Haaga-Helia
ammattikorkeakoulu Oy

Haasteet ICT-ratkaisun kehityksessä sairaalaympäristöön

Jani Maaranen

Opinnäytetyö
Tietojenkäsittely
2020

Tiivistelmä

Haaga-Helia ammattikorkeakoulu Oy

Tekijä(t)
Jani Maaranen

Koulutusohjelma
Tietojenkäsittely

Raportin/Opinnäytetyön nimi
Haasteet ICT-ratkaisun kehityksessä sairaalaympäristöön

Sivu- ja liitesivumäärä
45 + 3

Suomen terveydenhuolto on tunnetusti maailman huippuluokkaa, mutta SOTE-uudistus ja ikääntyvät ikäluokat lisäävät kysyntää terveydenhuollolle ja herättävät ihmisissä kysymyksiä, miten samaa hyvää terveydenhuoltoa voidaan jatkossakin toteuttaa? Henkilökuntaa ja osaamista vaaditaan entistä enemmän, mutta samalla pitäisi säästää terveydenhuollosta aiheutuvista kustannuksista.

Opinnäytetyö käsittelee potilastietojärjestelmien ja muiden sairaalan ICT-ratkaisujen kehitysprossia sovelluskehittäjän näkökulmasta. Se esittelee lukijalleen nykypäivän sairaalojen ICT-ratkaisuja, niiden toimintoja, osuutta potilaan hoidossa ja miten ne tehostavat sairaalan toimintaa. Työssä käydään läpi sairaalajärjestelmiin liittyvä lainsäädäntö ymmärrettävästi ja miten se tulee ottaa huomioon kehitysvaiheessa. Lisäksi se esittelee sovelluskehitysprosessin, eli kuinka sovellus tai palvelu tulisi toteuttaa nimenomaan sairaalaan onnistuneesti.

Sosiaali- ja terveydenhoitoalalla käytetään paljon potilas- ja asiakasdataa. Sähköisten asiakastietojen käsittelylle on asetettu lukuisia yhteen toimivuudelle, tietoturvalle, tietosuojalle ja toiminnallisuudelle säädettyjä vaatimuksia, jotka on määritelty Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä (159/2007). Lain toteutumista Suomessa valvoo Valvira. Esimerkkejä tällaisista asiakas- ja potilastietojärjestelmistä on Apotti ja Centricity High Acuity Critical Care.

ICT-ratkaisuilla pyritään ratkaisemaan terveyspalveluiden saatavuuteen, laatuun ja kustannuksiin liittyviä haasteita, joihin liittyvät paineet kasvavat jatkuvasti. Sairaalat koostuvat lukuisista erilaisista osastoista ja eri hoitoalojen osa-alueista, joiden kaikkien toimintatavat eroavat toisistaan. Kehitettävän ratkaisun on sulauduttava erinomaisesti sairaalan omiin toimintamalleihin ja tuotava konkreettista hyötyä käyttäjilleen. Ratkaisujen halutaan olevan monipuolisia, tehokkaita, ratkaisevan useampia ongelmia kerralla ja olla räätälöitävissä sairaalojen omiin tarpeisiin.

Itse sairaalasovelluksen kehitysprosessissa käytettävät peruskehitysmallit ja metodit eivät eroa juurikaan perinteisen sovelluksen kehityksestä. Sairaalaan kehitettäessä kuitenkin tulee ottaa huomioon lukuisia asioita tiukempaan lainsäädäntöön, potilasturvallisuuteen, liitettävyyteen, räätälöintiin, validointiin, testaukseen ja toiminnan jatkuvuuteen liittyen. Hyviä perusperiaatteita, asiakkaan vaatimusten ja lainsäädännön asettamia kriteerejä noudattamalla, sekä käyttäjien erityistarpeet huomioimalla voidaan kehittää onnistunut ICT-ratkaisu sairaalaan.

Asiasanat
Asiakastietojärjestelmä, potilastietojärjestelmä, sovelluskehitys, ICT

Sisällys

1 Johdanto

1.1 Tausta

Suomalainen ilmainen terveydenhuolto on tunnetusti maailman huippuluokkaa. SOTE-uudistuksen aikana sekä suurten ikäluokkien ikääntyessä ihmisille herää kysymys, miten vastaavat laadukkaat terveyden- ja hyvinvoinnin palvelut taataan myös tuleville sukupolville? Henkilökuntaa ja osaamista vaaditaan yhä enemmän ja enemmän, mutta samanaikaisesti kustannuksissa pyritään säästämään lisääntyvän hoitopaineen takia. (Junko 2018)

Esimerkkejä uusista ja innovatiivisista ratkaisuista terveydenhoidossa ilmaantuu koko ajan, ja jatkuvasti digitalisoituva yhteiskunta ja kehittyvä teknologia tarjoavat hyvinkin monipuolisia ratkaisuja tulevaisuuden haasteisiin. Uusien terveysteknologia innovaatioiden avulla voitaisiin esimerkiksi pidentää vanhusten mahdollisuutta viettää eläkepäiviään omassa kodissaan erilaisien älykoti ratkaisujen avulla. Oman terveyden tarkkailu itsenäisesti tehokkaammin mahdollisestaan lukuisilla mobiilisovelluksilla ja kehon varaosien tulostus -mahdollisuus on nähtävissä tulevaisuudessa 3D-tulostuksen avulla. Suomen yliopistollisten sairaalojen Virtuaalisairaala 2.0 hankkeen tuloksena uusia terveydenhuollon helpotuksia on nähtävissä muun muassa Terveyskylä.fi verkkopalvelun synty. Se on palvelu, johon on luotu tietopankki erinäisistä sairauksista, oppaita, omahoito-ohjeita ja digihoitopolkuja, joilla pyritään täydentämään perinteistä sairaalahoitoa. (Junko 2018)

Perinteisesti terveydenhuolto on kuitenkin keskittynyt sairaaloihin, ja niissäkin uusia hoitojärjestelmiä uusitaan ja kehitetään jatkuvasti. Sairaalat ovat valtavia laitoksia, joissa työskentelee lukuisia eri sidosryhmiä, erilaisissa terveydenhuollon tehtävissä, kuten tehohoidossa, nukutus-, leikkaus- ja röntgentoiminnassa. Sairaalat hyödyntävät potilaan hoidossa paljon erilaisia lääkinnällisiä laitteita ja järjestelmiä, joiden sujuva integroituminen yhteen tehostaa sairaalan työntekoa, ennakointia, päätöksentekoa ja hallintaa. Tällaista sairaalan perustoimintaa tehostetaan potilastietojärjestelmillä, jotka ovat sähköisiä järjestelmiä, jotka sisältävät potilaan ja potilaan hoitoon liittyvää dataa. Potilastietojärjestelmillä saadaan sairaalalle ajallisia säästöjä, tehostetaan hoidon laatua ja helpotetaan koko sairaalan kokonaisvaltaisempaa hallintaa.

Jotta sairaalassa voitaisiin ottaa käyttöön uusia ICT-ratkaisuja, on näiden noudatettava lukuisia säädöksiä, lakeja ja vaatimuksia, jotta käytettävä tuote voidaan todeta potilasturvalliseksi. Sovellusratkaisuja tuottavan yhtiön on kyettävä osoittamaan tuotteensa luotettavuus ja tietoturva, ennen kuin sitä voidaan harkita käytettäväksi potilashoidossa. Jotta so-

vellus tai ICT-ratkaisu saataisiin sairaalakäyttöön, on kehittäjän hyvä noudattaa hyviä perusperiaatteita, sovelluksen kehityksen koko elinkaaren läpi ja suhteutettava sovelluksensa tukemaan sairaalan toimintaa, siten että tästä on käyttäjälleen konkreettista hyötyä.

1.2 Tavoitteet ja rajaukset

Pyrin kartoittamaan lukijalle, mitkä ovat sairaalojen haasteet ICT-ratkaisujen kehityksessä sairaalaympäristöihin, ja mikä niiden tilanne on nykypäivänä sovelluskehittäjän näkökulmasta. Tällä pyrin osoittamaan, millaisia ratkaisuja tänä päivänä on käytössä sairaaloissa ja mitä hyötyä ne tuovat sairaalalle tai käyttäjilleen. Työ esittelee sairaalaan toiminnot, joissa erinäisiä teknologisia ratkaisuja on käytössä, mitä niissä tapahtuu ja miten laitteet sekä ICT ovat osana potilaan hoitoa. Tällä on tarkoitus herättää lukijan omaa ajattelua miettimään, kuinka teknologinen nykypäivän sairaala todellisuudessa on ja mahdollisesti keksimään jopa uusia tehokkaampia ratkaisuja esiteltäviin toimintoihin. Työ ottaa huomioon myös lainsäädännön ja säädösten näkökulman ja pyrkii esittelemään nämä lukijalle ymmärrettävästi. Sairaalan, malliesimerkkien ja lainsäädännön lisäksi, työ esittelee sovelluskehityksen -prosessin vaiheita yleisellä tasolla ja antaa sovelluskehittäjälle selkeän ohjenuoran, jota seuraamalla sairaalaan voidaan lähteä kehittämään sovellusta, niin että se on potilasturvallinen ja sopiva käyttötarkoitukseensa kehityksen alusta asti. Työn päämääränä on esitellä lukijalle, millaisia erikoisvaatimuksia erinäisillä sairaalajärjestelmillä ja sovellutuksilla on ja kuinka kehittäjä voi ottaa ne huomioon jo tuotteen kehityskaaren alussa ja millaisia kehitysprosesseja noudattamalla lopullinen tuote voi olla onnistunut. Työ ei käsittele kooditason ratkaisuja, eikä hae konkreettisia vaihtoehtoisia ratkaisuja tai kehitysehdotuksia nykypäivän sairaalaratkaisuille.

1.3 Tutkimusmenetelmät

Opinnäytetyö suoritetaan tutkimuksena. Työn aineisto on pääasiallisesti koottuja artikkeleita, blogeja, tieteellistä teoriaa, dataa yritysten omilta sivuilta, ammattilaisten haastatteluita ja sovelluskehityksen kirjallisuutta. Kokoan vaatimukset luettavaksi kokonaisuudeksi ja arvioin niitä myös oman kokemukseni kautta, jotta lukijalle voidaan tuottaa hyvä todellinen kuva alan nykytilanteesta ja kertoa välineistä, joilla kehittää ICT-ratkaisuja sairaalaympäristöön mahdollisimman oikeaoppisesti.

1.4 Termistö

Audit Trail – työkalu, johon sovelluksessa tehtävät toiminnot, muokkaukset, tallennukset, poistot jne tallentuvat. Audit trail:llä tehostetaan jäljitettävyyttä ja täten tietoturvaa. (Fair-Warning 2020)

BDD -testaus – Behavior-driven development, lähdekoodin testausmalli, jossa sovelluksen toiminnallisuuksia testataan ja käsitellään skenaarioina. (Nest 2020)

ICT – Information and communication technology, eli Suomeksi tieto- ja viestintätekniikka. Ala, joka keskittyy tietotekniikkaan, ohjelmointiin, kyberturvallisuuteen, tietoverkkojen ja tietojärjestelmien ylläpitoon ja hallintaan. (JAMK)

Scrum – Ketterä sovelluskehitysmalli, jossa sovelluskehitysprojekti etenee sprintteinä. (Meteoriitti)

Security by Design – Sovelluskehityksen lähestymistapa tai malli, jossa tietoturvaan liittyvät asiat otetaan huomioon jo tuotteen kehityskaaren alussa. (Reciprocity 2020)

SLA -sopimus – Service Level Agreement, eli palvelutaso -sopimus, jossa määritellään tarkasti asiakkaan ja palveluntarjoajan väliset roolit ja toimitettavan palvelun taso ja tämän tarkkailuun käytettävät mittarit. (Overby 2017)

Unit -testaus – eli yksikkötestaus, jossa lähdekoodia testataan ”yksikköinä”. Tarkoitus testauskella on selvittää, että lähdekoodi toimii todellakin suunnitellusti. (Rouse 2019)

Windows-klusterointi – Ylläpidollinen varotoimi, jossa Windows -palvelin voidaan jatkuvasti kopioida ja liittää klusteriin/joukkoon, jossa yhden klusterin pettäessä, identtinen kopio kyseistä hajonneesta palvelimesta voi edelleen jatkaa toimintaansa. Tällä tuetaan palveluun jatkuvuutta. (Kolu 2016, 27-28)

2 Asiakas- ja potilastietojärjestelmien haastealueet

Monissa Suomen sairaaloissa ja sairaanhoitopiireissä, esimerkiksi Keski-Suomen sairaanhoitopiirissä on haluttu ottaa käyttöön alueellisia prosesseja tukevia tietojärjestelmiä, sillä jokaisen sairaalan ja sairaanhoitopiirin toimintatavat voivat erota toisistaan ja järjestelmän tulee tukea yksittäisen sairaalan omia toimintamalleja. Uusissa järjestelmissä halutaan ottaa huomioon sekä asiakkaan, että heitä hoitavien ammattilaisten ja sidosryhmien tarpeet. Järjestelmien tulee tukea tulevaisuuden terveydenhuollon vaatimuksia: uusia organisaatiotason toimintamalleja, logistiikka, materiaalinhallintoa, tietojärjestelmiä ja ratkaisua, joilla henkilökunnan työaika saadaan kohdennettua tehokkaammin potilas- ja hoitotyöhön. Myös asiakkaan oman hoidon seuranta halutaan tehdä helpoksi sähköisen asioinnin avulla. (Keski-Suomen sairaanhoitopiiri 2016)

Sosiaali- ja terveydenhoitoalalla käsitellään paljon potilas- ja asiakasdataa sähköisissä järjestelmissä. Niissä prosessoidaan asiakas- ja potilastietoja, sekä niiden sisältämiä henkilökohtaisia tietoja. Tällaisten tietojärjestelmien ja ohjelmistojen tulee täyttää yhteistoimivuudelle, tietoturvalle, tietosuojalle ja toiminnallisuudelle asetetut vaatimukset, ennen kuin se voidaan ottaa käyttöön sairaalaympäristössä. Valvira on Suomessa taho, joka valvoo tällaisten asiakas- ja potilastietojen käsittelyyn tarkoitettujen tietojärjestelmien vaatimusten toteutumista. Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä (159/2007) määrittelee lainmukaiset kriteerit tietojärjestelmien vaatimuksille, valmistajille ja terveydenhuollon palvelujen tarjoajille. Valvira on vastuussa vaatimuksenmukaisten tietojärjestelmien rekisterin ylläpidosta ja näihin liittyvien potilasturvallisuutta vaarantavien ilmoitusten ja poikkeamien käsittelystä. Valvira täten ohjaa asiakastietojen käsittelystä annetun lain toteutumista, jonka nojalla Valviralla on oikeus lisäksi suorittaa sen edellyttämiä tarkastuksia. (Valvira 2020) Myös terveyden ja hyvinvoinninlaitos (THL) vaikuttaa välillisesti tietoturvaan, potilasturvallisuuteen ja potilaan hoitoon. THL osallistuu tietojärjestelmien yhdistämiseen ja toisten järjestelmien yhteen toimivuuden tukemiseen. Kanta -palvelu on yksi hankkeista, joiden kehitykseen THL osallistuu aktiivisesti. Kannasta voidaan hakea asiakas- ja potilastietoja, jotta niitä voidaan käyttää yli organisaatiorajojen ja helpottaa täten ammattilaisten toimintaa ja yksittäisen henkilön tiedonsaantia. (THL 2019)

Seuraavat kappaleet käsittelevät esimerkkejä oikeista asiakas- ja potilastietojärjestelmistä, joita sairaaloissa on käytössä tänä päivänä. Sairaalajärjestelmät ovat valittu niiden suosion ja uutisoinnin määrän perusteella Kappaleissa käydään läpi järjestelmien toiminta ja hyödyt perustasolla ja niihin liittyvää uutisointia, mikäli tästä on saatavilla tuoretta aineistoa.

