen, A., & Puusniekka, A. (2006). KvaliMOTV - Menetelmäopetuksen tietovarasto. Noudettu 30. maaliskuuta 2017, osoitteesta www.fsd.uta.fi/menetelmaopetus

Schmidt, R. (2013). *Software Engineering : Architecture-driven Software Development.* Waltham, Mass.: Morgan Kaufmann.

Schwaber, K., & Sutherland, J. (2012). *Software in 30 Days*. Hoboken, NJ: John Wiley & Sons, Incorporated.

Schwaber, K., & Sutherland, J. (2016). Scrum Guide | Scrum Guides. Noudettu 13. maaliskuuta 2017, osoitteesta http://www.scrumguides.org/scrum-guide.html

Seale, C., Gobo, G., Gubrium, J., & Silverman, D. (2004). *Qualitative Research Practice*. London: Sage Puplication Ltd. http://doi.org/10.4135/9781848608191

SEBoK authors. (2016). Introduction to Systems Engineering - SEBoK. Noudettu 19. toukokuuta 2017, osoitteesta http://sebokwiki.org/wiki/Introduction_to_Systems_Engineering

Selby, R. W., Basili, V. R., & Baker, F. T. (1987). Cleanroom Software Development: An Empirical Evaluation. *IEEE Transactions on Software Engineering*, *SE-13*(9), 1027–1037. http://doi.org/10.1109/TSE.1987.233525

Shostack, A. (2014). *Threat modeling : designing for security*. Indianapolis: John Wiley & Sons.

Sindre, G., & Opdahl, A. L. (2005). Eliciting security requirements with misuse cases. *Requirements Engineering*, *10*(1), 34–44. http://doi.org/10.1007/s00766-004-0194-4

Sommerville, I. (1996). Software process models. *ACM Computing Surveys, 28*(1), 269–271. http://doi.org/10.1145/234313.234420

Sommerville, I., & Sawyer, P. (1997). *Requirements Engineering: A Good Practice Guide*. Chichester: Wiley.

Souag, A., Mazo, R., Salinesi, C., & Comyn-Wattiau, I. (2016). Reusable knowledge in security requirements engineering: a systematic mapping study. *Requirements Engineering*, *21*(2), 251–283. http://doi.org/10.1007/s00766-015-0220-8

Spears, J. L., & Barki, H. (2010). User Participation in Information Systems Security Risk Management. *Mis Quarterly*, *34*(3), 503–522.

Stoneburner, G., Goguen, A., & Feringa, A. (2002). Risk Management Guide for Information Technology Systems. *National Institute of Standards and Technology, Special Publication*, *800–30*, 55. http://doi.org/10.1111/j.1745-6622.2008.00202.x

Sutherland, J. (2004). Agile development: Lessons learned from the first scrum. *Cutter Agile Project Management Advisory Service: Executive Update*, *5*(20), 1–4.

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Bus. Rev*, *64*(1), 137–146.

Torr, P. (2005). Demystifying the threat-modeling process. *IEEE Security & Privacy*, *3*(5), 66–70. http://doi.org/10.1109/MSP.2005.119

Tran, H., Holmes, T., Zdun, U., & Dustdar, S. (2011). *Relating Software Requirements and Architectures*. (P. Avgeriou, J. Grundy, J. G. Hall, P. Lago, & I. Mistrík, Toim.), *Relating Software Requirements and Architectures*. Berlin, Heidelberg: Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-642-21001-3

Töttö, P. (2004). *Syvällistä ja pinnallista : teoria, empiria ja kausaalisuus sosiaalitutkimuksessa*. Tampere: Vastapaino.

van Lamsweerde, A. (2004). Elaborating Security Requirements by Construction of Intentional Anti-Models. Teoksessa *Proceedings of the 26th International Conference on Software Engineering (ICSE'04* (ss. 148–157). Washington, DC: IEEE Computer Society.

