Heikki Ohtonen

Asiakkaan ja ohjelmistotalon välisen vuorovaikutuksen kehittäminen ohjelmiston ylläpidossa – case Datala Oy

Opinnäytetyö

tradenomi (YAMK), tietojenkäsittely ja liiketoimintaosaaminen

Kevät 2021

Tiivistelmä

Tekijä: Ohtonen Heikki

Työn nimi: Asiakkaan ja ohjelmistotalon välisen vuorovaikutuksen kehittäminen ohjelmiston ylläpidossa – Case Datala Oy

Tutkintonimike: Tradenomi (YAMK), tietojenkäsittely ja liiketoimintaosaaminen

Asiasanat: Ohjelmiston elinkaari, ylläpito, prosessien kehittäminen, asiakassuhteiden hallinta

Tämän opinnäytetyön tavoitteena oli kuvata toimeksiantajan Datala Oy:n ohjelmiston ylläpidon prosessi nykytilassa ja samalla kehittää kyseistä prosessia. Opinnäytetyön avulla toimeksiantaja voi kehittää liiketoimintaansa. Toimeksiantaja Datala Oy on ohjelmistoalan yritys, joka toimittaa ohjelmistotuotteita ja palveluita erityisesti elintarviketeollisuudelle. Opinnäytetyön tekijä on töissä toimeksiantajalla.

Opinnäytetyön teoriaviitekehys muodostuu ohjelmiston elinkaaresta ja ylläpidosta, prosessien kehittämisestä sekä asiakassuhteiden hallinnasta. Ohjelmiston elinkaaresta käytiin läpi erilaisia elinkaarimalleja ja kehitysmalleja. Elinkaaren vaiheista tarkemmin käytiin läpi ohjelmiston ylläpitovaihetta. Prosessien kehittämisen teorian avulla luotiin prosessikuvaus nykytilasta. Saman teorian avulla tehtiin myös prosessin kehittäminen. Asiakassuhteiden hallinnan teoriassa käytiin läpi asiakkaan tarpeiden tunnistamista ja asiakastiedonhallintaa. Teoriaviitekehys on muodostettu kirjallisuudesta ja tieteellisistä artikkeleista.

Opinnäyteyön tutkimusosa on kvalitatiivinen tapaustutkimus. Tutkimusmenetelminä käytettiin teemahaastatteluja ja havainnointia. Aineistosta etsittiin asiasisältöä, jotta siitä löydettiin ilmiötä kuvaavia sisältöä. Prosessianalyysin avulla tutkittiin ylläpidon prosessia ja kehitettiin prosessia.

Opinnäytetyön tutkimuksen tuloksena syntyi nykytilan kuvaus. Opinnäytetyössä nykytilan kuvauksen perusteella ohjelmiston ylläpidon prosessia kehitettiin, niin että sen myötä työtapoja ja dokumentointitapoja voidaan yhtenäistää ja toimintaa tehostaa. Opinnäytetyössä prosessit kuvattiin sanallisesti, toimintotaulukkoina ja prosessikaavioina. Opinnäytetyön tuloksia ei ole viety tämän opinnäytetyön puitteissa toimeksiantajan toimintaan.

Abstract

Author: Ohtonen Heikki

Title of the Publication: Developing the Interaction between the Customer and the Software Company in Software Maintenance – Case Datala Oy

Degree Title: Master of Business Administration

Keywords: Software life cycle, software maintenance, process development, customer relationship management

The purpose of this Master's thesis was to describe the current state of software maintenance process for the client Datala Oy. In addition, the aim of thesis was also to develop that process. The thesis offers Datala Oy opportunity to develop its business. The client, Datala Oy, is a software company that makes software products and services especially for the food industry. The thesis's author works for the client.

A theoretical framework of the thesis includes software lifecycle and maintenance, process development and customer relationship management. The life cycle of the software was reviewed through various lifecycle models and software development models. The maintenance phase of the software life cycle was explored more detailed than other phases. The theory of process development was used to create a process description of the current state. The same theory was used to develop the process. The customer relationship management theory was explored to how identify customer needs and how to management customer data. The reference framework for theory is constructed from literature and journal periodicals.