2.1.1 Haasteet Apotissa

Apotti on uusi sosiaali- ja terveydenhuollon ammattilaisten käyttöön tarkoitettu yksi, yhtenäinen asiakas- ja potilastietojärjestelmä, joka mahdollistaa reaaliaikaisen datan käytön palvelu- ja hoitopaikasta riippumatta. Ensimmäisiä Apotti -järjestelmän käyttöönottaneita sairaaloita oli Vantaan HUS:n alueen Peijaksen sairaala, jossa käyttöönotto tapahtui 10.11.2018 (Apotti 2018). Apotti on samalla myös sairaalaan toiminnanohjausjärjestelmä, joka sisältää asiakas- ja potilasprosesseja noudattavia työnkulkuja, joka helpottaa hoitohenkilökunnan työtä. Järjestelmä myös takaa laadun toteutumista muistuttamalla, jos jokin toimenpide on jäänyt välistä. Järjestelmästä saatavan analytiikan ja riskimittareiden ansiosta järjestelmä osaa myös itse tunnistaa mm. sairaalan riskipotilaita. Asiakkailla on käytössä puolestaan asiakasportaali Maisa, jonka avulla asiakas voi helpommin saada yhteyden terveydenhuollon ammattilaiseen ja hoitaa omaa asiaansa sähköisesti ajasta ja paikasta riippumatta. (Apotti)

Apotti -hankkeen tavoitteena on tukea SOTE -uudistusta, jossa sosiaali- ja terveydenhuollon järjestelmän halutaan olla yhtenäinen kaikkialla. Tällä tavoitellaan toiminnan kehitystä, sekä parempaa hoidon laatua. Apotti ei ole kokonaisuutena ainoastaan IT-hanke, vaan toiminnanmuutoshanke, joilla pyritään yhdistämään eri kuntien toimintamalleja samanlaisiksi. Apotti -järjestelmän kerrotaan perustuvan ammattilaisten palvelu- ja hoitotilanteista vuosien varrella kertyneeseen kokemukseen, jolla halutaan aidosti tukea ammattilaisten tekemää hoitotyötä. (Vantaa)

Mikrobitin sivuilla julkaistussa artikkelissa kerrotaan kuinka HUS -sairaanhoitopiiriin kuuluvat työntekijät ovat turhautuneita uuteen Apotti -järjestelmään, joka on noussut huolenaiheeksi etenkin koronapandemian aikana. Ongelmallinen järjestelmä hidastaa sairaalan toimintaa ja aiheuttaa hoitohenkilökunnalle lisähuolia. Osa työntekijöistä kertoo, että tahtoisi palata vanhoihin järjestelmiin, mutta kehittämisjohtaja Visa Honkasen mukaan tämä ei ole mahdollista, Apotin keräämä data tulisi siirtää käsin vanhoihin järjestelmiin, joka olisi todella hidas prosessi. Apotin on raportoitu hidastavan työtä ja vaarantavan potilaiden potilasturvallisuutta muun muassa yksinkertaistamalla prosesseja liikaa, valmiiden vastauspohjien takia, eikä anna käyttäjän kirjata potilaan hoidon kannalta kriittistä dataa käsin esimerkiksi kommenttina, jolloin potilaan oirekuvaukset saattavat jäädä liian suppeiksi. Järjestelmästä ei saa nopealla vilkaisulla käsitystä siitä, kuinka potilas todellisuudessa voi. Järjestelmä on myös perustoiminnaltaan monimutkainen ja viivästyttää tehohoitoa ja leikkaustyötä. Sosiaali- ja terveysalaa valvovalle taholle Valviralle on jätetty 1. helmikuuta – 6. maaliskuuta 2020 välisellä ajalla 48 ilmoitusta liittyen potilastietojärjestelmä Apottiin. Virhetilanteita lääkäreiden mukaan tapahtuu usein lääkkeiden annostelussa, sillä Kanta-lääkejärjestelmä ei ole ajantasainen ja lääkelistojen erinäköisyydestä eri käyttäjärooli -tasoilla

sekä henkilökunnan merkintävirheistä, jotka ovat seurausta järjestelmän monimutkaisuudesta. Helsingin Sanomien haastatteleman lääkärin mukaan ”Virhetilanteita tapahtuu osastolla päivittäin”. (Mikrobitti 2020)

Koska Apotti on myös varsin uusi järjestelmä, kaikkia sen ongelmakohtia ei ole vielä havaittu ja tällä on ollut vakavia seurauksia sairaanhoitopiireissä, joissa se on otettu käyttöön ensimmäisenä. Tämän takia Apotti -järjestelmä on lähiaikoina ollut suuren kohun ja kritisoinnin kohteena. Helsingin ja Uudenmaan sairaanhoitopiirin (HUS) alueella Apotti -järjestelmän aiheuttaman vaaratilanteen takia yksi ihminen on kuollut ja aiheuttanut 54 vaaratilannetta yksinään Peijaksen sairaalassa Vantaalla. (Pikkarainen 2020)

Apotin aiheuttamien vaaratilanteiden seurauksena sairaanhoitopiirien luotto järjestelmään on heikkoa ja eivät halua vaarantaa muiden potilaiden turvallisuutta käyttämällä kehityksensä alussa olevaa järjestelmää. Keusoteen kuuluvat Keski-Uudenmaan kunnat ovat ilmoittaneet haluttomuutensa siirtyä Apotti järjestelmään ja ovat täten kieltäytyneet ottamasta sitä käyttöön alueensa sairaaloissa. Keusoteen kuuluvia kuntia ovat Hyvinkää, Järvenpää, Mäntsälä, Nurmijärvi, Pornainen ja Tuusula. (Nironen 2020) Apotti on myös suuri lovi kuntien budjetissa ja vuoden 2013 väestötietoihin suhteutetut maksuosuudet olisivat seuraavat:

Taulukko 1. Apotin maksuosuudet Keusotessa vuoden 2013 väestötiheyteen suhteutettuna (Niemi 2019)

Kunta	**Hinta**
Hyvinkää	27 milj. euroa
Nurmijärvi	24,5 milj. euroa
Järvenpää	23,5 milj. euroa
Tuusula	20 milj. euroa
Mäntsälä	12 milj. euroa
Pornainen	3 milj. euroa

(Niemi 2019)

Suomen suurin sairaanhoitopiiri HUS siirtyi 30 – 31.10.2020 välisenä yönä potilastietojärjestelmä Apottiin. Uudesta järjestelmästä on tehty lukuisia läheltä piti -tilanteita ja kanteluita, mutta näiden määrän kerrotaan romahtaneen jo maaliskuussa. Muun muassa ongel-

mat lääkkeiden annostelussa ovat huomattavasti vähentyneet sitten alkuvuoden. Ongelmia kuitenkin edelleen esiintyy esimerkiksi sairaalasta poistuvien potilaiden kohdalla, koska Apotin ja Kansallisen reseptikeskuksen lääketietojen kirjaustavat eroavat merkittävästi toisistaan. Myös Valviralle tehtyjen ilmoitusten määrä on huomattavasti vähentynyt. Maaliskuun 7. – Lokakuun 29. välisenä aikana ilmoituksia vaaratilanteista on jätetty enää 35. Apottiin ei kuitenkaan edelleenkään olla lähellekään tyytyväisiä. Finnish Journal of eHealth and eWelfare -lehdessä julkaistussa artikkelissa, neljätuhatta sairaanhoitajaa, jotka käyttävät potilastietojärjestelmiä työssään, arvioivat kuinka hyvin erilaiset järjestelmät tukevat heidän työtään. Apotti arvioitiin koko arvostelussa kouluarvosanoin huonoimmaksi kaikista (5,6). Korkeimman arvosanan sai Esko -järjestelmä (8,20). Apotti sai positiivista palautetta teknisestä toimivuudesta, joka tarkoitti vakautta ja sen kykyä reagoida komentoihin. Kritisointia se sai tiedonkulkuun ja yhteistyöhön, sekä hoidon laatuun liittyen. Jopa 48 prosenttia Apottia arvostelleista hoitajista arvioivat, että sen virheellinen toiminta on aiheuttanut vakavan haittatapahtuman potilaalle. Apotti -järjestelmään on tehty satoja muutoksia edellisestä käyttöönotosta alkaen ja HUS:n omien lukujen mukaan Apotti-sairaaloiden potilastyytyväisyys on kasvanut. HUS kuitenkin arvioi, että ensimmäinen käyttöönottokuukauden ajan (Marraskuu 2020), HUS kykenee hoitamaan hieman vähemmän potilaita kuin aikaisemmin järjestelmän aiheuttamien haasteiden takia. Normaali tahti pitäisi kuitenkin jatkua ensimmäisen kuukauden jälkeen. Apotti tulee kuitenkin jatkossa yhtenäistämään eri sairaaloiden toimintatapoja ja datan kulkua samanlaisiksi ja täten takaa, että potilas saa samanlaista korkealaatuista hoitoa kaikissa sairaanhoitopiireissä, joissa Apotti on käytössä. (Aalto 2020)

2.1.2 Tehohoidon valvontajärjestelmä (Centricity High Acuity Critical Care)

Tehohoidossa hoidetaan potilaita, jotka sairauden tai onnettomuuden seurauksena ovat saaneet hengenvaarallisia vammoja tai muuten vaativat erityistarkkaa valvontaa terveydentilansa takia. Tehohoidossa olevien potilaiden vammat on arvioitu ohimeneviksi ja hoidolla pyritään ehkäisemään, diagnosoimaan ja hoitamaan kyseisiä vammoja. Tehohoito hyödyntää lukuisia vaativia valvonta- ja hoitomenetelmiä, joissa hyödynnetään monialaista ja osaavaa ammatillista erityisosaamista ja erilaisia kehittyneitä teknologioita, joiden takia tehohoidon kustannukset ovat myös korkeat. (Lääkäriliitto)

Tehohoidon yksiköiden kustannukset ja potilaskuolleisuus ovat suurimpia koko terveydenhuollon alalla ja voivat kattaa jopa 20 % koko sairaalaan kokonaiskustannuksista. Tästä syystä tehohoidon valvontaan liittyy erityisen suuri paine, minkä tarkoituksena on vähentää kokonaishoitokustannuksia. Centricity High Acuity Critical Care (CHACC) on tehoosastoille tarkoitettu järjestelmä, joka pyrkii tarjoamaan ratkaisun kustannustehokkaaseen

tehohoitoon. Se on yhtenäinen järjestelmä, joka tukee sairaalan toimintaa tarjoamalla selkeän aikataulun suunnittelun, reaaliaikaisen datansiirron ja datan tuonnin lääkinnällisiltä laitteilta suoraan sovellukseen, sekä sen esittämisen helposti tulkittavana tietona käyttäjälle. Lisäksi se tunnistaa riskipotilaat, jotta tarvittava apu voidaan priorisoida sitä nopeiten tarvitseville. CHACC mahdollistaa myös potilaan lääketilaukset ja potilaan hallinnan hoitolaitoksessa, sekä analytiikan tuottamisen sairaalaan toiminnasta. Esimerkiksi Englannissa, GE Healthcaren CHACC:tä käyttävien asiakkaiden hoidossa olleiden potilaiden aikaa hoitolaitoksessa onnistuttiin vähentämään 19 %, optimoitiin kliinistä aikaa vähintään kymmenellä tunnilla ja säästettiin 0,5 miljoonalla punnalla kuluja automaattisen datan keräyksen ansiosta. (GE Healthcare 2019)

Seuraavissa kappaleissa käydään läpi haasteita sairaalan toiminnoissa, joissa käytetään lukuisia lääkinnällisiä laitteita ja järjestelmiä ja joissa eri sidosryhmät tarvitsevat potilasdataa mahdollisimman reaaliaikaisesti. Kyseisissä prosesseissa laitteiden ja järjestelmien vakaa toiminta on sairaalan kannalta elintärkeää.

2.2 Tehohoidon valvonnan haasteet

Tehohoitoon ohjataan kriittisesti sairaat potilaat, joille on ilmennyt äkillinen, tilapäinen henkeä uhkaava yhden tai useamman elintoiminnan häiriö. Tarve tehohoidolle voi olla muun muassa onnettomuus, vakava sairaus tai ison leikkauksen jälkihoito. Hoidettavia sairauksia ovat esimerkiksi vaikeaksi edenneet infektiot, hengitysvajaudet, verenkiertovajaudet, elvytyksen jälkihoito, vakava munuaisten toimintahäiriö, aivoverenvuodot ja aivovammat. Lisäksi tehohoito voidaan katsoa tarpeelliseksi isojen leikkausten jälkihoitona, aivoverenkiertohäiriöissä, myrkytyksissä ja akuuttien nestetasapainohäiriöiden aikana. (TAYS 2020)

Tehohoidossa potilaan vointia ja elintoimintoja monitoroidaan valvontamonitoreilla ja elimistön perustoimintoja tuetaan erinäisillä lääkkeillä, hoitolaitteilla, joita on muun muassa erilaiset hengityslaitteet ja munuaisten korvaushoito. Lisäksi potilaalle voidaan suorittaa erinäisiä toimenpiteitä ja leikkauksia. Potilaan tilaa valvotaan erilaisilla laboratorio- ja kuvantamistutkimuksilla, joita on esimerkiksi röntgen-, tietokonekerros- ja magneettikuvaukset. Muita selvittelyitä voivat olla neurofysiologiset tutkimukset. (TAYS 2020)

2.2.1 Koronan hoito teho-osastolla

Koronan takia vuonna 2020 Suomen sairaalojen teho-osastoilla on ollut ruuhkaa. Pääasiallinen syy tehohoitoon joutumiseen on vaikea kaasujenvaihtohäiriö. Potilas siirretään teho-osastolle, kun tämä tarvitsee tehokkaampaa happi- tai hengityslaitehoitoa happimaskin, korkeavirtaushappihoidon tai kaksoispaineventilaation sijasta. Osa teho-osastoilla

käytettävistä hengityslaitteista joudutaan lainaamaan leikkaussaleista, joka hidastaa sairaalaan leikkaustoimintaa. Suomessa hengityslaitteiden uskotaan kuitenkin riittävän tämänhetkiseen hoitoon. (Palmén 2020)

Potilaalle koronan tehohoidossa yleisesti suoritettava toimenpide on intubaatio, jossa hengitysputki asetetaan suoraan tämän henkitorveen ja tarvittava laitehoito annetaan tätä kautta. Useimmat koronaviruspotilaat teho-osastolla kytketään hengityslaitteisiin, jotta heidän hengitystään voitaisiin helpottaa. Potilaan ei välttämättä anneta hengittää lainkaan omin avuin, vaan hengityslaite hoitaa tämän kokonaan heidän puolestaan. Hengityshoidon alkaessa potilas pidetään yleensä nukutettuna, ellei tämän tajunnantaso ole liian matala. Tajunnantasoa madalletaan opiaatti -pohjaisilla ja muilla rauhoittavilla lääkkeillä suonensisäisesti. Potilaan ravinnonsaantia ylläpidetään nenämahaletkun kautta ja nestehoitoa annetaan suonensisäisesti. Potilasta pidetään osastolla myös usein noin 16 tuntia vatsallaan ja loput selällään. Täsmälääkkeen puuttuessa potilaita hoidetaan antibiooteilla bakteerien varalta. (Palmén 2020)

Sairaaloihin on perustettu omia teho-osastoja nimenomaan koronapotilaille, joiden avulla heidät pidetään tarkasti eristyksissä muista sairaalan potilaista. Osastoilla toimiva henkilökunta suojautuu tartunnan riskin minimoimiseksi FFP3-FFP2 -tason suojamaskeilla, suojavisiireillä, laseilla, myssyillä, suojatakeilla ja suojaesiliinoilla. Henkilökunta joutuu säätämään hengityslaitteita useasti päivässä potilaan hoidon tarpeen mukaan. Tehohoidossa olevat koronapotilaat viipyvät teho-osastolla pitkään, yleensä 2-3 viikkoa. Laadukkaasta hoidosta ja hengitystuesta huolimatta, kansainvälisten tilastojen mukaan tehohoitoon joutuneiden potilaiden kuolleisuus on surullinen 30-50 %. (Palmén 2020)

2.3 Anestesia- ja leikkaustoiminnan haasteet

Anestesia, eli nukutus on lääkinnällisin keinoin aikaan saatu tila, jossa potilas on tunnottomassa tilassa, eikä reagoi mitenkään toimenpiteistä aiheutuvaan kipuun. Tällaista nukutuksen muotoa sanotaan yleisanestesiaksi. Nukutuksen aikana potilaan hengitystä, keuhkojen toimintaa, verenkiertoa ja sydämen toimintaa, unen syvyyttä, kivuttomuutta, lihasjänteyttä, lämpötilaa, nestetasapainoa ja leikkauksen vaiheiden vaikutusta potilaan elintoimintoihin valvotaan lukuisin valvontamenetelmin ja avustavien laitteiden avulla. Nukutuksen onnistuttua, hoitajat ja lääkäri varmistavat, että potilas pysyy nukutettuna koko suoritettavan leikkausoperaation aikana. Anestesia- ja leikkaustoiminta kulkevat täten käsi kädessä. (Terveyskylä 2017)

2.4 Lääkinnällisten laitteiden käytön haasteet

Edellä mainittuihin hoitoihin liittyvien laitteiden lisäksi sairaalassa käytetään lukuisia muita lääkinnällisiä laitteita, joilla tuetaan potilaan hoitoa. Laitteilla potilaasta hankitaan vitaalia dataa, joka auttaa hoitajia ja lääkäreitä elintärkeässä päätöksien teossa ja tehostavat potilaan paranemisprosessia.

Esimerkki lääkinnällisten laitteiden valmistajasta on Dräger, joka valmistaa laitteita leikkaussaleihin, tehohoitoon ja vastasyntyneiden osastoille, sekä nukutus -tehtäviin (anestesia). Lisäksi se valmistaa erillisiä ventilaattoreita hengityshoitoon, vastasyntyneiden lämmönsäätely -laitteita, potilasvalvontajärjestelmiä ja hoitotilojen hallintaan sekä kliinisen tiedon hallintaan liittyviä järjestelmiä. (Dräger) Muita lääkinnällisten laitteiden valmistajia ovat esimerkiksi Maquet, GE Healthcare, Phillips ja Fresenius.

Sosiaali- ja terveydenhuollossa käytettävien tietojärjestelmien lisäksi Suomessa lääkinnällisiä laitteita saa tuoda käyttöön sairaalaan vain, jos ne täyttävät niille asetetut vaatimukset. Valmistajan on pystyttävä todistamaan myytävän tuotteen turvallisuus, käyttötarkoituksen sopivuus ja suorituskyky, ennen markkinoille tuomista. Laitteissa tulee vaatimusten mukaisuutta osoittava EU:n CE-merkintä, jolla laitteen valmistaja osoittaa sen täyttävän EU-direktiivien vaatimukset, poikkeustapauksia lukuun ottamatta. (Fimea)

2.5 Lainsäädännön haasteet

Asiakas- ja potilastietojärjestelmien ja sovellusten on noudatettava lukuisia säädöksiä ja lakeja, jotta niitä voidaan harkita edes käytettäväksi varsinkaan sairaalaympäristöön. Erilaisista säädöksistä säädetään maa- ja EU-tasolla ja lisäksi jos dataa käsitellään yli mantereiden, on otettava huomioon myös kohdemaan vallitsevat säädökset ja noudatettava niitä. Erilaisia säädöksiä on noudatettava sovelluksen tai ICT-ratkaisun kehityksen alkuvaiheessa noudattamalla Security by Design -menetelmää, jossa tietoturvaan liittyvät asiat otetaan huomioon koko sovelluksen kehityskaaren aikana. Security by Design:ia käsitellään tarkemmin kappaleessa 6.1.2.