Venkatesh, V., & Bala, H. (2008). Technology Acceptance Model 3 and a Research Agenda on Interventions. *Decision Sciences*, *39*(2), 273–315. http://doi.org/10.1111/j.1540-5915.2008.00192.x

Wang, H., & Wang, C. (2003). Taxonomy of Security Considerations and Software Quality. *Communications of the ACM*, *46*(6), 75–78.

Wang, J., Gupta, M., & Rao, H. R. (2015). Insider Threats in a Financial Institution: Analysis of Attack-Proneness of Information Systems Applications. *Mis Quarterly*, *39*(1), 91–112.

Wiegers, K. E., & Beatty, J. (2013). *Software requirements*. Redmond, WA: Microsoft Press.

Yu, Y., Franqueira, V. N. L., Than Tun, T., Wieringa, R. J., & Nuseibeh, B. (2015). Automated analysis of security requirements through risk-based argumentation. *Journal of Systems and Software*, *106*, 102–116. http://doi.org/10.1016/j.jss.2015.04.065

Liitteet

A Käsittelyvaatimuksia osoittavat suojaustasot

Valtioneuvoston asetus tietoturvallisuudesta valtionhallinnossa 1.7.2010/681, 3. luku Asiakirjojen luokittelu 9 §. Salassa pidettävien asiakirjojen luokittelussa käytetään seuraavia luokkia:

1) suojaustaso I, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa erityisen suurta vahinkoa salassapitosäännöksessä tarkoitetulle yleiselle edulle;

2) suojaustaso II, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa merkittävää vahinkoa salassapitosäännöksessä tarkoitetulle yleiselle edulle;

3) suojaustaso III, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa vahinkoa salassapitosäännöksessä tarkoitetulle yleiselle tai yksityiselle edulle;

4) suojaustaso IV, jos asiakirjaan sisältyvän salassa pidettävän tiedon oikeudeton paljastuminen tai oikeudeton käyttö voi aiheuttaa haittaa salassapitosäännöksessä tarkoitetulle yleiselle tai yksityiselle edulle.

Edellä 1 momentin 4 kohdassa tarkoitettuun suojaustasoon kuuluvaksi voidaan luokitella muukin kuin salassa pidettäväksi säädetty asiakirja, jos asiakirjan luovuttaminen on lain mukaan viranomaisen harkinnassa tai asiakirjaan sisältyviä tietoja saa lain mukaan käyttää tai luovuttaa vain määrättyyn tarkoitukseen ja jos tiedon oikeudeton paljastuminen voi aiheuttaa haittaa yleiselle tai yksityiselle edulle tai heikentää viranomaisen toimintaedellytyksiä.

B Haastattelupyyntö

Opiskelen parhaillaan Jyväskylän yliopistossa Informaatioturvallisuutta ja teen pro gradu-tutkimusta. Tutkimus tulee Puolustusvoimien Järjestelmäkeskukseen, jossa itsekin työskentelen. Pro gradutyön aiheena on puolustusvoimille hankittavien järjestelmien tietoturvavaatimusten kehittäminen. Tutkimuksen tarkoituksena on selvittää tietoturvavaatimuksiin liittyviä toteutushaasteita ja toisaalta kehitysideoita. Tutkimus toteutetaan haastattelututkimuksena puolustusvoimien yhteistyöyrityksiin. Haastattelussa käsitellään vapaamuotoisesti seuraavat kysymykset:

- Mitä mieltä olet puolustusvoimien tietoturvavaatimusten ymmärrettävyydestä?
- Voidaanko tietoturvaan tarvittava työ hinnoitella vaatimusten perusteella?
- Kun suunnittelette tietoturvavaatimusten toteutusta, millaisia ongelmia tulee esille?
- Miten tietoturvavaatimukset yleensä sopivat järjestelmän suunnitteluprosessiin?
- Mielipiteitä Katakrin ja puolustusvoimissa käytettävien standardien käytettävyydestä?
- Miten olette kokeneet ulkopuolisten tietoturva-asiantuntijoiden käytön järjestelmien kehitystyössä?
- Jos saisit tehtäväksi laatia järjestelmän tietoturvavaatimukset, mitä asioita ottaisit huomioon?
- Saavutetaanko tietoturva pelkillä tietoturvavaatimuksilla, vai tulisiko toimintaa kehittää jollain muulla osa-alueella?