Qualitative case research method was used in this thesis. The research tools that were used were thematic interviews and observation. The interview material was analyzed to find key words describing the current state of the software maintenance process. As a result of the research, a description of the current state of the maintenance process was created. Based on the research results, the software maintenance process was developed so that working methods and documentation methods can be harmonized and operations are made more efficient. With the help of the analysis, the current state of the maintenance process was described and a proposal for the development draw.

In the thesis, the processes are described as a text, as function tables and as process diagrams. Implementation of the proposed software maintenance process is not included into this thesis.

Sisällys

Käsitteet

CRM	Customer Relationship Management, asiakassuhteiden hallinta, joka käsitetään usein pelkästään asiakkuudenhallintaohjelmistoksi. CRM:n avulla organisaatiot voivat parantaa nykyisiä asiakassuhteita ja hankkia uusia asiakkaita.
Debugger	Ohjelma, jolla voidaan jäljittää ohjelmointivirheitä tietokoneohjelmista. Sen käyttö edellyttää useimmiten tutkittavan ohjelman lähdekoodin läsnäoloa. Suomen kielellä debuggeria voitaisiin kutsua virheenjäljittimeksi.
Kanban	Visuaalinen työnhallintatapa, jossa työn etenemistä seurataan värikkäin kortein.
Lean	Menetelmä, jossa pyritään kehittämään prosesseja. Pyrkimyksenä on jättää tekemättä kaikki sellainen mikä ei tuota asiakkaalle lisäarvoa. Lean-menetelmää on käytetty ohjelmistotuotannossa 2000-luvulta lähtien.
Prosessi	Prosessi on toistuva joukko toisiinsa liittyviä toimintoja, jotka tuottavat tuloksen. Tulokset ovat joko tuotteita tai syötteitä toisille prosesseille, jotka voivat lisätä tuotteen tai toisen prosessin arvoa.
Scrum	Ohjelmistokehityksessä käytetty ns. ketterä menetelmä. Scrum peräisin Japanista, jossa sitä käytettiin tuotekehitykseen. Ohjelmistonkehitykseen se on tullut 1990-luvulla.

1 Johdanto

Datala Oy on toiminut vuodesta 1983 tarjoten palveluitaan lähinnä elintarviketeollisuudelle ja tehden vähäisessä määrin alihankintatöitä (Datala 2020). Datala Oy tuottaa palveluita ja ohjelmistoja, joiden avulla esimerkiksi meijerit voivat palvella maidontuottajia. Näitä palveluita ovat meijerin tekemät maitotililaskelmat ja tuottajien webportaali, jossa tuottajat voivat seurata eläimistä, maidosta ja rehuista otettuja koetuloksia. Koetulosten perusteella maidontuottajat voivat kohdentaa ruokintaa eläinkohtaisesti ja parantaa maidontuotantoa. Lisäksi yritys tarjoaa toiminnanohjausjärjestelmä kokonaisuuksia tai osia niistä asiakkaan tarpeiden mukaan. Datala Oy tarjoaa myös ohjelmistojen ylläpitopalvelua, koska yrityksessä on osaamista erityisesti Cobol-ohjelmointikielen osalta. (Datala 2020.) Cobol-kielen osaaminen on tällä hetkellä yrityksen vahvuus, koska ko. kielen osaajista on pulaa. Monessa organisaatiossa on vielä tätä ohjelmointikieltä käyttäviä ohjelmia päivittäisessä käytössä ja niitä on kehitettävä ja ylläpidettävä vastaamaan tämän päivän tarpeita.

Organisaatiorakenne Datala Oy:ssä on hyvin matala, koska työntekijöitä on kolme. Tämä rakenne mahdollista hyvin kevyen hallinnon. Toimitusjohtaja vastaa myynnistä, markkinoinnista ja taloushallinnosta. Hallinnollisten tehtävien lisäksi hän hoitaa osaltaan myös ohjelmistojen kehitystä ja ylläpitoa. Kaksi työntekijää vastaa ohjelmistojen kehityksestä ja ylläpidosta. Ohjelmistojen ylläpito on toiminnallisesti eniten aikaa vievää ja siihen liittyy dokumentointia. Opinnäytetyön lähtötilanteessa henkilöstö tuottaa ylläpitotilanteessa eritasoisia ja laatuisia dokumentteja. Dokumentoinnin hyödyntäminen muissa ylläpidollisissa tilanteissa voi olla haastavaa, koska tapausten hakeminen on muistin varaista. Tämän vuoksi on hyvä vakioida dokumentointi ja se, miten ja minne dokumentit tallennetaan. Ylläpitoprosessia ei ole kuvattu aiemmin, eikä sitä ole kirjattu ylös.