2.5.1 Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä

Laki säätelee Suomen sähköisten sosiaali- ja terveydenhuollon asiakastietojen tietoturvallista käsittelyä, sekä sen edistämistä. Lain mukaan sähköisten asiakastietojen käsittelyssä tulee turvata tiedon saatavuus ja käytettävyys, ja sen eheys ja muuttumattomuus tulee taata koko datan säilytysajan. Laki määrittää, että sosiaali- ja terveydenhuollon palvelujen

antajan tulee pitää rekisteriä omien asiakastietojärjestelmien ja asiakasrekisterien käyttäjistä ja heidän käyttöoikeuksistaan. Potilasasiakirjat tulee pystyä määritellä erityistä suojausta edellyttäviksi. Asiakas, terveydenhuollon palvelujen antaja, muu asiakastietojen käsittelijä tai näiden edustaja sekä tietotekniset laitteet tulee olla tunnistettavissa luotettavasti. Tunnistamisen lisäksi myös todentaminen on pakollista tietojen käsittelyssä. (Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä 9.2.2007/159)

2.5.2 General Data Protection Regulation (GDPR)

GDPR on EU -tasolla säädetty yleinen tietosuoja-asetus, joka asettaa yrityksille ja organisaatioille tarkat vaatimukset henkilötietojen keräämistä, säilytyksestä ja hallinnoimisesta. Säädös koskee yrityksiä ja organisaatioita, jotka käsittelevät EU:n alueella sijaitsevia henkilötietoja, riippumatta siitä, mistä käsin itse käsittely tapahtuu. Lisäksi, jos yritys tai organisaatio sijaitsee EU:n ulkopuolella, mutta hyödyntää henkilötietoja, jotka liittyvät palveluiden tai tuotteiden tarjoamiseen henkilöille EU:n alueella, tai jos yritys/organisaatio seuraa yksilöiden käyttäytymistä EU:n alueella, on noudatettava GDPR:n asettamia vaatimuksia. GDPR:ää ei kuitenkaan sovelleta, mikäli rekisteröity on kuollut, oikeushenkilö tai jos tietoja käsittelee henkilö, joka käsittelee tietoja sellaisessa tarkoituksissa, jotka eivät kuulu tämän alaan, liiketoimintaan tai ammattiin. (Euroopan Unioni 2020)

Henkilötiedoiksi luetaan kaikki tiedot, jotka koskevat tunnistettua tai tunnistettavissa olevaa henkilöä:

- nimi
- osoite
- henkilökortin tai passin numero
- tulot
- kulttuurinen profiili
- IP-osoite
- sairaalan tai lääkärin tiedot henkilöstä (terveydenhuollon piirissä potilaan yksilöivät tiedot)

Yritys tai organisaatio ei kuitenkaan saa käsitellä sellaisia tietoja, jotka koskevat henkilön:

- rotua tai etnistä taustaa
- seksuaalisuutta
- poliittista mielipidettä
- uskonnollista tai filosofista vakaumusta
- ammattiliittoon kuulumista
- geneettisiä, biometrisiä tai terveydellisiä tietoja (paitsi jos käsittelylle on annettu suostumus, tai kyseessä on huomattava yleinen etu EU:n tai kansallisen lainsäädännön perusteella)
- rikostuomioita tai rikkomuksia koskevia tietoja (paitsi jos EU:n tai kansallinen lainsäädäntö toisin sallii)

Henkilötietojen käsittely on sallittua, mikäli yritys tai organisaatio on saanut asianomaisen suostumukset, tietoja tarvitaan käsiteltävän henkilön sopimusvelvoitteen täyttämiseksi, käsittelyyn on laillinen velvoite, suojataan käsiteltävän henkilön elintärkeitä etuja, käsittely tapahtuu yleisen edun mukaisen tehtävän suorittamiseksi tai yritys toimii oikeutetun etunsa rajoissa. Henkilölle, jonka tietoja käsitellään, on selkeästi ilmoitettava:

- kuka tietoja käsittelee
- henkilötietojen käsittelyn syy
- käsittelyn oikeusperusta
- tarvittaessa tietojen vastaanottaja

Lisäksi joissain tapauksissa käsittelevän yrityksen tai organisaation on myös pystyttävä näyttää toteen:

- tietosuojavastaavan yhteystiedot
- yrityksen oikeutettu etu, kun sellaista pidetään käsittelyn oikeudellisena perustana
- sovellettavat toimenpiteet tietoja siirrettäessä EU:n ulkopuoliseen maahan
- tietojen säilytysaika
- henkilön tietosuojaoikeudet (oikeus nähdä, korjata, poistaa, rajoittaa tai vastustaa omien henkilötietojen käsittelyä tai siirtää tiedot toiseen järjestelmään)
- omien tietojen käsittelyn suostumuksen perumisen prosessi
- henkilötietojen antamisen lainvoimaisuus tai sopimukseen perustuva vaatimus
- automaattiseen päätöksentekoon liittyvät tiedot logiikasta, merkityksestä ja seurauksista

(Euroopan Unioni 2020)

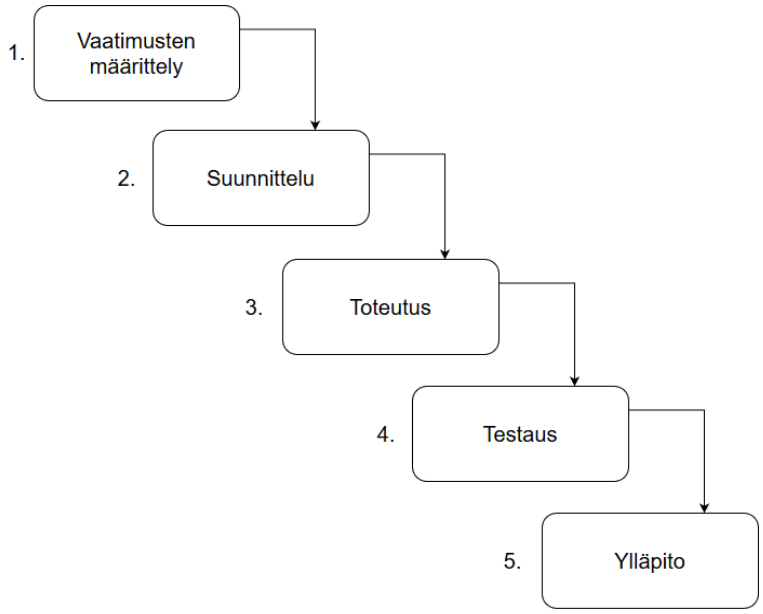
3 Sovelluskehitysmallien teoriaa

3.1 Erilaiset kehitysmenetelmät ja niistä sopivimman valinta

Sovellusta kehittämään lähtiessä tulee tarkastella haluttua ratkaisua tilannekohtaisesti ja päättää sille paras lähestymistapa, oikea kehitysmenetelmä. Menetelmiä on lukuisia ja tulee suhteuttaa tilannekohtaisesti. Käytettävästä mallista voidaan keskustella yhdessä asiakkaan kanssa, mutta lopullisen päätöksen mallin valinnasta tekee kuitenkin yleensä palvelun tuottaja.

3.1.1 Waterfall

Vesiputousmalli – Waterfall model, on Winston Roycen vuonna 1970 kehittämä prosessimalli, jota hyödynnetään erityisesti sovelluskehityksessä. Malli ei ole itsessään ollenkaan joustava, mutta se on yksinkertainen ja jakaa kaikki kehityksen vaiheet selkeisiin vaiheisiin. (Helsingin yliopisto 2009)



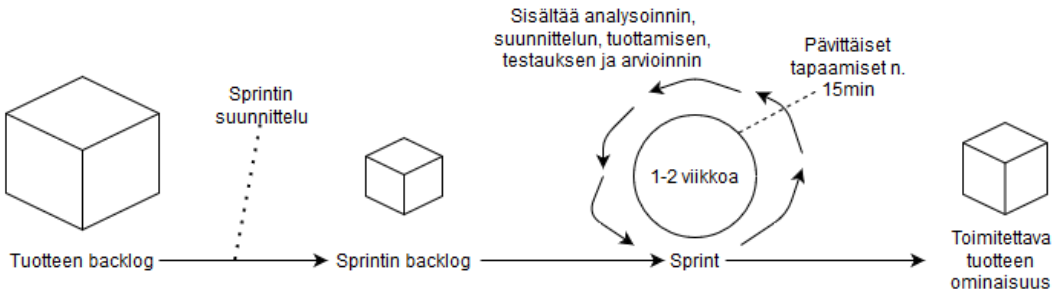
Kuva 1. Vesiputousmalli (mukaillen Thinking portfolio 2016)

Mallin avulla on helppo suunnitella projektien vaiheistusta ja siirtymiä, mutta ohjelmistotuotannossa se on turhan kankea. Mallista on kuitenkin olemassa useita vaihtoehtoisia käyttötapoja, jotka ovat käytössä lukuisissa ohjelmistokehitys -yrityksissä. Kaikista malleista vesiputousmalli tarjoaa käyttäjälleen parhaimman teoria- sekä työkalutuen. (Helsingin yliopisto 2009)

3.1.2 Agile & Scrum

Agile, eli ketterät menetelmät ovat kykyä luoda ja vastata jatkuvasti muuttuvaan kehitykseen, työkaluihin ja asiakkaan tarpeisiin. Ketterät työskentelytavat kuten Scrum rakentuvat useiden menetelmien ympärille, kuten pariohjelmointi, testauskehitys, päivittäiset tapaamiset, suunnittelu -sessiot ja sprintit. Ketterissä kehitysmalleissa keskitytään muista kehitysmenetelmistä poiketen työskenteleviin ihmisiin ja kuinka he voisivat työskennellä paremmin yhdessä. (Agilealliance)

Ratkaisut ja ratkaisujen toteutusmenetelmät syntyvät keskenään toimivien tiimien yhteistyön ja itse -organisoinnin tuloksena. Agile -menetelmissä kuten Scrum, ei ole varsinaista ”johtajaa”, joka käskee tiimin tehdä tiettyjä asioita, vaan hän on henkilö, joka pyrkii edistämään omaa tiimiään olemaan tehokkaampi ja onnistuneempi. Agile -tiimin johtaja antaa oman tiiminsä kehittää itse ratkaisut prosessin aikana ilmeneviin ongelmiin, mutta pyrkii avustamaan, jos tiimi ei itse saa kehitettyä ratkaisua vallitsevaan ongelmaan. Ketterät -menetelmät ovat täten paljon joustavampia kehitysmalleja, joiden avulla alkuperäistä suunnitelmaa ja asetettuja tarpeita voidaan paremmin muokata ja antaa itse sovelluskehittäjille vapaammat kädet kehitystyökalujen ja puitteiden valintaan. Menetelmä myös mahdollistaa kehittäjien paremman reagoinnin kehityksen aikana syntyviin esimerkiksi teknillisiin ongelmiin. (Agilealliance)

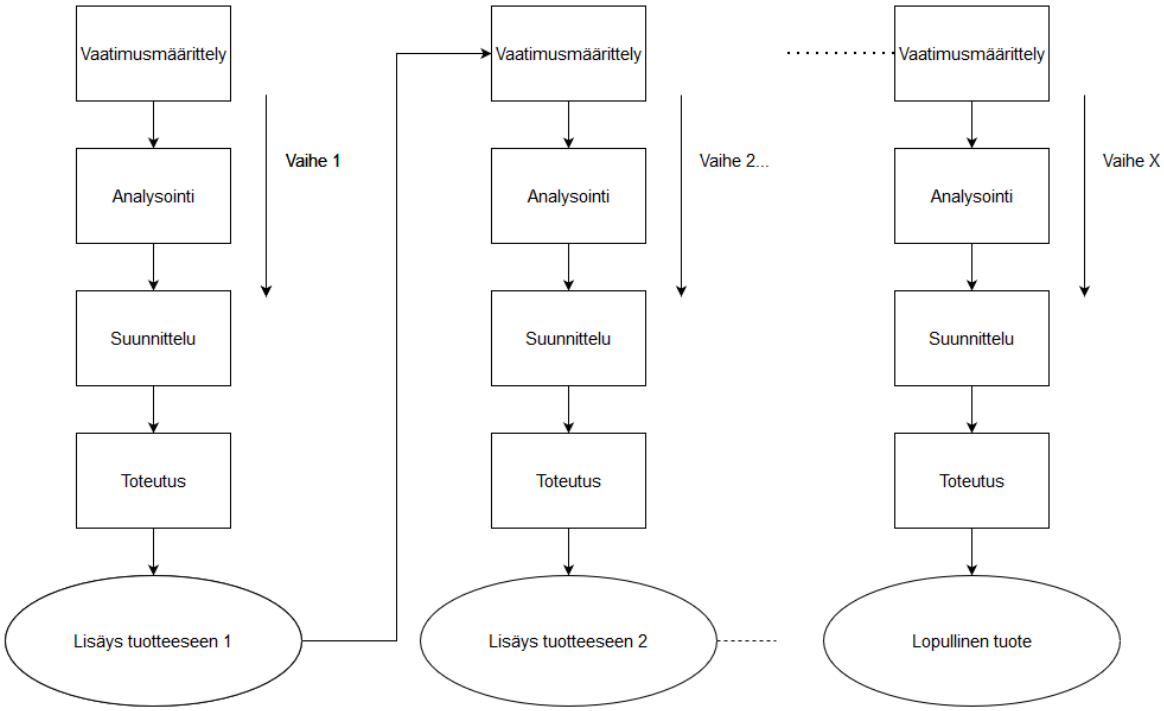


Kuva 2. Agile (mukaillen Kotian 2017)

Scrum menetelmiä noudattaen sovelluskehitysprojekti etenee Sprintein, jotka ovat iteraatioita, joilla on määrätty aikaväli. Sprinttien pituus on usein kahdesta kolmeen viikkoon, mikä sopii parhaiten omalle tiimille. Sprinttien pituus ei kuitenkaan koskaan tulisi olla pidempiä kuin kuukausi. (Rawsthorne)

3.1.3 Incremental & Iterative

Incremental -asteittainen ja Iterative -toistuva -kehitys on sovelluskehitysmalli, jossa toistetaan vaiheita, jotta voidaan tuottaa valmis tuote. Vaiheita toistetaan niin kauan, kunnes lopullinen tuote on valmis. (Tutorialspoint)



Kuva 3. Asteittainen -kehitys (mukaillen Tutorialspoint)

Asteittaisen kehityksen hyötyjä on muun muassa kyky tuottaa laadukas vaatimuslista, nopea alkutuotteen toimitus, tärkeät toiminnallisuudet voidaan tuottaa kehityksen alkuvaiheessa, kustannustehokkuus eri vaiheissa, asiakkaalla on jatkuvasti toimiva tuote käsissä, johon lisätään jatkuvasti ominaisuuksia, palautteen saanti asiakkaalta on helpompaa ja vaatimusmuutokset voidaan helposti ottaa huomioon. (Tutorialspoint)

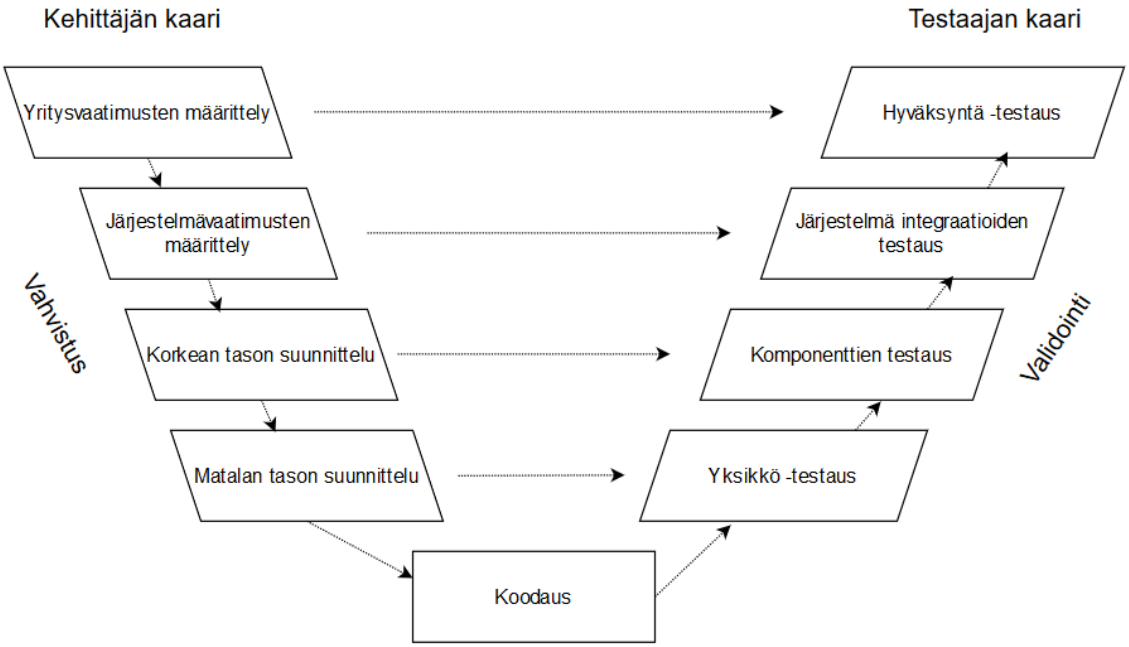
Asteittainen kehitys kuitenkin vaatii jokaisen vaiheen tarkkaa suunnittelua, tuotteen rakenne tulee olla helposti muokattavissa uusien ominaisuuksien jatkuvan implementoinnin takia, halutun lopputuotteen ominaisuudet on tiedettävä kehityksen alkuvaiheessa, eikä lopullisen tuotteen yhteiskustannukset ole todellisuudessa pienemmät kuin muissa malleissa. (Tutorialspoint)