Haastatteluun kannattaa varata aikaa noin 1 tunti. Haastattelu äänitetään, jotta asioiden kirjaamiseen ei kuluisi haastattelussa aikaa. Haastattelun tietoturvataso on lähtökohtaisesti julkinen. Mikäli haastattelussa tulee esille asioita, jotka ovat yritykselle luottamuksellista tietoa, niistä tulee ilmoittaa haastattelussa tai haastattelun jälkeen. Yrityksen kanssa sitten sovitaan, muokataanko tietoa vai poistetaanko se kokonaan haastattelumateriaalista.

Voisinko tulla tekemään haastattelun teidän yritykseen henkilölle, joka työskentelee tämän aihealueen parissa?

Ystävällisin terveisin,

Tarja Lauhikari

C Koodiluettelo ja koodien toistuvuus

Koodi	X1	X5	X9	X10	X6, X7	X4	X3	X2	X8	Yhteensä:
Kehitysprosessi>Auditoinnit	0	10	1	7	5	3	7	18	3	54
Kehitysprosessi>Auditointi asiantuntijan nä	0	0	0	0	0	0	0	0	5	5
Kehitysprosessi>Iteratiivisuus, kettärä kehit	3	10	2	9	2	9	0	9	1	45
Kehitysprosessi>Muutokset vaatimuksissa	0	0	1	0	0	0	6	1	0	8
Kehitysprosessi>Testaus ja todentaminen	0	1	0	1	6	3	2	3	0	16
Kehitysprosessi>Tietoturva ja elinkaari	3	0	0	1	0	1	0	2	1	8
Kehitysprosessi>Tietoturvatyön aloitus	3	5	3	3	3	4	3	1	7	32
Tituvaatimukset>Integraatio ja tietoturva	7	0	0	0	1	2	8	0	2	20
Tituvaatimukset>Katakri ja vaatimusmäärit	0	7	6	1	1	8	1	7	3	34
Tituvaatimukset>Kokemukset PV:n tietotur	0	5	1	4	10	2	5	2	0	29
Tituvaatimukset>Kokonaisvaltaisesti tehty t	5	0	0	2	0	0	0	6	5	18
Tituvaatimukset>käytettävyys ja tietoturva	6	5	0	2	1	7	5	2	0	28
Tituvaatimukset>Käyttöympäristö ja tietotu	0	0	2	1	2	2	6	2	3	18
Tituvaatimukset>Perusteet vaatimuksille	0	2	0	0	0	2	2	2	0	8
Tituvaatimukset>Suojaustason määritys	0	5	3	5	6	0	2	1	2	24
Tituvaatimukset>Toiminnalliset vaatimukse	1	0	0	0	2	4	0	2	1	10
Tituvaatimukset>Uhka-analyysi	0	6	2	2	1	4	0	2	1	18
Tituvaatimukset>Vaatimuksissa tulisi ottaa	1	0	4	2	1	0	2	1	2	13
Tituvaatimukset>Vaatimusten tarkkuudesta	1	4	2	0	2	1	5	2	5	22
Tituvaatimukset>Vahti-ohjeet	0	0	2	0	3	3	2	0	0	10
Yhteistyö>Asiakkaan osallistuminen	4	2	1	2	3	1	2	3	0	18
Yhteistyö>Asiantuntija-apu	2	5	3	8	5	9	6	4	1	43
Yhteistyö >Kehitysideoita	0	3	0	1	2	1	6	6	3	22
Yhteistyö >Koulutus	3	0	6	3	10	1	5	1	2	31
Yhteistyö >Raportit	3	1	0	0	0	3	2	0	0	9
Yhteistyö >Tuotekehitystilat	0	0	0	0	7	0	1	0	0	8
Yhteensä:	42	71	39	54	73	70	78	77	47	551