Opinnäytetyön aihe valikoitui työyhteisössä käytyjen keskustelujen myötä. Työyhteisössä nähtiin tarvetta kehittää toimintoja ohjelmistojen ylläpitoon liittyvän työn osalta. Opinnäytetyössä käydään läpi asiakkaan ja ohjelmistotalon välistä vuorovaikutusta ohjelmiston ylläpitoprosessin näkökulmasta. Tutkimusvaihe toteutettiin teemahaastatteluin sekä havainnoinnilla. Tutkimusvaihe toteutettiin tammikuun 2021 aikana tutkijan työpaikalla. Tutkimusvaiheessa tehtiin teemahaastatteluita ja suoritettiin havainnointia tutkijan työpaikalla. Haastattelujen ja havaintojen analysoinnin jälkeen toimeksiantajalle tehtiin prosessikuvaus ohjelmiston ylläpidon nykytilanteesta. Kehittämisvaiheessa prosessia analysoitiin ja sen myötä sitä kehitettiin. Kehitetystä prosessista tehtiin prosessikuvaus. Kehittämistyössä esitettiin käytännön toimenpide ehdotuksia työn ja dokumentoinnin yhtenäistämiseksi. Kehittämistyössä kuvattujen toimenpiteiden käyttöönotto ja kehittäminen ei kuulu tähän opinnäytetyöhän, vaan ne ovat mahdollisen jatkotutkimuksen tai jatkokehityksen piirissä.

Opinnäytetyö koostuu katsauksesta aiheeseen liittyvään kirjallisuuteen, menetelmäkirjallisuuteen, varsinaisesta tutkimuksesta ja kehittämisestä sekä johtopäätöksistä ja pohdinnasta. Opinnäytetyön lopussa on liitteet, joissa esitetään haastattelukierrosten teemat sekä esimerkki havaintopäiväkirjasta.

Opinnäytetyön aihekirjallisuus katsaus on luvussa kaksi ja siinä käsitellään tutkimukseen liittyviä teorioita. Käsiteltäviksi teorioiksi valikoituivat ohjelmiston elinkaari, prosessit ja niiden kehittäminen sekä asiakassuhteiden hallinta. Ohjelmiston elinkaaresta esitellään muutamia erilaisia elinkaarimalleja, sekä niihin liittyviä vaiheita. Teoriassa keskitytään erityisesti ohjelmiston ylläpidettävyyteen ja siihen vaikuttaviin seikkoihin. Tämän jälkeen käydään läpi varsinaista ylläpitoa työnä. Prosessien osalta teoriassa käydään läpi niiden tunnistamiseen ja luokitteluun vaikuttavia seikkoja. Sen jälkeen tutkitaan erilaisia näkökulmia niiden kehittämiseen.

Prosessin kehittämisen näkökulmien esittämisen myötä työssä tutustutaan opinnäytetyön kannalta merkityksellisiin kuvaamistapoihin. Kuvaamistapojen jälkeen käydään läpi erilaisia prosessin kehittämismalleja. Asiakassuhteiden hallinnassa käydään läpi asiakassuhde ja sen hallinta. Sen myötä käydään läpi asiakkaan tarpeiden tunnistaminen sekä asiakastiedon hallinta. Substanssiteorian lopuksi tutustutaan asiakaspalveluun ja asiakastyytyväisyyteen.

Menetelmäkirjallisuudesta käydään läpi tutkimuksen tekemiseen liittyviä seikkoja luvussa kolme. Menetelmäkirjallisuuden käsittelyn avulla tutkija hankki itselleen käsityksen tutkimuksen tekemisestä ja siinä käytettävistä menetelmistä. Luvussa kolme käsitellään tarkemmin tutkimuskysymys, tutkimusote, aineiston kerääminen ja sen käsittely sekä aineiston analysointi ja kehittäminen. Menetelmäkirjallisuus osion lopussa käydään läpi tutkimukseen liittyvät valinnat ja perustellaan ne lukijalle. Tutkimuksen toteutusta ja sen tuloksia esitetään luvussa neljä. Siinä käydään tarkemmin läpi tutkimuksen osoittama nykytilan kuvaus ja sen pohjalta tehty kehittämistyö.