Asteittaista kehitystä kannattaa käyttää, kun suurin osa lopputuotteen vaatimuksista tiedetään ennalta ja ne on kyetty priorisoimaan. Perustoiminnallisuuden toimituksen tulee olla nopeaa, ja sillä tulee olla pidennettävissä oleva aikataulu, tai projekti hyödyntää paljon uusia erilaisia teknologioita. (Tutorialspoint)

3.1.4 V-Shaped

V-mallista voidaan puhua myös todennus- ja validointimallina. V-mallin vallitseva kehitysvaihe tulee suorittaa kokonaan loppuun, ennen kuin seuraava voi alkaa ja täten se vastaa perusrakenteeltaan hyvin paljon vesiputousmallia. Mallissa varmistetaan, että toimittajan ja asiakkaan näkökulmat kohtaavat. (Javatpoint)

Tuotteelle asetetaan kehittäjien näkökulmasta vaatimukset, jotka vahvistetaan asiakkaalla, jotta varmistetaan yhteisymmärrys lopullisesta tuotteesta sen jokaisessa kehitysvaiheessa. Vahvistus -vaiheet eivät sisällä ollenkaan koodausta, vaan siinä varmistetaan, että asetetut vaatimukset täyttyvät yhdessä asiakkaan kanssa. Validointi -vaiheessa varmistetaan sovelluksen toimivuus ajamalla kehitettyä koodia ja todentamalla tämän toimivuus asiakkaan vaatimuksien kanssa. V-malli on helposti omaksuttava, sisältäen kattavan suunnittelun ja testauksen, säästäen aikaa. (Javatpoint)



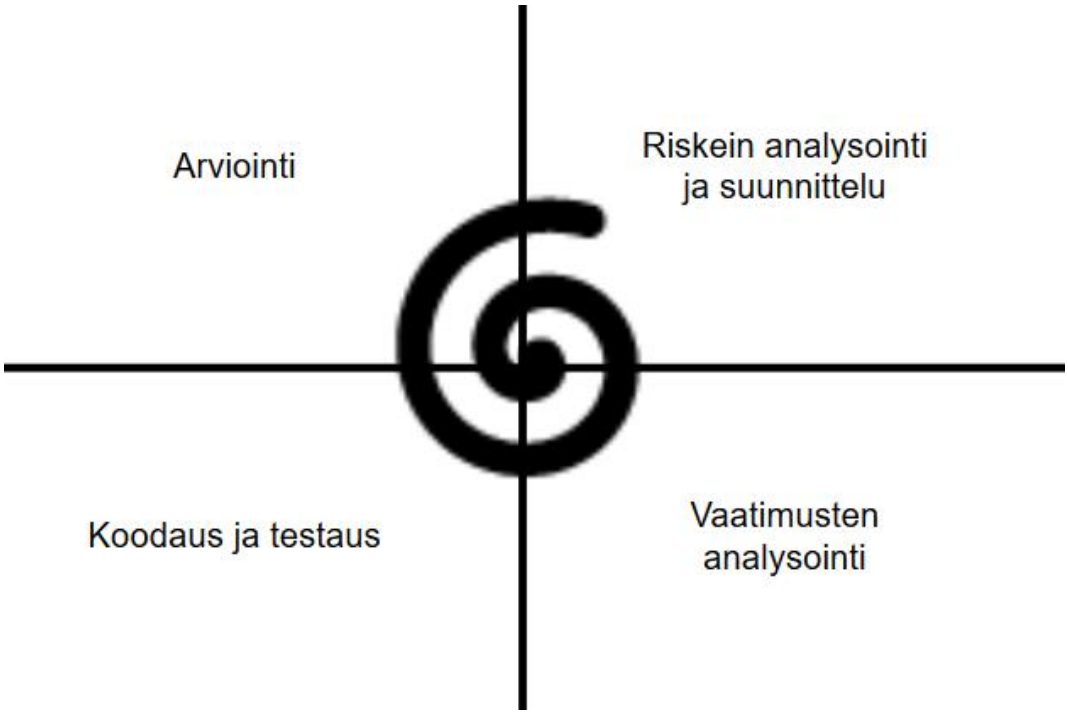
Kuva 4. V-malli (mukaillen Javatpoint)

V-mallia voidaan hyödyntää pienissä ja keskisuurissa projekteissa, jossa vaatimukset ovat selkeitä ja hyvin määritelty. Mallia voidaan hyödyntää myös, mikäli saatavilla on teknillisiä lähteitä esimerkkeineen. (Javatpoint)

3.1.5 Spiral

Spiraali -malli on sovelluskehitys -malli, jonka tarkoituksena on vähentää projektin riskiä, hyödyntäen vesiputous- ja prototyyppi -mallien menetelmiä. Mallissa toistetaan kehityksen vaiheita, kunnes lopullinen tuote syntyy. (Neha 2020)

Tuotteen kehitys lähtee spiraalin keskiosasta ja etenee ulospäin, seuraten spiraalin muotoa, joka näyttää projektin etenemisen. Spiraalin viivan pituus havainnollistaa myös vaiheeseen käytettävää aikaa. Jokaisessa periodissa kartoitetaan vallitsevat riskit ja niiden minimoimiseksi vaadittavat toimenpiteet, joka on mallin tärkeimpiä ominaisuuksia. Kun koko spiraalin sykli käydään läpi, mallissa julkaistaan ”evoluutiollinen tuote”. Se voi olla esimerkiksi pelkkä malli tuotteen vaatimuksista tai jopa prototyyppi. Spiraalin sykliä toistamalla luodaan lopulta lopullinen tuote. (Neha 2020)



Kuva 5. Spiraali (mukaillen Neha 2020)

Mallia kannattaa käyttää projekteissa, jotka sisältävät suuria riskejä ja mahdollistavat prototyypin kehittämisen. Spiraali osoittaa potentiaalinsa myös, kun asiakas ei kykene määrittelemään kaikkia lopullisen tuotteen vaatimuksia, projektin aikataulu on pitkä, jatkuvaa tuotteen arviointia, tarkastelua ja asiakaspalautetta vaaditaan. Malli ei kuitenkaan toimi pienissä projekteissa. (Neha 2020)

3.2 Lopullisen sovelluksen kehitysmallin valinta

Sairaalaan kehitettävän sovelluksen kehitysmallin valinta riippuu tilanteesta, mutta useimmiten kyseisten järjestelmien kompleksisuuden ja lukuisien ominaisuuksien takia on hyvä käyttää tilanteeseen sopivaa ketterää mallia – Agile, joka luonteeltaan kykenee helposti vastaamaan muutostarpeisiin. Ketterällä kehitysmallilla voidaan helpoiten hyödyntää uusia teknologioita, ottaa huomioon asiakkaiden muuttuvat tarpeet ja saadaan kehittäjille vapaammat kädet työkalujen valintaan. Koska ketterissä malleissa voidaan helposti reagoida projektin aikana syntyviin muutostarpeisiin, ovat ne suosittuja myös sairaalaympäristöön kehittäessä.

Seuraavissa kappaleissa tarkastellaan Agile -kehitysmallin Scrum:in hyödyntämistä sovelluksen kehityksessä sairaalaan ja kuinka prosessi eroaa perinteisen sovelluksen kehityksestä.

3.3 Sovelluskehitysprojektin valmistelu

Projektin valmistelussa (Sprint 0) valmistaudutaan tulevaan projektiin ja luodaan peruspuitteet sen toteutukselle:

- Luodaan ”toteutus arkkitehtuuri”, joka sisältää kehityksen, laadunvalvonnan ja testiympäristöjen luonnin. Tällä varmistetaan, että kun projekti lähtee liikkeelle, sovelluskehitystiimi voi aloittaa työskentelyn jo ensimmäisenä päivänä.
- Suunnitellaan alustava kattava tuotteen backlog, jotta sprint 1:n suunnittelu olisi mahdollisimman helppoa.
- Projektin ja tapaamisten aikataulutus
- Alustava sovelluksen arkkitehtuurin suunnittelu
- Työkalujen valinta: Scrum taulut, Jira, Wrike ja niin edelleen

(Herrick 2016)

Valmistelujen lisäksi sovitaan, milloin sprint voidaan tulkita valmiiksi, jotta voidaan siirtyä seuraavaan kehitysvaiheeseen. Määritellään, mitä tulee olla valmiina ja milloin voidaan tulkita, että haluttu ominaisuus on ”valmis”. (Herrick 2016)

Ketterissä menetelmissä on tärkeää luoda sovellukselle priorisoitu tehtävä- ominaisuuslistaus eli sovelluksen backlog. Sen tarkoituksena on mahdollistaa iteraatioiden tarkka suunnittelu, joka sisältää myös vaiheet, jotka eivät koskaan näy asiakkaalle. Backlogin sisältämät tehtävät tai ominaisuudet kirjataan sovelluksen toiminnallisuuksina ja käyttökuvauksina, joita sanotaan ”Epiceiksi”. (Radigan) Esimerkiksi: ”käyttäjä voi sisään kirjata potilaan sairaalaan”. Tällaisilla ”tarinamaisilla” kirjauksilla pakotetaan myös sovelluskehityksen ”piilovaiheet” esiin. Jotta käyttäjä voi sisään kirjata potilaan sairaalaan on tällä oltava jokin

käyttöliittymä, johon tiedot kirjataan. Lisäksi tietojen on tallennuttava johonkin, joten sovelluskehittäjän tulee implementoida jonkinlainen tietokanta, johon potilaan tiedot tallennetaan. Sairaalaa -ympäristössä tulee myös huomioida potilaiden anonymisointi, eli millaista dataa potilaasta tallennetaan ja tallennetaanko se tietokantaan selkeästi luettavaan muotoon vai salaamismenetelmin. Täten yksi ominaisuus sisältää lukuisia vaiheita ja pakottaa sovelluskehittäjän miettimään ratkaisua laajemmin ja suhteuttamaan sen suunniteltuun arkkitehtuuriin. Backlogissa pääkohdat kannattaa kirjata tarinallisin ominaisuuksin, eli Epic:einä, joiden alle voidaan yksilöidä ala -tehtäviä, kuten tietokannan suunnittelu, käyttäjätietojen anonymisointi ja niin edelleen. Sovelluskehittäjät voivat täten valita backlogista tehtäviä tehtäväkseen, kun heillä on resursseja tämän toteutukseen. Backlogissa olevia tehtäviä voidaan priorisoida käyttäjän tarpeen, palautteen saannin tärkeyden tai tehtävän vaikeuden mukaan.

3.4 Sovellusarkkitehtuurin suunnittelu

Jotta varsinkin laajempi ja vaativampaan ympäristöön kehitettävä sovellus voi olla onnistunut, on syytä kiinnittää huomiota kehitettävän ratkaisun sovellusarkkitehtuuriin ja tämän suunnitteluprosessiin. Yleisenä luulona on, että sovellusarkkitehtuuri on luovaa työtä, jolla ei ole omia varsinaisia määriteltyjä prosessejaan. Todellisuudessa sovellusarkkitehtuuri on prosessi, joka sisältää korkean skaalan suunnittelua, suurille, monimutkaisille järjestelmille ja on luonteeltaan kurinalainen prosessi, jolla tuetaan kehittäjien luovuutta järjestelmällisesti ja vuorovaikutteisemmin. Sovellusarkkitehtuurille on täten oma kiertokulkunsa, jolla on omat vaiheet ja tehtävät ja joilla on kaikilla omat ennakkovaatimukset ja soveltamiskohteensa. (Ali Babar, Brown & Mistrik. 2013. 4-5.)

Alkuperäinen sovellusarkkitehtuurin suunnittelu perustuu viidestä vaiheesta:

I. Ongelman analysointi: Määrittele ICT-ratkaisulla ratkaistavat ongelmat, tarkastele arkkitehtuurilliset vaatimukset, ota huomioon sidosryhmien mielipiteet ja priorisoi, listaa ja erittele arkkitehtuurilliset vaatimukset niiden tärkeyden mukaan.

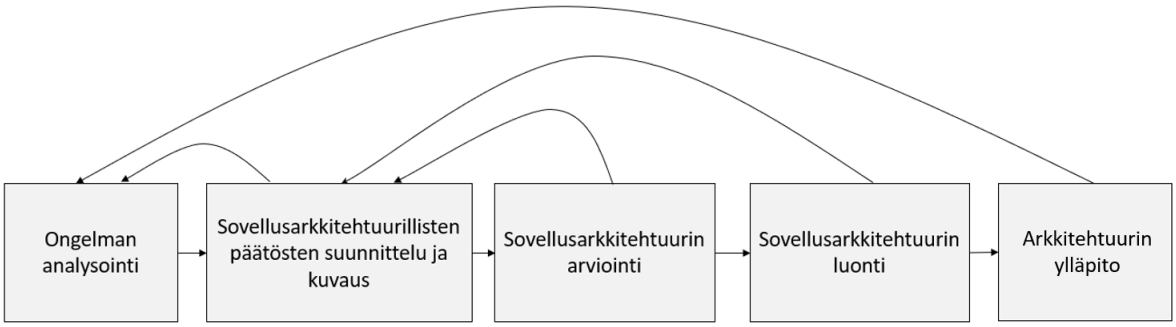
II. Sovellusarkkitehtuurillisten päätösten suunnittelu ja kuvaus: Päätetään keskeiset päätökset arkkitehtuurin suunnittelua koskien. Arkkitehti suunnittelee muutamia vaihtoehtoja järjestelmäarkkitehtuurin toteutukselle ja valitsee niistä tilanteeseen sopivimman. Arkkitehti dokumentoi suunnitellun arkkitehtuurin käyttäen asianmukaisia dokumentteja ja pohjia.

III. Sovellusarkkitehtuurin arviointi: Varmistetaan, että edellisessä vaiheessa tehdyt päätökset arkkitehtuurillisista ratkaisuista ovat tilanteeseen nähden sopivimmat. Varmistetaan, että kehitetyt arkkitehtuurilliset ratkaisut ratkaisevat ja tukevat ensimmäisessä kohdassa asetetut ongelmakohtia.

IV. Sovellusarkkitehtuurin luonti: Tässä vaiheessa arkkitehtuuri toteutetaan yksityiskohtaisesti ja otetaan käyttöön. Sovelluskehittäjät joutuvat kyseisessä vaiheessa kiinnittämään erityistä huomiota siihen, että heidän tekemänsä ratkaisut ovat linjassa sovellusarkkitehdin asettamien korkean tason vaatimusten kanssa.

V. Arkkitehtuurin ylläpito: Uusia arkkitehtuurillisia päätöksiä toteutetaan aikaisemman arkkitehtuurin päälle sovelluksen muuttuessa tai kasvaessa, kuitenkin että erinäiset suunnittelupäätökset arvioidaan uudestaan ja suhteutetaan mahdollisiin niiden aiheuttamiin vaikutuksiin täten ja tehdään uusia rakenteellisia päätöksiä vahingoittamatta sovellusarkkitehtuurin eheyttä.

Jokaiseen sovellusarkkitehtuurin suunnittelu -vaiheen edellisiin vaiheisiin voidaan myös esimerkiksi Vesiputous -mallista poiketen palata, vaikkakin toista vaihetta toteutettaisiinkin jo. (Ali Babar ym. 2013. 5-6.)



Kuva 5. Sovelluksen arkkitehtuurillinen prosessi (mukaillen Ali Babar ym. 2013. 5.)

3.4.1 Arkkitehtuurin dokumentointi ja arviointi

Sairaalaan sovellusta kehitettäessä sovelluksen dokumentointiin on kiinnitettävä erityistä huomiota perinteisestä sovelluksen dokumentoinnista poiketen. Dokumentoinnilla on muun muassa pystyttävä osoittamaan alan valvoville tahoille kuten Valviralle Suomessa, että tuote vastaa asetettuja säädöksiä ja lakeja, kuten esimerkiksi lakia sosiaali- ja terveyhdenhuollon asiakastietojen käsittelystä (159/2007).

Arkkitehtuurin dokumentoinnilla varmistetaan, myös sidosryhmät kykenevät hyödyntämään sovelluskokonaisuutta tehokkaasti. Kattava dokumentaatio sovelluksen arkkitehtuurista tarjoaa sidosryhmille ”avaintietoa” sovelluksen tärkeimmistä arkkitehtuurillisista kohdista, joita voidaan hyödyntää tulevaisuudessa arkkitehtuurin jatkosuunnittelussa, tehtävien erittelyyn ja ylläpidollisissa asioissa. Kattavan dokumentaation tuottaminen on kuitenkin paljon resursseja ja aikaa vaativaa työtä, jota ei yleensä ketterissä menetelmissä käytetä. Kehitettävän sovelluksen luonteesta kuitenkin riippuen tämä voi olla välttämätöntä. (Ali Babar ym. 2013. 9.)