Johtopäätöksiä työstä käydään läpi luvussa viisi. Siinä on tiivistelmä tutkimuksen tekemisestä ja tuloksista. Siinä pohditaan myös työn luotettavuutta ja tehdään itsearviointi omasta oppimisesta. Lopuksi luvussa on pohdintaa jatkotutkimuksesta sekä kehittämisen mahdollisuuksista.

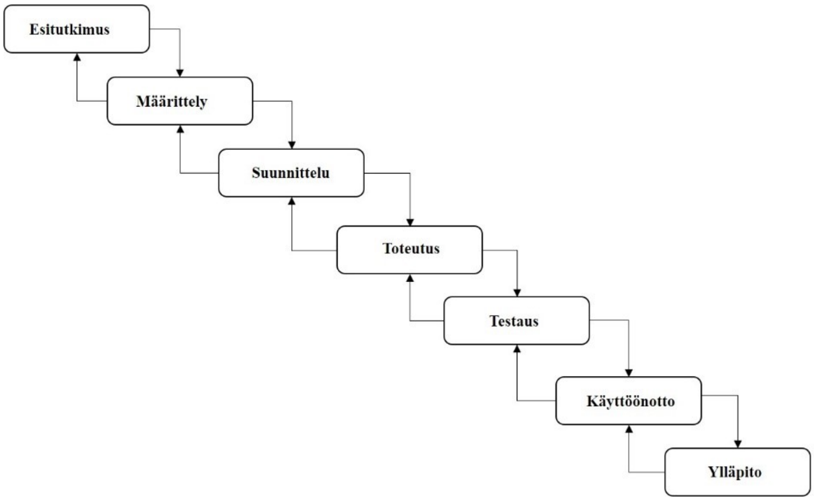
2 Tutkimuksen teoria

Tutkimuksen ja sitä seuraavan kehittämistyön kannalta oleellisimpia teorioita ovat ohjelmiston elinkaari, prosessien kehittäminen ja asiakassuhteiden hallinta. Ensimmäisenä käydään läpi ohjelmiston elinkaari ja keskitytään siinä erityisesti ohjelmiston ylläpitovaiheeseen, joka on useimmiten ohjelmiston elinkaaren pisin osa. Seuraavaksi käsitellään prosessien kehittämistä aloittaen prosessin kuvaamisesta ja siitä edeten prosessien analysointiin. Viimeisenä käydään läpi asiakassuhteiden hallinta ja sitä, kuinka asiakassuhteita voidaan ylläpitää ja edistää hyvän palvelutason myötä.

2.1 Ohjelmiston elinkaari ja ohjelmiston ylläpito

Ohjelmisto käsitteenä tarkoittaa tiettyyn tehtävään tai tietokoneeseen tehtyä ohjelmaa. Ohjelmistoja ovat mm. käyttöjärjestelmät ja eri hallinnonaloille tarkoitetut ohjelmat, kuten esimerkiksi toiminnanohjausjärjestelmät. Tietojärjestelmällä tarkoitetaan laajempaa kokonaisuutta, joka ei sisällä pelkästään ohjelmistoja, vaan se koostuu käyttäjistä, erilaisista laitteista ja ohjelmistoista. Tietojärjestelmän tarkoituksena on tehostaa tai mahdollistaa tiedon käsittelyä. (Pohjonen 2002, 5–6.)

Ohjelmiston elinkaari muodostuu peräkkäisistä ja osin päällekkäisistä vaiheista ja se kuvaa sitä aikaa, joka kuluu kehittämisen aloittamisesta aina sen käytöstä poistamiseen. Vaiheiden avulla voidaan määrittää tehtäviä, ajoituksia ja niiden riippuvaisuuksia toisistaan. Samalla sen avulla voidaan muodostaa viitekehys kullekin vaiheelle, jotta siihen liittyvät käsitteet, ongelmat ja menetelmät ovat hallittavissa. Useimmiten vaiheet kuvataan elinkaareen ns. vaihejakomallisesti, joista tavallisin on ns. vesiputousmalli. (Pohjonen 2002, 26; Haikala & Märijärvi 2004, 36 - 37.) Kuvassa 1 on kuvattu ohjelmiston elinkaari vesiputousmallin mukaan, jonka vaiheet on kuvattu seuraavaksi.