Sovellusarkkitehtuurin suunnittelun valmistuttua, se arvioidaan kriittistä arviointikykyä käyttäen. Arviointi on tärkeä osa sovellusarkkitehtuurin luontiprosessia, sillä arvioinnilla varmistetaan, että arkkitehtuuriehdotus tai valinta täyttää sille asetetut laadulliset vaatimukset ja tunnistetaan sen mahdollisia uhkia. Tyypillisimpiä arviointitapoja on skenaario -pohjaiset arvioinnit, joilla voidaan konkretisoida ratkaisun käyttötilannetta ja vastaako se tosiaan kaikkiin asetettuihin ongelmiin. Arvioinnissa voidaan esimerkiksi käydä vaihe vaiheelta läpi, miten hoitaja tai lääkäri käyttäisi sovellusta sairaalassa. Arvioinnilla halutaan ensisijaisesti varmistaa, että ylläpito ja käytettävyys ovat hyvällä tasolla esimerkiksi performanssin tai skaalautuvuuden sijasta. (Ali Babar ym. 2013. 9.)

3.5 Scrumin eteneminen ja seuranta

Projekti etenee suorittamalla lyhyitä iteraatioita eli sprinttejä. Sprintit suunnitellaan priorisoimalla tuotteen backlogissa olevia Epic:kejä ja näiden sisältämiä yksityiskohtaisempia tehtäviä. Jokaisen sprintin tuloksena tulisi olla joka kerta parempi tuote uusin ominaisuuksin. Sprinttejä jatketaan niin kauan, että sovellus on valmis ja aikaa riittää. (Herrick, 2016)

3.5.1 Päivittäispalaveri

Scrumissa toteutetaan päivittäisiä epävirallisia tapaamisia, jotka pidetään seisten. Päivittäisen Scrumin (Daily Scrum) tarkoituksena on kartoittaa oman tiimin tilanne, mitä kukin on viimeksi tehnyt ja mitä tulee tekemään seuraavaksi. Daily Scrumissa katsastellaan myös mahdollisia ongelmia päivän tehtävien suorittamiselle ja ratkaistaan nämä. Daily Scrum on aina pituudeltaan enimmiltään 15 minuuttia, jota ei tule ylittää. (Itemis)

3.5.2 Scrum sprinttien arviointi

Sprintit aina arvioidaan arvostelu -tapaamisessa, johon osallistuu tuotteen omistaja ja muut sidosryhmät. Tuotetut tulokset arvioidaan ja Scrum -tiimi saa palautetta tästä. Arvioinnin jälkeen pidetään Scrum -tiimin sisäinen kokous (Retrospective), jossa arvioidaan, mitä tiimi teki hyvin, missä on parantamisen varaa ja kuinka ongelmakohtia voidaan korjata. (Itemis)

3.6 Roolit Scrum:issa

3.6.1 Scrum Master

Jokaisella Scrum -tiimillä on nimetty henkilö Scrum Masteriksi, joka ei ole projektin johtaja vaan enemmänkin ohjaaja. Scrum Masterin tehtävä on tukea tiiminsä toimintaa ja varmistaa, että Scrum -menetelmiä noudatetaan. Scrum Master palvelee koko projektin eri osapuolia eri tavoin:

Tuotteen omistaja (Product Owner):

- Tavoitteiden toteutumisen, laajuuden ja tuotteen käyttötarkoituksen ymmärtämisen varmistaminen omalla Scrum tiimille mahdollisimman hyvin.
- Backlogin hallintatekniikoiden kehitys.
- Varmistaa, että Scrum tiimi luo selkeän ja ytimekkään backlogin.
- Scrum Master ymmärtää tuotesuunnittelun empiirisessä ympäristössä.
- Varmistaa, että tuotteen omistaja osaa järjestää tuotteen backlogia.
- Ymmärtää ketteriä menetelmiä.
- Järjestää tapaamisia, mikäli tarpeellista.

Scrum tiimi (Development Team)

- Valmentaa tiimiä toimimaan paremmin yhdessä.
- Avustaa tiimiä luomaan laadukkaita tuotteita.
- Pyrkii estämään häiriöiden syntymistä.
- Kouluttaa tiimiä hyödyntämään Scrummia, mikäli tarvetta.

Organisaatio

- Johtaa organisaation Scrum -ymmärrystä
- Scrumin hyödyntämisen suunnittelu
- Auttaa ymmärtämään kaikkia osapuolia Scrumin käytöstä ja empiirisestä tuotekehityksestä
- Pyrkii muutoksiin, joilla tehostetaan Scrum -tiimin toimintaa
- Muiden Scrum Mastereiden kanssa yhteistyö, jolla tehostaan Scrumin käytäntöjä organisaation sisällä.

(Scrum.org)

3.6.2 Tuoteomistaja (Product Owner)

Tuotteen omistaja on henkilö, joka pyrkii tehostamaan tuotteen arvoa. Tuotteen omistaja on Scrumin mukaan vastuussa oleva henkilö tuotteen backlogin hallinnasta. Tuotteen omistaja pyrkii selkeästi ilmaisemaan backlogissa olevat tiedot, backlogin järjestäminen, optimoi kehitystiimin tuottamaa arvoa, varmistaa backlogin selkeyden, läpinäkyvyyden ja seuraavat vaiheet sekä varmistaa, että kehitystiimi ymmärtää backlogissa olevat kohdat riittävällä tasolla. (Scrum.org)

3.6.3 Tuotteen kehitystiimi (Development Team)

Kehitystiimi on koottu joukko ammattilaisia, jotka tuottavat itse tuotteen. Kehitystiimi organisoi kehitettävää tuotetta ja omaa työtään itse, joka tehostaa tiimin kokonaisvaltaista tehokkuutta. Kehitystiimit toimivat tiiviissä yhteistyössä keskenään jakaakseen tietämystään ja tuottaakseen halutun tuotteen. Kehitystiimin jäsenillä ei tunnisteta olevan eriarvoisia titteleitä, vaan ovat kaikki samanarvoisia tiimin sisällä. Jokaisella tiimin jäsenellä saattaa olla omia osaamisalueita, joihin he keskittyvät, mutta vastuu projektista jakautuu tiimin sisällä tasan. Kehitystiimi koostuu yleensä 3-9:stä henkilöstä. (Scrum.org)

4 Uudet ICT-ratkaisut sairaalan tietojärjestelmien tukena

4.1 Uudet ICT-ratkaisut lisäävät sujuvuutta

Terveyspalvelujen saatavuus, laatu ja kustannukset ovat nykypäivänä haaste, jolla on kova yhteiskunnallinen paine. Erilaisilla moderneilla digitalisaation tuomilla ICT-ratkaisuilla pyritään parantamaan näitä ongelmakohtia, niin että ne ovat konkreettisesti todettavissa. (Salo 2019)

4.1.1 Esimerkki Kaunialan sairaalasta

Kaunialan sairaalassa vanha tietojärjestelmä oli seitsemän vuotta vanha ja ”hajoamispisteessä”. Käyttäjillä oli lukuisia vaikeuksia järjestelmään kirjautumisessa, joka kesti joka kerta noin 3-4 minuuttia, mikä hidasti hoitohenkilökunnan toimintaa huomattavasti. (Arrow ECS Finland 2017, 0:00 – 0:18)

Ongelmaa lähdettiin ratkomaan Citrix -ratkaisun -avulla, joka on sovellusvirtualisointialusta, jolla työasema luodaan virtuaalisesti. Järjestelmän oli tarkoitus olla mahdollisimman luotettava, turvallinen ja varsinkin nopea. Citrixin tuotteita on käytössä myös muissa sairaaloissa, joissa kyseisen palveluntarjoajan palvelut ja tuotteet on todettu myös hyvin toimiviksi. Lopullinen ratkaisu tuotettiin hyödyntämällä Citrixin NetScaler -tuotetta, joka takaa, että loppukäyttäjän palveluiden toimivuus ja palveluihin kiinni pääseminen toimii aina. Työasemille voidaan kirjautua helposti, nopeasti ja turvallisesti tunnistekorttia käyttämällä. Palvelut toimivat nopeasti ja vakaasti kellon ympäri, sillä ne on rakennettu Atlantis HyperScale -alustan päälle, joka on uusinta uutta tekniikkaa. (Arrow ECS Finland 2017, 0:18 – 1:41)

Sairaalahenkilökunta on ollut hyvin tyytyväinen uuteen järjestelmään, sillä alkuperäinen kirjautumisongelma on poistunut kokonaan, työasemien käynnistyminen on nopeaa ja koko ympäristö toimii luotettavasti. ICT-ratkaisun toteutuksen seurauksena sairaalan IT-tukipyynnöt ovat vähentyneet n. 90 %. Aikaisemman järjestelmän aikana joskus hoitajan kirjautuminen järjestelmään saattoi kestää jopa 30min, mutta nykyään kirjautumiseen menee n. yksi minuutti. Vuositasolla tästä koituu useiden henkilötyövuosien säästöt. (Arrow ECS Finland 2017, 1:41 – 2:41)

ICT-ratkaisuilla voidaan täten saavuttaa merkittäviä vuositason säästöjä ajallisesti ja rahallisesti, parantaa ja helpottaa hoitohenkilökunnan työtä, nopeuttaa potilaan virheettömämpää hoitoa, sekä suojata ja hyödyntää hoidosta saatavaa dataa turvallisemmin. ICT-

ratkaisut helpottavat myös lisäksi sairaalan johtoa johtamaan sairaalan toimintaa entistäkin onnistuneempaan suuntaan. (FrontEnders 2016)

4.1.2 Esimerkki Mainiokoti Kaislasta

Mainiokoti Kaisla tuottaa tehostettua asumispalvelua ikäihmisille. Suurella osalla asiakkaista on jo vanhuuden tuoma jonkinlainen muistisairaus tai toimintakyvyn heikkeneminen, niin ettei kotona asuminen enää onnistu. Kaislassa on otettu käyttöön uutena ICT-ratkaisuna hälytysranneke, jonka on tuottanut 9Solutions. Ranneke on käytössä kaikilla ikäihmisillä, jotka sitä kuinkin pystyvät vain käyttämään. Rannekkeella voidaan kutsua hoivakodin hoitaja huoneeseen, kun asukkaalla on hätä, toive, seurustelutarve tai vain jotain kerrottavaa. Ranneke osaa myös indikoida hoitajille, mikäli rannekkeen käyttäjä kulkee kielletylle alueelle tai vaikkapa karkaa, niin tämä tuottaa hälytyksen hoitohenkilökunnan puhelimeen. Rannekkeen avulla asukas saa avun paikalle tehokkaasti oikeaan paikkaan, kun sitä tarvitsee ja lisää hoitohenkilökunnan ja asukkaan välistä vuorovaikutusta. (9Solutions 2018)

5 ICT-ratkaisujen kehittäminen sairaalaympäristöön

Sairaalat koostuvat lukuisista osastoista ja eri hoitoalojen osa-alueista, joilla on kaikilla omat toimintatapansa. Ratkaisun on sulauduttava hyvin sairaalan omiin toimintamalleihin ja sillä on oltava konkreettista hyötyä käyttäjilleen. Sovelluksilla pyritään ajallisiin ja rahallisiin säästöihin, joilla pyritään parantamaan sairaalan hoidon tehokkuutta ja laatua.

Laajan käyttäjien kirjon ja toiminnallisuuksien takia, suurin osa sairaaloista haluaa hyödyntää sovelluksia, jotka eivät tuo ratkaisua vain yhteen ongelmaan, vaan tarjoavat monipuolisia toiminnallisuuksia ja ratkovat useita sairaalan ongelmakohtia tehokkaasti. Kehitettävän sovelluksen tulee tukea koko sairaalan toimintaketjua ja tukea sairaalan johdon, IT -osaston, lääkärien ja hoitajien toimintaa. (Blogspot 2017)

Sovelluksessa tulee olla mahdollisuus käsitellä potilaan digitaalisia potilastietoja, jotka koskevat potilaan hoitoa. Myös datan ylläpito, säilytys ja jakaminen tulee olla huomioitu myös sovellusta ylläpitävän tahon näkökulmasta. Mikäli mahdollista, kehitettävään ratkaisuun tulee implementoida mahdollisuus hallita potilaan hoitoa: aikaleimaukset, potilaan hoidon suunnittelu, milloin potilas on saanut hoitoa ja niin edelleen. Potilas on myös mahdollisesti saanut hoitoa toisen ammattilaisen toimesta eri paikkakunnalla, jolloin on tärkeää tämän hoidon tehokkuuden ja laadun kannalta, että terveydenhoidon ammattilaisilla on pääsy myös potilaan aikaisempaan dataan. Täten eri järjestelmien väliset integraatiot ovat keskeinen elementti. Monimutkaiselta kuulostava järjestelmä ei kuitenkaan saa olla monimutkainen käyttäjien näkökulmasta. On huomioitava, etteivät käyttäjät välttämättä ole IT -alan ammattilaisia ja kehitettävän sovelluksen on täten taattava yksinkertaisuus ja helppokäyttöisyys, jotta sen käyttö on myös tehokasta, eikä itsessään hidasta sairaalan toimintaa. (Blogspot 2017)

GE Healthcaren sovelluskehitystiimin johtajan Mari Karhusen mukaan perinteisen sovellusratkaisun yritykselle tai organisaatiolle tulee ratkaista jokin konkreettinen ongelma tai tuoda helpotusta liiketoiminnan toteuttamiselle. (Karhunen 2020) Päämäärä on täten sama kuin sairaaloihin kehittävien sovellusten kanssa, mutta perinteisissä sovelluksissa lainsäädäntö, standardit ja validointi eivät ole yhtä tiukkaa. Esimerkiksi Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä 19 §:n mukaan ”Sosiaali- tai terveydenhuollon asiakastietojen käsittelyssä käytettävän tietojärjestelmän tulee täyttää yhteen toimivuutta, tietoturvaa ja tietosuojaa sekä toiminnallisuutta koskevat olennaiset vaatimukset.” Lisäksi 19 §:ssä mainitaan että: ”Valmistajan on annettava tietojärjestelmän yhteydessä järjestelmän käyttäjälle yhteen toimivuuden, tietoturvallisuuden ja tietosuojan sekä toiminnallisuuden kannalta tarpeelliset tiedot ja ohjeet järjestelmän käyttöönotosta,

tuotantokäytöstä ja ylläpidosta. Tietojärjestelmän mukana olevien tietojen ja ohjeiden on oltava suomen, ruotsin tai englannin kielellä. Tietojärjestelmää käyttävälle sosiaali- tai terveydenhuollon henkilöstölle tarkoitettujen tietojen ja ohjeiden on kuitenkin oltava suomen ja ruotsin kielellä. Lisäksi valmistajalla on oltava laatujärjestelmä, jota sovelletaan tietojärjestelmän suunnitteluun ja valmistukseen.” Sairaalakäyttöön kehittävillä järjestelmillä on täten lukuisia vaatimuksia, jotka näiden tulee täyttää todennettavasti. Perinteisten sovellusratkaisujen tulee toki noudattaa alueensa omia lakeja, kuten esimerkiksi GDPR:ää ja toteuttaa täten esimerkiksi oikeaoppista henkilötietojen suojaamista, mutta näille asetettujen vaatimusten kirjo on lähinnä asiakkaan vaatimuksia, kun taas sairaalaan kehitettävän sovelluksen perusvaatimukset määräävät asiakkaan lisäksi laki.

Sairaaloilla on lukuisia käyttäjiä ja sidosryhmiä, jotka tarvitsevat pääsyn dataan vuorokauden ympäri, joka tuo omia haasteita muun muassa palvelimien tehokkuudelle, huoltamiselle, ylläpidolle ja ongelmatilanteille. Pienempien sovellusratkaisujen päivitykset saatetaan voida toteuttaa aiheuttamalla katko palvelussa huollon ajaksi, mutta esimerkiksi sairaalan tulee päästä käsiksi potilasdataan myös huoltokatkojen aikana. Lisäksi sairaalojen toiminnan jatkuvuus tulee taata myös kriittisissä ongelmatilanteissa. Jotka kehitettävän järjestelmän tulee myös ottaa huomioon. Esimerkiksi tietokanta -palvelin numero 1:n vikatilanteessa, sairaalan data tulee olla varmuuskopioituna ja vara -palvelimen numero 2 tulee astua automaattisesti käyttöön, jotta toiminta voi jatkua ja sairaalan data on turvattu. Tällaisissa ongelmatilanteissa sairaaloilla on myös omat vaatimuksensa saatavan tuen nopeudesta ja laadusta. Sairaalojen koko, laajuus ja käyttäjämäärät kasvattavat myös datavirtaa ja verkkoliikennettä. Sairaalajärjestelmien resurssien minimivaatimukset ovat täten myös perinteisistä sovelluksista poiketen yleensä korkeampia, jotta sovelluksen käytettävyys pysyy stabiilina.

Karhusen mukaan sairaalaympäristöön kehitettävän sovelluksen kehitys eroaa perinteisestä sovelluskehityksestä siten, että sairaalaympäristöön kehitys on huomattavasti enemmän säänneltyä. Kehityksessä kehitys painottuu erityisesti tuotteen laatuun, projektin riskien hallintaan ja riskien minimointiin. (Karhunen 2020)

Kehityksessä kaikkein tärkeimmäksi elementiksi Karhunen toteaa potilasturvallisuuden. Tuotteen käytettävyyskin on korkealla tasolla, mutta erilaisia käytettävyyteen liittyviä asioita lähestytään nimenomaan riskien hallinnan kautta, potilasturvallisuus huomioiden. Erilaiset lääkinnälliset laitteet tulee CE-merkitä. ”CE-merkintä on merkintä, jolla tuotteen valmistaja tai valtuutettu edustaa vakuuttaa, että tuote täyttää tuotetta koskevien EU:n direktiivien ja asetusten olennaiset vaatimukset” (Tukes). Lisäksi tuotteet tulee tarvittaessa maakohtaisesti rekisteröidä, ennen niiden käyttöönottoa. Kehitysprosessit ovat tarkkaan

määriteltyjä, joiden noudattaminen on ehdotonta. Lopulliselle tuotteelle tulee lisäksi suorittaa kliininen arviointi, kunnes tuote voidaan lopullisesti julkaista. (Karhunen 2020)

5.1 Osto- ja kilpailutusprosessi

5.1.1 Asiakastarpeen syntyminen

Asiakkaalle tuotettavan tuotteen kehitysprosessi alkaa asiakastarpeesta. Tarve voi olla iso tai pieni, kokonaan uusi järjestelmäratkaisu tai pienempi olemassa olevan järjestelmän laajennus, joita voi olla muun muassa tuotteen toiminnallisuus, liityntä, uusi moduuli, analytiikka ja niin edelleen. Asiakastarpeet syntyvät sairaalan sisäisistä työnkulkuun liittyvistä seikoista tai ulkoisista herätteistä: asiakas voi esimerkiksi nähdä tai kuulla uudesta kiinnostavasta järjestelmästä, joka arvioidaan sisäisesti ja tästä syntyy asiakastarve, joka käynnistää itse hankintaprosessin. (Wahlström 2020)

Tarpeen syntyessä asiakkaan on itse määriteltävä hankittavan tuotteen tarve ja kriteerit, jotta yritykset kykenevät tarjoamaan asiakkaalle ongelmaa ratkaisevaa tuotetta. Hankinnan määrittelyn avulla asiakkaat pystyvät myös itse arvioimaan, kuinka hyvin heille tarjottu tuote vastaa juuri heidän tarvettansa ja vaatimuksia. Hankinnan määrittely toteutetaan yleensä sairaalan oman IT-osaston, osto-osaston ja loppukäyttäjien yhteistyössä. Joskus asiakkailla on selkeä ja hyvä käsitys siitä, mitä he tarvitsevat ja mitä on tarjolla ja kykenevät tekemään määrittelyn sisäisesti. Ennen hankinnan määrittelyä erilaisilla toimijoilla on kuitenkin mahdollisuus esitellä omaa tuotettaan ja niiden ominaisuuksia ja täten mahdollisesti vaikuttaa asiakkaan tekemän määrittelyn sisältöön. Joissakin tapauksissa asiakas voi julkaista virallisen tietopyynnön, jossa eri toimittajia kutsutaan vuoropuheluun, jossa asiakkaan on mahdollista kartoittaa tarjolla olevia ratkaisuvaihtoehtoja. Vuoropuhelu voidaan suorittaa kirjallisena vastauksena tai suorittamalla esittelytilaisuus asiakkaan valikoidulle arviointiryhmälle. (Wahlström 2020)

5.1.2 Sovelluksen vaatimusmäärittelyn teoria

Vaatimusmäärittelyn luominen on hankittavan tietojärjestelmän, sovelluksen tai ratkaisun perusedellytyksiä ja kulmakivi hankkeen onnistumiselle. Onnistuneen vaatimusmäärittelyn tuloksena projekti säästää kuluissa, nopeuttaa hankkeen kulkua ja takaa vaadittujen ominaisuuksien toimittamisen. Sen luomisella taataan, että toimittava osapuoli ymmärtää asiakkaan vaatimukset ja asiakas näkee, että heidän toiveensa on kuultu. Määrittelemällä vaatimukset luodaan perusta, miksi ja mitä tarpeita hankinnan tulee tyydyttää. Vaatimusmäärittely tehdään riippumatta lopullisen tuotteen käyttötarkoituksesta. (Kehmet Helsinki)

Toiminnallisuuksien priorisointia voidaan ajatella MVP:llä (Minimitoiminnallisuus, Minimum Viable Product), joka tarkoittaa riittävää toiminnallisuutta, jolloin palvelu voidaan ottaa käyttöön. Vaatimusmäärittelyä luodessa vaatimuksille annetaan tärkeystaso:

1 – Pakollinen: Vaatimuksen on pakko olla mukana ratkaisussa
2 – Hyödyllinen: Vaatimus tuottaa suuren hyödyn toiminnalle
3 – Toivottu: Vaatimus helpottaa toimintaa, muttei ole välttämätön

Tason 1 -vaatimukset muodostavat itsessään MVP:n, joita ilman palvelua ei voida ottaa käyttöön. Pakollisiin ei kuitenkaan sisälly ominaisuuksia, joita ilmankin palvelu on käytettävissä. Ominaisuudet tuotetaan, mutta tärkeys asettaa toteutusjärjestyksen vaatimuksille. Mikäli aikataulu on tiukka tai uhkaa ylittyä, 2- ja 3-tason vaatimuksia voidaan toteuttaa sovitusti myös tuotteen ylläpitovaiheessa. (Kehmet Helsinki)

Vaatimuksia voi olla kahdenlaisia, toiminnallisia ja ei-toiminnallisia. Toiminnalliset vaatimukset ovat järjestelmän käyttäytymisiä ja toiminnallisuuksia, joka määrittelee, miten ohjelmiston tai palvelun on käyttäydyttävä erilaisissa tilanteissa. Toiminnallisuuksia voi olla esimerkiksi: tilan tai ajan valinta, varatun ajan muuttaminen, toistuvien varausten teko tai tietyn tilan varauksen tekeminen luo myös varauksen toiseen tilaan. Ei-toiminnallisia vaatimukset ovat rajoituksia ja reunaehtoja, toiminnallisille vaatimuksille. Mitä ehtoja tuotteen on täytettävä, jotta toiminnallisia vaatimuksia voidaan ylipäätään toteuttaa. Tällaisia huomioon otettavia ei-toiminnallisia vaatimuksia voi olla esimerkiksi tietoturva ja tietosuoja, johon sisältyy esimerkiksi tunnistus- tai sisäänkirjaustoiminto, tai sovelluksesta kerättävien lokien sisältö. Vaatimusmäärittelyn osana tulee olla myös käytön ja ylläpidon järjestäminen. Mitkä ovat vaatimukset palvelun saatavuudelle ja luotettavuudelle ja mitkä ovat palvelun järjestämiselle asetetut vaatimukset (pilvipalvelu, puhelinpalvelu. (Kehmet Helsinki)

Projektin toteutus -kumppaneita kilpailutettaessa, asiakas luo itse oman vaatimusmäärittelynsä hankittavalla tuotteelle, omien tietojensa ja vaatimustensa pohjalta. Kun nämä ovat julkaistu, ei vaatimuksia voida kilpailutuksen osalta enää muuttaa. Projektin toteutusvaiheessa tehtyjä vaatimuksia voidaan tarkentaa projektin edetessä, mutta jos ne muuttavat projektin laajuutta, niin muutospyynnöt on vietävä muutoksenhallinnan kautta, mikä lisää projektin kokonaiskustannuksia. (Kehmet Helsinki)

5.1.3 Hankinnan aloittaminen

Kun tuotteen vaatimukset ovat selvillä, voidaan varsinainen tuotteen hankintaprojekti aloittaa. Kun asiakas on hankkimassa kokonaan uutta järjestelmää, tämän tapahduttava julkisen kilpailutusprosessin kautta, josta julkaistaan hankintailmoitus tähän tarkoitukseen tarkoitetulla alustalla. (Wahlström 2020)

Hankinnoista, jotka ylittävät EU-kynnysarvon, tulee ilmoittaa EU-laajuisesti. EU:n kynnysarvo määritellään seuraavasti rahallisena arvona: ”Sovelletaan vähintään kansallisen kynnysarvon (60.000 €, alv 0 %) suuruisiin tavara- ja palveluhankintoihin” (Rakennerahastot). Tarkemmat ehdot on määritelty laissa: Laki julkisista hankinnoista ja käyttöoikeussopimuksista (1397/2016). EU-ilmoituksissa hyödynnetään vakiolomakeasetuksessa asetettuja vakiolomakkeita. Erilaisia julkisia hankintailmoituksia jätetään erilaisin kriteerein muun muassa HILMA -palveluun ja SIMAP -palveluun, joista yleisempi on kuitenkin HILMA. HILMA:an julkaistaan julkisesti suoritettavat hankinnat ja SIMAP:iin muun muassa suorahankintaa koskevat hankintailmoitukset. Hankintayksikkö saa myös jättää hankintailmoituksen tai siihen liittyviä tietoja, tarkoituksenmukaiseen tiedotusvälineeseen, kuten sanoma- ja ammattilehtiin tai oman yrityksen verkkosivuille vasta kun hankintailmoitus on julkaistu HILMA -palvelussa. HILMA:ssa puolestaan tietoja ei voida julkaista ennen kuin tiedot julkaistaan EU:n virallisessa lehdessä. Hankkivan osapuolen on kyettävä tarvittaessa osoittamaan päivämäärä, jolloin se on jättänyt ilmoituksen julkaistavaksi. EU:n julkaisutoimisto toimittaa hankkivalle osapuolelle vahvistuksen ilmoituksen vastaanottamisesta ja tietojen julkaisupäivämäärän, joka toimii riittävänä osoituksena hankinnan julkaisemisesta. (Hankinnat.fi 2016)

Hankintailmoitus sisältää tietoja siitä, kuka on hankkija, mitä hankitaan, mikä on vastausaika ja mistä on saatavilla lisätietoja asiasta (hankinnassa olevat vaatimukset, pisteytykset ja tieto siitä, minne ja miten vastaus toimitetaan). Wahlström kertoo myös, että nykyään hankintaprosessissa käytetään lukuisia portaaleja, joihin tarjoajat voivat täydentää liitetiedostoja, yritysesittelyitä, tuotekuvauksia, hinnoitteluerittelyjä, alustavia projektisuunnitelmia ja niin edelleen. Julkisia hankintoja tehdessä tarjouskilpailussa esitetyt kysymykset ja vastaukset ovat julkisia tietoja kaikille tarjouskilpailuun osallistujille.
(Wahlström 2020)

5.1.4 Tarjouksen syntyminen

Asiakas käsittelee saapuneet tarjoukset asetettuun määräaikaan mennessä ja pisteyttää ne asettamiensa kriteerien mukaisesti. Tarjouskilpailusta yleensä valitaan yksi tarjoaja, jonka kanssa myös yleensä sopimus lopulta syntyy. Tarjouksen synnyttyä, varsinainen

tuotteen tai palvelun toimitus käynnistyy. Asiakkaalla ei ole kuitenkaan vaatimusta valita myöskään ketään tarjoajista tai voidaan suorittaa toinen arviointikierros valitsemalla muutama potentiaalinen tarjoaja. Tällaisia tarkkoja usean kierroksen valintaprosesseja on sairaalaympäristöissä nähty. (Wahlström 2020)

5.1.5 Suorahankinta

Mikäli tuotteen tai palvelun hankinta ei ylitä EU:n suorahankinnan kynnysarvoa tai suunniteltu hankinta on luonteeltaan sellainen, jonka vain yksi tietty toimittaja kykenee toimittamaan, kuten: jo olemassa olevan järjestelmän laajennus tai integraatioiden muutokset, voidaan tehtävä hankinta myös suorittaa suorahankintana. Tällöin asiakas suorittaa tarjous-hankintapäätös-menettelyn suoraan heidän valitseman toimittajan kanssa. Tällaisesta hankinnasta tulee myöskin julkaista julkinen suorahankintailmoitus HILMA:ssa, jossa asiakas perustelee, miksi päädyttiin suorahankintaan eikä julkiseen kilpailutukseen. Hankinta perustuu kuitenkin loppupeleissä samoihin vaiheisiin:

1. Asiakastarve
2. Vaatimusmäärittely
3. Toimittajan tekemä tarjous
4. Neuvottelu
5. Hankintapäätös
6. Sopimus
7. Toimitus

(Wahlström 2020)

5.1.6 Hintaerot perinteisiin sovellusratkaisuihin verrattuna

Perusperiaate, kuten kaikessa muussakin myynnissä on, ettei omien kustannusten alle myydä. Esimerkiksi kokonaisen järjestelmähankkeen toteutus on monien miljoonien eurojen kokoinen hanke. Yleisesti sairaalaan kehitettävät tuotteet ovat huomattavasti kalliimpia kuin esimerkiksi toimistoympäristöön tehtävä tuote, koska sairaalajärjestelmät vaativat useammin todella vaativaa ja tarkkaa asiakaskohtaista räätälöintiä. Perinteiset sovellukset ovat usein valmiita ”paketteja”, jotka voivat toimia suoraan, mutta sairaalatuotteet vaativat lähes poikkeuksetta asiakaskohtaista muokkaamista, joka nostaa tuotteen hintaa merkittävästi. Myös tavallisista sovelluksista poiketen sairaalaympäristön tuotteen vaativat tarkan validointi ja testausprosessin, jolla varmistetaan tuotteen vaadittava laatutaso. Kilpailu kuitenkin asettaa ratkaisujen hintakaton. Asiakas useimmiten valitsee halvimman ratkaisun, mutta tuotetta myytäessä on itse pystyttävä osoittamaan, miksi et itse nosta tuotteen hintaa kuten kilpailijat. (Kolu & Mäkeläinen 2020)

6 Sovelluskehitysprosessin ominaispiirteitä sairaalaympäristössä

6.1 Kehitys

Sovelluksen kehitysmenetelmät eivät sairaalaympäristöön ratkaisua tehtäessä eroa perinteisistä kehitysmenetelmämalleista. Kehityskaari kuitenkin seuraa tarkkaa riskienhallintaa ja dokumentointia korostaen potilasturvallisuutta tärkeimpänä elementtinä. Projektin käynnistyessä tuotteen kehitys aloitetaan portfolio -tason suunnitellulla yritystason prioriteettien mukaisesti. Tämän toteuduttua tästä kerätään laajasti asiakaspalautetta sairaaloista ja heidän käyttäjiltään, josta tehdään konseptitason analysointia ja suunnittelua. (Karhunen 2020)

Itse kehitys perinteisesti toteutetaan ketterien SAFe -menetelmien mukaisesti sprintti kerrallaan. Lopulta toteutettu softa verifioidaan ja validoidaan käyttäjä- ja softavaatimuksia vasten. Kaiken lisäksi toteutetaan ”summatiiviset käytettävyystestit”, joissa varmistetaan, että tuote on turvallinen käyttää. Tuote voidaan esimerkiksi toimittaa sairaalaan testattavaksi. Testeistä kerätään laajaa palautetta sairaalahenkilökunnalta ja tarvittaessa toteutetaan muutoksia näiden perusteella, ennen tuotteen lopullista julkaisua. (Karhunen 2020)

6.1.1 SAFe -menetelmät

The Scaled Agile Framework (SAFe) on ketterien menetelmien rakenne, jossa sovelluskehitys -tiimit muodostetaan käyttäen kolmea peruspilaria: tiimi, ohjelma ja portfolio. SAFe:lla pyritään saamaan kehitysprosessiin joustavuutta ja ratkaista suurempien organisaatioiden ongelmia koskien ketterien menetelmien käytössä. Se tarjoaa hyvän tietopohjan parhaista käytännöistä, joita oikeat kehitystiimit ovat hyödyntäneet onnistuneiden tuotteiden kehityksessä. (ProductPlan)

SAFe -metodologian hyötyjä on erilaisten tiimien yhteistoiminnan tehostus, hyvä organisaatiotason läpinäkyvyys ja projektin eri näkökulmien parempi kohdentaminen suurempiin liiketoiminnan tavoitteisiin. SAFe:a ei kuitenkaan pidetä 100 % ketteränä, sen laajan suunnitteluvaatimuksien takia. SAFe vaatii myös enemmän ”ylhäältä-alas katsomista”, kuin tiimipohjaista lähestymistapaa. (ProductPlan)

6.1.2 Security by Design

Security by Design – on sovelluskehityksen lähestymistapa, jossa tietoturvaan liittyvät tekijät otetaan huomioon jo kehityksen alkuvaiheessa. Sen tarkoituksen on estää tietoturvamurtoja ennen kuin ne edes tapahtuvat. Security by Design:ssa sovelluskehittäjät huomioivat ulkoiset uhat siten, että on epätodennäköisempää, että lähdekoodista löytyisi sellaisia haavoittuvuuksia, joita voitaisiin hyödyntää järjestelmään murtautumisessa tai väärinkäytössä. Jotta Security by Design:a voitaisiin hyödyntää tehokkaasti, on yrityksen ja kehittäjän ymmärrettävä ja kartoitettava koko tuotteen elinkaari tietoturvan perspektiivistä. Se vaatii, että tuotetta jatkuvasti hallitaan, monitoroidaan ja ylläpidetään, jotta se voisi pysyä tietoturvallisena jatkossakin. Tietoturvan suunnittelu tuotteen alusta lähtien on tärkeää, sillä on vaikeampaa lisätä tuotteen tietoturvaa ja paikata haavoittuvuuksia jälkikäteen, kuin kehityksen jo ollessa käynnissä ja näiden huomioimisen kehitysvaiheessa. Myös reaaliaikainen haavoittuvuuksien paikkaaminen on vaikeampaa. Mikään jälkikäteen suoritettu korjaus ei tule olemaan koskaan niin tehokas, kuin alusta asti tietoturva-asiat huomioon ottanut kehitys. (Reciprocity 2020)