Kuva 1. Vesiputousmalli Pohjosen mukaan, jossa kuvataan ohjelmiston elinkaari (Pohjonen 2002, 40).

Ohjelmistojen elinkaaren vaiheet voivat vaihdella hieman, mutta pääsääntöisesti elinkaari alkaa aina esitutkimuksesta. Esitutkimusvaiheessa selvitetään edellytykset, sille onko hanke toteutettavissa. Sen aikana ei tehdä mitään, vaan selvitetään, minkä ongelman hanke ratkaisee ja asiakkaan todellisen tarpeen selville saaminen ja ymmärtäminen. Samalla siinä tulisi kuvata tavoitteet ja rajaukset sekä erilaiset vaihtoehdot toteutukselle perusteluineen. Esitutkimus ei kuitenkaan välttämättä johda hankkeen etenemiseen, koska sen myötä voi selvitä seikkoja, jotka puoltavat hankkeen hylkäämistä. Väärät asiakasvaatimukset voivat johtaa toteutettuna huonon lopputuloksen. Esitutkimus tuottaa laajalti aineistoa, jota voidaan hyödyntää nykyisten ohjelmistojen ja järjestelmien kehittämisessä. (Pohjonen 2002, 26-28; Haikala & Märijärvi 2004, 37.)

Vaatimusmäärittelyssä on esitetty ohjelmistolle asetetut vaatimukset, jotka jaotellaan toiminallisiin ja ei-toiminnallisiin ja reunaehtoihin. Vaatimuksissa ei oteta kantaa tekniseen toteutukseen. Toiminnalliset vaatimukset määrittävät mitä ohjelmiston odotetaan tekevän, ei-toiminnallisissa vaatimuksissa määritellään esimerkiksi käyttöliittymän tyyli. Reunaehdoissa voi olla mm. vaatimus käyttöjärjestelmästä, jossa ohjelmiston on toimittava. (Pohjonen 2002, 28; Haikala & Mikkonen 2011, 61.)

Asiakkaan esittämien vaatimusten kerääminen on haastavaa, koska siihen ei ole määritetty yhtä ainoaa tapaa, jolla saadaan kasattua riittävän laaja lopputulos. Usein käytettyjä menetelmiä, ovat sidosryhmien haastattelut ja ryhmätyöt, kuten ideointipalaverit. Palavereita joudutaan pitämään useita ennen kuin vaatimusmäärittely on saatu tehtyä kokonaan. Muita vaatimusmäärittelyyn vaikuttavia tekijöitä voivat olla myös kuluttajien toiveita ja odotuksia mittaavat tutkimukset. Lisäksi riippuen ohjelmiston käyttökohteesta on vaatimusmäärittelyssä huomioitava standardit, asetukset ja lainsäädäntö. Vaatimusmäärittelyssä on siis tärkeää pyrkiä huomioimaan monipuolisesti kaikki lähteet, jotta vaatimusmäärittely ei jää puutteelliseksi, koska puutteiden korjaaminen voi tulla kalliiksi myöhemmissä vaiheissa. (Pohjonen 2002, 28 – 29.)

Vaatimusmäärittelyissä on usein ongelmana, se että vaatimukset ovat keskeneräisiä ja niiden välillä on ristiriitaisuuksia. Näiden seikkojen vuoksi vaatimuksia joudutaan usein käsittelemään ja muokkaamaan ennen kuin niitä voidaan ottaa mukaan vaatimusmäärittelyyn. Vaatimusmäärittelyyn kirjattujen vaatimusten on oltava helposti luettavissa ja ymmärrettävissä, jotta niiden tulkinta menisi oikein. Sen vuoksi vaatimukset onkin kirjattava mahdollisimman tarkasti. (Pohjonen 2002, 29.)