6.2 Tuotteen erityispiirteet

Sairaalat toimivat vuorokauden ympäri ja henkilökunnan vaihtuvuus voi olla tiuhaa. Karhunen luettelee sairaalaympäristön tuotteiden keskinäisiksi ominaisuuksiksi tuotteen helppokäyttöisyyden, toimintavarmuuden, saatavuuden vuorokauden ympäri ja erilaisten käyttäjäryhmien ja käyttöolosuhteiden huomioinnin. Lisäksi tietoturvan tulee olla huippuluokkaa sillä järjestelmä sisältää yleensä potilastietoja, joten keskeisiksi tekijöiksi muodostuvat myös käyttöoikeuksien hallinta, joita ovat muun muassa audit trail ja kyberturvallisuuteen liittyvät seikat. (Karhunen 2020)

6.2.1 Audit Trail

Audit trail:it ovat turvallisuuden ja yksityisyyden kannalta kriittisiä komponentteja, joilla voidaan monitoroida ja dokumentoida sovelluksen tietoa siitä, kuka tarkalleen tietoja on tarkastellut tai muokannut. Hyvällä audit trail:llä parannetaan organisaation tietoturvaa ja yksilön yksityisyyden suojaa. Hyvän audit trail -toteutuksen tulisi sisältää vähintään seuraavia ominaisuuksia:

- Datan luonnin, muokkauksen tai poiston aikaleimaukset
- Aikaleimaus perustuu aikaan, jota ei voida muokata
- Audit trail:in sisältämä data säilytetään siten, että sen muokkaaminen tehdään mahdottomaksi millä tahansa käyttäjätasolla

- Tieto siitä, kuka dataa on tarkastellut
- Audit trail säilyttää kopion alkuperäisestä datasta, jota tarkasteltiin, muokattiin tai poistettiin

Audit trail:llä mahdollistetaan datan käytön seurantaa ja pystytään helposti jäljittää tiedon väärinkäyttäjät, sillä kaikesta mitä järjestelmässä tehdään, jää konkreettinen jälki. (Fair-Warning 2020)

Koska sairaalassa käsitellään todella arkaluontoista dataa, on kyettävä jäljittämään, kuka sitä käsittelee. Audit trail on tästä syystä oiva tapa hallinnoida ja monitoroida kaikkia sovelluksen tärkeitä tapahtumia. Myös mahdollisesti virheellisten kirjauksien korjaaminen on helpompaa, sillä alkuperäisestä kirjauksesta jää audit trail:iin jälki. Tämä on täten potilaan hoidon ja tietoturvan kannalta oleellinen ”turvamekanismi”.

6.3 Testaus ja validointi

Sairaalaan käyttöönotettavien tuotteiden on oltava todennetusti luotettavia ja täten niiden toiminta on testattava ja validoitava huolellisesti. Tuotteita testataan mahdollisimman kattavilla automaatiotestauksilla, joita on esimerkiksi unit- ja BDD-tason testit. Osan testeistä saa suorittaa vain ja ainoastaan kliiniset asiantuntijat, sekä testauksessa käytettävät työkalut ovat aina validoituja. Lisäksi osana testausta suoritetaan erilaisia formatiivisia ja summatiivisia käytettävyystestejä, sekä suorituskykytestejä. Itse sovelluskehitystiimi testaa sovelluksen lähdekoodia staattisilla analysointityökaluilla, sen laadun varmistamiseksi ja lisäksi kyberturvallisuuteen liittyen suoritetaan lukuisia testejä ja koodin skannauksia. (Karhunen 2020)

Tärkeänä osana testauksia on myös sovelluksen vakaus. Sovellukset ovat jatkuvassa käytössä vuorokauden ympäri, joten näiden on toimittava vaadittavalla tavalla. SLA-sopimuksissa määritellään tarkasti, mikä on oltava sovelluksen jatkuva saatavuus vikatilanteista huolimatta. (Kolu & Mäkeläinen 2020)

6.3.1 Unit – ja BDD -testaus

Unit -testaus eli yksikkötestaus, on kooditason testausta, jossa sovelluksen eri testattavia osia (yksikköjä) testataan yhdessä ja erikseen. Testauksia suorittaa tuotteen kehittäjät tuotteen kehitysvaiheessa. Sen tarkoituksena on selvittää, että lähdekoodi todella toimii niin kuin sen on tarkoituskin. (Rouse 2019)

BDD -testaus (Behavior-driven development) on malli, jolla voidaan testata lähdekoodin toiminnallisuuksia ja haluttuja vaikutuksia. BDD:ssä hyödynnetään kolmen sanan mallia: ”Given-when-then”, jota käytetään kuvaustyökaluna, joka kuvastaa ”keskustelua” sovelluksessa.

- Given: Käyttäjä on profiilissaan.
- When: Käyttäjä päivittää syntymäpäivänsä profiiliin.
- Then: Käyttäjän syntymäpäivä esitetään profiilikuvan alapuolella.

BDD -testauksessa sovelluksen keskustelua kuvataan lukuisin skenaarioin ja varmistetaan, että sovelluksen toiminnallisuudet toimivat tarkoituksen mukaisesti. (Nest 2020)

6.3.2 SLA -sopimukset

SLA – eli service-level agreements (palvelutasosopimukset) ovat sopimuksia, jotka määrittelevät tarkasti tuotettavan palvelun tason. Ne ovat ulkoistamisen kannalta kriittisiä, sillä ne määräävät palvelun saatavuuteen, tyyppiin ja laadulle asetetut kriteerit. Sopimus sisältää vaatimuksen siitä, minkä prosenttiosuuden ajasta ostettu palvelu on oltava saatavilla vikatilanteista huolimatta. Se määrittelee lisäksi ylläpidolliset vastuualueet ja niiden toteuttajat. Sopimuksella on tarkoitus suojella molempia (toimittaja ja ostaja) osapuolia riitatilanteessa ja määritellä selkeästi palveluntuottajalta vaaditut tehtävät. (Overby 2017)

Palvelutasosopimukset sisältävät tiedon asiakkaan ja palveluntarjoajan välisistä yhteisistä pelisäännöistä, termistöistä, käytettävistä mittareista sekä tavoitteista. Palvelutasosopimuksien toteutumista mitataan erilaisin mittarein ja vaaditun tason alittuessa palveluntarjoajalle seuraa asiasta usein sanktio. Palvelutasosopimusten toteutumista seurataan jaksoittain tapahtuvissa seurantapalavereissa asiakkaan ja toimittajan kesken. Sopimuksissa jaetaan asiakkaan ja palveluntarjoajan roolit ja vastuualueet, joiden toteutumisella pyritään lopulta yhteisen päämäärän saavuttamiseen. (Hovila 2015)

6.4 Toiminnan jatkuvuus

6.4.1 Vikatilanteet

Kuten aikaisemmin kerrottiin, SLA-sopimukset määrittelevät palvelun tai sovelluksen saatavuuden ja siihen liittyvien ylläpidollisten vastuualueiden osapuolet. Toiminnallisuuden jatkuvuutta voidaan parantaa erinäisin keinoin, joilla parannetaan järjestelmän saatavuutta, joka puolestaan tukee SLA-sopimuksia. Asetetut vaatimukset riippuvat suoraan sairaalasta, sen koosta, järjestelmän luonteesta ja sen kriittisyydestä suhteessa potilaan hoitoon. (Kolu & Mäkeläinen 2020)

Esimerkiksi, mikäli serveri, jossa käytettävää tuotetta tai palvelua ajetaan, on virtualisoitu, voidaan virtuaaliympäristöstä tehdä jatkuvaa kopiota toiseen fyysiseen sijaintiin. Ongelmatilanteen sattuessa pääserverillä tai jopa sen hajotessa, on tieto varmuuskopioituna identtisesti toisessa sijainnissa. Myös tilapäisissä laitteistotason vioissa, joissa esimerkiksi pääpalvelimen kovalevy hajoaa, osaa järjestelmä kytkeytyä varajärjestelmään automaattisesti, kun se huomaa pääpalvelimen ongelmatilanteen. Täten sairaala kykenee jatkamaan toimintaansa myös vikatilanteissa ilman pidempiä katkoja. Vastaavat turvatoimet ovat kuitenkin parhaimmillaan satojen tuhansien eurojen arvoisia ja näiden hankinnasta päävastuussa on itse laitteiston omistaja, eli sairaala. Osa sairaaloista ei myöskään halua tietoturvallisista syistä luovuttaa edes varmuuskopiointi -tarkoituksessa arkaluontoista dataa säilytykseen kolmannelle osapuolelle, vaan toteuttavat tarvittavat turvatoimet paikallisesti. Kolmannelle osapuolelle varmuuskopioinnin vastuun siirtäminen tietyin ehdoin on kuitenkin lisääntyvä trendi. (Kolu & Mäkeläinen 2020)

6.4.2 Windows-klusterointi

Käytettävyyttä ja palvelun saatavuutta voidaan lisätä sairaalan toiminnan kannalta kriittisille sovelluksille yhdistämällä erillisiä palvelimia, jotka varmistavat palveluiden saatavuuden myös vikatilanteissa. Palveluita tai kokonaisia palveluita voidaan tarvittaessa monistaa, siten että siitä saadaan toinen tai useampi täydellinen kopio. Klusteriin kuuluvat jäsenet voivat täten jatkaa toimintaansa ja jatkaa loppukäyttäjän palvelun tuottamista, vaikka yhden palvelimen toiminta lakkaisikin kokonaan. Sairaalan toimintaa vikatilanteissa voidaan tukea huomattavasti klusteroinnin tuottamisella. Tällaisessa Failover -klusterissa liitettynä on usein vähintään kaksi palvelinta, joista toiseen yhteyden katketessa syystä X, voidaan automaattisesti siirtyä käyttämään toista identtistä palvelinta usein ilman, että palvelun käyttäjät tätä edes havaitsevat. Failover -operaatio voidaan käynnistää myös manuaalisesti ja täten ohjata palvelu käyttämään haluttua palvelinta esimerkiksi huoltokatkon aikana. (Kolu 2016, 27-28)

6.4.3 Tukipalvelut

Tilapäisien ongelmatilanteiden lisäksi järjestelmissä voi sattua kriittisempiä vikatilanteita, joka estää koko järjestelmän toiminnan ja täten hidastaa radikaalisti koko sairaalan toimintaa. Sairaalat varautuvat tällaisiin vikatilanteisiin järjestämällä sisäisiä ja ulkoisia tukipalveluita. Näidenkin saatavuus riippuu eniten järjestelmän kriittisyydestä, sairaalasta ja sen aiheuttamista kuluista. Sairaala voi tarvittaessa esimerkiksi maksaa palvelun toimittajalle 24h tuen järjestämisestä tai ottaa tuen vain ”toimistoaikojen” aikana. Tarkemmat määrittelyt tuen järjestämisestä ja vastuu osapuolista on määritelty tarkasti SLA-sopimuksissa. (Kolu & Mäkeläinen 2020)

6.4.4 Päivitykset

Jotta sovellukset ja palvelut pysyvät ajan tasalla, on näitä päivitettävä tietyin väliajoin. Päivityksillä korjataan sovelluksessa huomattuja vikoja, tehdään parannuksia ja räätälöidään tuotteita asiakkaiden vaatimusten mukaisiksi. Jatkuvasti käytössä olevan kriittisen järjestelmän päivitykset on syytä suunnitella tarkasti etukäteen. Päivitysten toteuttaminen riippuu täysin sen aiheuttaman katkon pituudesta ja sairaalasta. Lyhyet katkot voidaan ilmoittaa etukäteen ja suorittaa palvelukatkona, jolloin sairaala hyödyntää muuta tietojen kirjaustapaa kuten perinteiset paperilomakkeet, jotka tuodaan katkon päätyttyä järjestelmään. Pitkissä katkoissa asiakkaille tulee järjestää vaihtoehtoinen sähköinen tietojen kirjaustapa. Päivitykset on syytä ajoittaa aikaan, jolloin sairaalan toiminta on normaalia pienempää, kuten yöt ja viikonloput. Sairaaloissa on alkanut myös lisääntyvä trendi ”0 - downtime”, joka tarkoittaa nimensä mukaisesti 0:aa katkoa. Tällaisissa tilanteissa päivitykset tulee pystyä toteuttamaan järjestelmän ollessa käytössä. (Kolu & Mäkeläinen 2020)

7 Päätelmät

ICT-ratkaisujen toteutus sairaalaympäristöihin ei eroa sovelluskehityksessä käytettäviltä metodeilta ja käytännöiltään ei-niin-kriittiseen ympäristöön tuotettavilta toteutuksilta. Kuitenkin sairaalaan kehitettäessä kehittäjän on noudatettava normaalista poiketen suurta määrää erilaisia lakeja ja säädöksiä, sekä suoritettava tarkkaa validointia ja testausta erilaisten sidosryhmien ja ammattilaisten tiiviissä yhteistyössä. Sairaalan kehitystyössä korostuu normaalia enemmän erilaisten dokumentointien ja suunnitteluiden tärkeys, turvallisuus, sekä järjestelmän monipuolisuus. Täten kiireessä tehdyllä, tai huonosti suunnitelulla sairaalatuotteella on suuri riski epäonnistua ja menettää maineensa jo kehityksen alkuvaiheessa. Sovelluskehittäjän tulee jo kehityksen alkuvaiheessa noudattaa hyviä toimintamalleja ja tehdä taustatyönsä kehitettävän epäkohdan ja ratkaisun vaikutuksesta koko sairaalan toimintaan ja potilasturvallisuuteen.

Potilasturvallisuus on kehityksen keskiössä ja tulee huomioida kehityksen jokaisessa vaiheessa siten, että se myös pidetään aina tärkeimpänä elementtinä. Sairaalaratkaisuissa vaaditaan perinteisiä sovellutuksia selkeästi enemmän taustatyötä ja kokemusta sairaalan omista toimintatavoista ja toimintaketjuista, jotta kehittävä tuote voi näitä aidosti tehostaa ja tuoda hyötyä käyttäjilleen. Henkilön, joka ei omaa kokemusta sairaalaympäristössä toimisesta on lähes mahdotonta toteuttaa minkäänlaista tuotetta onnistuneesti. Tuotteiden on oltava luonteeltaan monipuolisia, että ne ratkaisevat useita toimintaketjujen epäkohtia, eivätkä vain yhtä. Ne on oltava liitettävissä muihin olemassa oleviin järjestelmiin, mikä lisää integraatioiden tärkeyttä ja sulautuvuutta sairaalan järjestelmäkokonaisuudeksi. Sairaalat voidaan luokitella omiksi yksilöiksi, joilla kaikilla on omat eroavat toimintamallinsa ja järjestelmänsä, jolloin myös kehittävän tuotteen tulee pystyä olla räätälöitävissä jokaisen sairaalan omiin tarpeisiin ja integroitavissa jo käytössä oleviin järjestelmiin.

Laki, sopimukset ja asiakkaat asettavat tuotteen luotettavuudelle määrätyt kriteerit. Ratkaisut ovat usein luonteeltaan kriittisiä, koska ne myös sairaalan toiminnan kannalta ovat sen välttämättömiä peruselementtejä. Näiden tuotteiden tuki ja varotoimet on järjestettävä huolellisesti, jotta potilaan hoitoa voidaan jatkaa mahdollisimman katkotta. Sairaalaratkaisujen hinnat ovat korkeat, mutta tämä johtuu niiden jatkuvasta räätälöinnistä, monimutkaisuudesta, liitettävyydestä, korkeasta laadusta ja vaativasta testausprosessista. Järjestelmäratkaisujen korkeasta hinnasta huolimatta ne tukevat sairaalan toimintaa merkityksellisesti, joita ilman sairaala toimisi tehottomasti, eikä kykenisi vastaamaan hoidon kysyntään, joka Suomessa kasvaa jatkuvasti suurten ikäluokkien takia. Myös suuret pandemiat kuten korona vuonna 2020 ovat lisänneet sairaalan hoitopainetta ja täten kasvattaneet

sähköisten järjestelmien merkitystä sairaalassa entisestään. Sairaalaratkaisuja kehittämällä säästetään sairaalan hoito- ja ylläpitokustannuksissa pitkällä aikavälillä, taataan tulevaisuudessa entistäkin monipuolisempi terveydenhuolto ja kyetään vastaamaan entistäkin suurempaan kysyntään, saadaan potilaasta monipuolisempaa dataa ja luodaan uusia innovatiivisia keinoja toteuttaa samaa ennalta tuttua laadukasta suomalaista terveydenhuoltoa.

8 Lähteet

Aalto M. 2020. Suomen suurin sairaalakeskittymä siirtyy ensi yönä Apotti-aikaan – Näin alkukaaoksen aiheuttanut järjestelmä on toiminut.
Luettavissa: https://www.hs.fi/kaupunki/art-2000006704986.html?share=165d0d3b9a04867e68fb462d971c10be. Luettu 31.10.2020

Agilealliance. Agile 101.
Luettavissa: https://www.agilealliance.org/agile101/. Luettu 05.10.2020

Ali Babar, M & Brown, A & Mistrik, I. 2013. Agile Software Architecture. Aligning Agile Processes and Software Architectures. Morgan Kaufmann

Apotti. Apotti järjestelmänä.
Luettavissa: https://www.apotti.fi/apotti-jarjestelmana/. Luettu 30.09.2020

Apotti. 2018. Apotin käyttöönotto alkaa 10. marraskuuta Peijaksen sairaalassa.
Luettavissa: https://www.apotti.fi/apotin-kayttoonotto-alkaa-10-marraskuuta-peijaksen-sairaalassa/. Luettu 12.11.2020

Arrow ECS Finland. 2017. Arrow Refenssitarina Kaunialan Sairaala. Youtube video.
Katsottavissa: https://youtu.be/bG9XbKhkYz8. Katsottu 30.09.2020

Blogspot. 2020. How to Choose The Best Hospital Software For Business Success?
Luettavissa: https://ehmspro.blogspot.com/2020/01/how-to-choose-best-hospital-software.html. Luettu 07.10.2020

Dräger. Rinnallasi sairaalatoiminnassa.
Luettavissa: https://www.draeger.com/fi_fi/Hospital. Luettu 30.09.2020

Euroopan Unioni. 2020. Yleinen tietosuoja-asetus.
Luettavissa: https://europa.eu/youreurope/business/dealing-with-customers/data-protection/data-protection-gdpr/index_fi.htm. Luettu 01.10.2020

FairWarning. 2020. What Is an Audit Trail, and Why Is It Important?
Luettavissa: https://www.fairwarning.com/insights/blog/what-is-an-audit-trail-and-why-is-it-important. Luettu 29.10.2020

Fimea. Lääkinälliset laitteet.
Luettavissa: https://www.fimea.fi/laakinnalliset_laitteet. Luettu 30.09.2020

FrontEnders. 2016. Information Communication Technology in HealthCare.
Luettavissa: https://www.frontenders.in/blog/information-communication-technology-healthcare.html. Luettu 01.10.2020

GE Healthcare. 2019. GE Healthcare High Acuity Critical Care. YouTube video.
Katsottavissa: https://youtu.be/4SiaSUKL8bU. Katsottu 30.09.2020

Hankinnat.fi. 2016. Hankintailmoitusten julkaiseminen.
Luettavissa: https://www.hankinnat.fi/eu-hankinta/ilmoittaminen/hankintailmoitusten-julkaiseminen. Luettu 27.10.2020

Helsingin Yliopisto. 2009. Ohjelmistoprosessit ja ohjelmistojen laatu.
Luettavissa: https://www.cs.helsinki.fi/u/taina/opol/k-2009/pdf/luku-6_2.pdf.
Luettu 01.10.2020

Herrrick M. 2016. How to Start Your First Agile Project Using Scrum Methodology.
Luettavissa: https://www.dragonspears.com/blog/agile-project-using-scrum-methodology.
Luettu 17.10.2020

Hovila J. 2015. Palvelutasosopimus – tärkein asia markkinoinnin ja myynnin integraatiossa. Luettavissa: https://www.powermarkkinointi.com/blogi/palvelutasosopimus-tarkein-asia-markkinoinnin-ja-myynnin-integraatiossa. Luettu 14.11.2020

HUS. Osastot.
Luettavissa: https://hus.fi/sairaanhoito/sairaalat/meilahden-tornisairaala/osastot/Sivut/default.aspx. Luettu. 30.09.2020

Itemis. Scrum Process.
Luettavissa: https://www.itemis.com/en/agile/scrum/compact/introduction-to-scrum/scrum-process. Luettu 17.10.2020

JAMK. ICT-ala.
Luettavissa: https://www.jamk.fi/fi/Koulutus/ICT-ala/.
Luettu 15.11.2020

Javatpoint. V-Model.
Luettavissa: https://www.javatpoint.com/software-engineering-v-model. Luettu 05.10.2020

Junko T. 2018. Tulevaisuuden sairaala nojaa terveysteknologiaan ja digitalisaatioon.
Luettavissa: https://www.auroralehti.fi/tulevaisuuden-sairaala-nojaa-terveysteknologiaan-ja-digitalisaatioon/. Luettu 08.10.2020

Laki sosiaali- ja terveydenhuollon asiakastietojen sähköisestä käsittelystä.
Luettavissa: https://www.finlex.fi/fi/laki/ajantasa/2007/20070159. Luettu 01.10.2020
Lääkäriliitto. Tehohoito.
Luettavissa: https://www.laakariliitto.fi/laakarinetiikka/hoidon-erityiskysymyksia/tehohoito/. Luettu 14.11.2020

Karhunen, M. 22.10.2020. Director - Software Engineering. GE Healthcare. Sähköposti-haastattelu.

Kehmet Helsinki. Vaatimukset (perinteinen hanke)
Luettavissa: https://kehmet.hel.fi/menetelmalaari/vaatimukset/. Luettu 08.10.2020

Keski-Suomen sairaanhoitopiiri. 2016. ICT-ratkaisut.
Luettavissa: https://www.ksshp.fi/fi-FI/Sairaanhoitopiiri/Uusi_sairaala_projekti/ICTratkaisut. Luettu 30.09.2020

Kolu T. 2016. Korkean käytettävyyden tekniikoiden hyödyntäminen tehohoidon ja anestesiologian tietojärjestelmissä. Pro gradu -tutkielma, Jyväskylän yliopisto. Luettavissa: https://jyx.jyu.fi/bitstream/handle/123456789/50829/URN%3aNBN%3afi%3ajyu-201607133602.pdf?sequence=1&isAllowed=y. Luettu 31.10.2020

Kolu, T. I&U Coordinator. & Mäkeläinen, A. Staff Customer Support Team Manager. 2020. GE Healthcare. Haastattelu. Microsoft Teams.

Kotian N. 2017. Agile Product Development Methodology.
Luettavissa: https://medium.com/@niteshk451/agile-product-development-methodology-2d23cd41c8fc. Luettu 07.10.2020

Meteoriitti. Ketteryys haltuun: Scrum pähkinänkuoressa.
Luettavissa: https://meteoriitti.com/2013/06/06/ketteryys-haltuun-scrum-pahkinankuoressa/. Luettu 15.10.2020

Mikrobitti. 2020. Apotti saa kritiikkiä, vanhat järjestelmät halutaan takaisin kriisin ajaksi – ”Jos meillä on Apotti ja korona, niin kuolleisuus lisääntyy”.
Luettavissa: https://www.mikrobitti.fi/uutiset/apotti-saa-kritiikkia-vanhat-jarjestelmat-halutaan-takaisin-kriisin-ajaksi-jos-meilla-on-apotti-ja-korona-niin-kuolleisuus-lisaantyy/8793dfc5-8f99-4458-a321-93c802b131fc. Luettu 09.10.2020

Neha T. 2020. Spiral Model.
Luettavissa: https://binaryterms.com/spiral-model.html. Luettu 05.10.2020

Nest K. 2020. BDD in Action: Dissecting the Converstaion.
Luettavissa: https://blog.gurock.com/bdd-in-action-conversation/?utm_source=adwords&utm_medium=cpc&utm_campaign=europe_en_dsa&utm_content=&creative=455532429356&keyword=&matchtype=b&network=g&device=c&gclid=Cj0KCQiAnb79BRDgARIsAOVbhRqvfcgu0AykcIIt0a3kqpxmoeBoMiBdV1mipmXwmRLY2HKwbhcX9J4aAk1HEALw_wcB. Luettu 14.11.2020

Niemi T. 2019. Iso päätös edessä – Apotti maksaisi Nurmijärvelle 24,5 miljoonaa.
Luettavissa: https://www.nurmijarvenuutiset.fi/paikalliset/1281904. Luettu 09.10.2020

Nironen S. 2020. Keusote ei lähde mukaan Apottiin.
Luettavissa: https://yle.fi/uutiset/3-11309663. Luettu 09.10.2020

Overby S. 2017. What is an SLA? Best practices for service-level agreements.
Luettavissa: https://www.cio.com/article/2438284/outsourcing-sla-definitions-and-solutions.html. Luettu 31.10.2020

Pikkarainen A. 2020. HS: Apotti-potilasjärjestelmä johti kuolemaan Helsingissä – vaaratilanteita yli 50. Luettavissa: https://www.iltalehti.fi/kotimaa/a/7bccadac-21af-4ba8-a04c-d50f9d5b6747. Luettu 09.10.2020

Palmén T. 2020. Näin koronapotilaita hoidetaan teho-osastolla – osaa potilaista pidetään enimmäkseen mahallaan: "30-50 prosenttia kuolee tehohoidosta huolimatta".
Luettavissa: https://www.mtvuutiset.fi/artikkeli/nain-koronapotilaita-hoidetaan-teho-osastolla-osaa-potilaista-pidetaan-enimmakseen-mahallaan-30-50-prosenttia-kuolee-tehohoidosta-huolimatta/7790584#gs.i371e6. Luettu 07.10.2020

ProductPlan. Scaled Agile Framework.
Luettavissa: https://www.productplan.com/glossary/scaled-agile-framework/.
Luettu 29.10.2020

Radigan D. The product backlog: your ultimate to-do list.
Luettavissa: https://www.atlassian.com/agile/scrum/backlogs. Luettu 17.10.2020

Rakennerahastot. Hankintaklinikka.
Luettavissa: http://www.rakennerahastot.fi/documents/13596/1258652/Uusi+hankinta-laki+-+Hankinnat+ja+dokumentointi/654fdd18-ac23-414e-a2a6-62a3a89c9ef3?version=1.1. Luettu 15.11.2020

Rawsthrone D. Sprint length: What Length is the Right Length?
Luettavissa: https://3back.com/scrum-tips/sprint-length-what-length-is-the-right-length/.
Luettu 17.10.2020

Reciprocity. 2020. What is Security by design?
Luettavissa: https://reciprocitylabs.com/resources/what-is-security-by-design/.
Luettu 14.11.2020

Rouse M. 2019. Unit testing.
Luettavissa: https://searchsoftwarequality.techtarget.com/definition/unit-testing.
Luettu 14.11.2020

Salo S. 2019. Sujuvuuttaa sairaalaan digitaalisilla ratkaisuilla.
Luettavissa: https://professio.fi/sujuvuutta-sairaalaan/. Luettu 30.09.2020

Scrum.org. What is a Scrum Development Team?
Luettavissa: https://www.scrum.org/resources/what-is-a-scrum-development-team.
Luettu 17.10.2020

Scrum.org. What is a Product Owner?
Luettavissa: https://www.scrum.org/resources/what-is-a-product-owner. Luettu 17.10.2020

Scrum.org. What is a Scrum Master?
Luettavissa: https://www.scrum.org/resources/what-is-a-scrum-master. Luettu 17.10.2020

TAYS. 2020. Tehohoito.
Luettavissa: https://www.tays.fi/fi-fi/palvelut/tehohoito. Luettu 07.10.2020

Terveyskylä. 2017. Nukutus eli yleisanestesia.

Luettavissa: https://www.terveyskyla.fi/leikkaukseen/yleistietoa/nukutus-ja-puudutus/nukutus-eli-yleisanestesia. Luettu 08.10.2020

Thinking portolio. 2016. Projektien vesiputousmalli ja sen viisi heikkoutta.
Luettavissa: https://thinkingportfolio.com/projektien-vesiputousmalli-ja-sen-viisi-heikkoutta/. Luettu 01.10.2020

THL. 2019. Mikä on THL?
Luettavissa: https://thl.fi/fi/thl/mika-on-thl. Luettu 14.11.2020

Tukes. CE-merkintä.
Luettavissa: https://tukes.fi/tuotteet-ja-palvelut/ce-merkinta. Luettu. 27.10.2020

Tutorialspoint. SDLC – Iterative Incremental Model.
Luettavissa: https://www.tutorialspoint.com/adaptive_software_development/sdlc_iterative_incremental_model.htm. Luettu 05.10.2020

Valvira. 2020. Sosiaali- ja terveydenhuollon tietojärjestelmät.
Luettavissa: https://www.valvira.fi/terveydenhuolto/sosiaali-ja-terveydenhuollon-tietojarjestelmat. Luettu 30.09.2020

9Solutions. 2018. Kokemuksia 9Solutions Oy:n hoitajakutsujärjestelmästä. Youtube-video.
Katsottavissa: https://www.youtube.com/watch?v=n55j3Ljsp1c. Katsottu. 08.10.2020

Vantaa. Apotti-hanke.
Luettavissa: https://www.vantaa.fi/hallinto_ja_talous/organisaatio/sosiaali-_ja_terveydenhuollon_toimiala/apotti-hanke. Luettu 14.11.2020

Wahlström, N. 2020. Digital Sales Direct. GE Healthcare. Sähköpostihaastattelu.

Liitteet

Liite 1. Ammattilaisille esitettyjä kysymyksiä (suomeksi)

Miten perinteinen sovelluskehitys ja sovelluskehitys sairaalaympäristöön eroavat?

- Millaisia vaiheita sairaalaan tarkoitetun sovelluksen kehitys pitää sisällään?
- Kuinka tuote validoidaan toimivaksi?
- Millaisia erityispiirteitä sairaalaympäristöön kehitettävällä tuotteella tulee olla?
- Miten sairaalaan kehitettyä tuotetta voidaan testata turvallisesti?

Osaatko kertoa, millainen on sairaalatoimijoiden sairaalajärjestelmien osto- ja kilpailutusprosessi?

Millaisia toiminnallisuuden jatkuvuuden takaavia keinoja sairaalassa on vikatilanteissa? Esimerkiksi, jos tietokantapalvelin lakkaa toimimasta, miten toiminta jatkuu, onko tämä huomioitu potilastieto- ja muissa järjestelmissä tai niiden vaatimuksissa?

Millaiset tukitoimet sairaalalla tulee olla kriittisissä sovelluksen vikatilanteissa?

Millaiset vaatimukset sairaalassa käytettävän sovelluksen vakaudelle on asetettu?

Miten sairaalaympäristössä tulee suunnitella päivitysten suorittaminen, miten sairaala jatkaa toimintaa päivityksen aikana?

Millaisia hintaeroja sairaalajärjestelmäratkaisujen ja perinteisten järjestelmäratkaisujen välillä on ja mistä hintaerot johtuvat?

Liite 2. Ammattilaisille esitettyjä kysymyksiä (englanniksi)

How traditional software development and development to hospital environment differ?

- What steps need to be considered while developing a software to hospital environment?
- Which special features a hospital software must have compared to a “regular” software?
- How can the hospital software be tested safely?
- How can the final product be validated and approved as compliant?

Can you tell from IT company perspective, how procurement and competition of offers is completed by hospitals acquiring patient information software?

What -contingency plans or continuity actions of operations patient information systems in hospitals have and is this considered at the information system level or in the system requirements? How is the patient information system protected from faults, power outages or critical errors? For example, if database -server stops working suddenly-, how does the information system and the hospital continue its operations?

What kind of IT support (people resources) hospitals have in place for a critical software failure and is this covered at the information system level, in the system requirements or just as additional purchased service?

What kind of standards (ISO standards, Best Available Techniques etc.) information software has in place in hospitals, considering stability, contingency or system restore?

How physical software updates are planned in practice, so that the hospital and the patient information system may continue its functions without disturbance?

What price differences hospital information systems and traditional systems have, and what factors are causing the difference?

Liite 3. Ohjelmoijan check -list

Tämä dokumentti kiteyttää sovelluskehittäjän näkökulmasta keskeiset elementit sairaalaan ICT-ratkaisua kehitettäessä ja sen tärkeimmät kohdat. Vaiheita seuraamalla ja toistamalla tuote voidaan luoda ja jatkojalostaa mahdollisimman onnistuneesti.

1. Suorita laaja ja tarkka kartoitus ja taustatyö sairaalasta.
    - Mikä sairaalan prosesseista on tehostettavissa?
    - Miten ratkaisusi tehostaa sairaalan toimintaa käytännössä?
    - Voitaisiinko sovelluksella/palvelulla ratkaista useampia ongelmia?
    - Onko ratkaisu hyödynnettävissä muissa sairaaloissa?

2. Arvioi tilanne ja mieti mikä kehitysmalli sopii parhaiten kehittävään ratkaisuun.
    - Waterfall?
    - Ketterät menetelmät? SAFe?
    - Incremental & Iterative?
    - V-Shaped?
    - Sprial?
    - Jokin muu?

3. Suunnittele sovelluksen arkkitehtuuri
    - Sovelluksen perusrakenteen suunnittelu
    - Dokumentoi arkkitehtuuri erittäin tarkkaan ja selitä se auki

4. Aloita kehitys.
    - Security by Design
    - Potilasturvallisuus kehityksen jokaisessa vaiheessa
    - Dokumentointi
    - Jäljitettävyys esim. audit trail
    - Asiakaskohtaiset vaatimukset
    - Liitettävyys muihin järjestelmiin

5. Testaa ja validoi
    - Sertifioinnit ja tarvittavat merkinnät
    - Validointi ja testaus kliinisten asiantuntijoiden kanssa jatkuvasti
    - Lakien ja lainsäädäntöjen noudatus jatkuvasti
    - Käytettävyys, suorituskyky, koodin laatu, kyberturvallisuus

6. Toiminnan jatkuvuus
    - Oman tuotteen varmuuskopiointi
    - Miten tuotetta voidaan tukea tehokkaasti?
    - Mahdolliset vara -ratkaisut esim. node
    - Päivitykset ja niiden toteutus, palvelukatko *vs* 0-downtime

7. Julkaisu
    - Tuotteen jatkuva kehitys
    - Tietoturvapäivitykset
    - Tuotteen laajentaminen?