Yksittäistä vaatimusta analysoitaessa on otettava huomioon, mikä on sen todellinen tarkoitus ja merkitys. Näin voidaan löytää lisävaatimuksia tai monimutkaisia ongelmia, joita ei välttämättä nähdä heti, kun vaatimusta käsitellään ensimmäistä kertaa. Vaatimusta analysoitaessa on otettava huomioon se seikka, että ohjelmistot eivät voi ratkaista ongelmia joihin organisaatiossa ei ole saatu aikaan ratkaisuja muutoinkaan. Nämä ongelmat voivat olla luonteeltaan sellaisia, että ne tuovat vaatimusmäärittelyyn epämääräisiä vaatimuksia. Organisaation esittämien vaatimusten toivotaan ratkaisevan ongelmat. Ongelmat eivät kuitenkaan tule ratkaistuiksi ilman, että niitä analysoidaan ja niihin ohjataan tarvittavat resurssit. Tällaiset ongelmat voivat nousta vaikeiksi kohdiksi ja aiheuttaa haasteita myöhemmin kehitystyön aikana. Analysoinnissa olisi kyettävä selvittämään jokaisen vaatimuksen tarve ja miten tärkeä vaatimus on. Analysoinnin on kyettävä sovittamaan samalla ristiriitaiset vaatimukset yhteen. (Pohjonen 2002, 29 – 30; Haikala & Märijärvi 2004, 95.)

Vaatimusten dokumentoinnin yhteydessä yksittäiset vaatimukset on syytä priorisoida, koska usein ohjelmistoja toteutetaan versioittain. Priorisoinnin avulla voidaan määrittää mitä kussakin ohjelmistoversiossa toteutetaan. Sitä voidaan hyödyntää myös silloin kun, kehitystyössä kohdataan aikataulu- ja kustannusongelmia. Aikataulu- ja kustannusongelmien vuoksi toiminnallisuudesta joudutaan tinkimään. Tällöin korkeimman prioriteetin vaatimukset voidaan toteuttaa ensimmäiseksi. (Pohjonen 2002, 30.)

Riippuen ohjelmiston laajuudesta, voidaan vaatimusmäärittelyn jälkeen tehdä vielä erillinen analyysi. Tässä vaiheessa analysoidaan vaatimukset ja niistä johdetaan sen jälkeen toiminnallinen määrittely. Siinä kuvataan toiminnot, käsiteltävät tiedot, liittymät ja yhteydet muihin ohjelmistoihin. Toiminnallinen määrittely voi pitää sisällään myös käyttäjien määrittelyn. Tässä vaiheessa dokumentoinnin ja sen selkeyden on oltava erityisen tarkkaa, koska sen pohjalta voidaan aloittaa myös testauksen suunnittelu ja käyttöohjeiden tekeminen. (Pohjonen 2002, 30 - 31.)

Edellisten vaiheiden jälkeen on edetty suunnitteluun, jossa asiakkaiden tarpeiden mukainen toiminnallinen määrittely muutetaan tekniseksi määrittelyksi. Teknisessä määrittelyssä kuvataan toteutustapa, joka voidaan jakaa tarvittaessa kahteen eri osa-alueeseen. Arkkitehtuurisuunnittelu on ohjelmistosuunnittelun keskeisin keino, sillä siinä määritetään yleinen rakenne, joka paloitellaan pienemmiksi osiksi, joita yksittäiset kehittäjät voivat suunnitella ja toteuttaa valitun ohjelmointikielen rakentein. Nämä osat muodostavat oman kokonaisuuden, jota kutsutaan moduuliksi tai komponentiksi. Moduuli kommunikoi muiden moduulien kanssa rajapintojen kautta. Suunnittelussa pyritään moduulien osalta mahdollisimman suureen itsenäisyyteen, jotta moduulien väliset yhteydet saadaan pidettyä pieninä. Moduulien välisten yhteyksien kasvaessa myös ohjelmiston rakenne monimutkaistuu ja siitä tulee vaikeammin testattava sekä ylläpidettävä. Siksi suunnittelussa on pyrittävä selkeisiin ja ymmärrettäviin ratkaisuihin. Tällöin ratkaisujen on oltava yksinkertaisia ja suoraviivaisia sekä ne on toteutettava yhdenmukaisesti. (Pohjonen 2002, 32 - 33; Haikala & Mikkonen 2011, 177 - 180.)

Moduulisuunnittelun tarkoituksena on suunnitella moduulin rakenne. Moduulin sisältämällä koodin rivimäärällä on merkitystä, koska pienemmät moduulit ovat yksinkertaisempia ja helpompia ylläpitää. Useimmiten moduulit tarvitsevat jonkun toisen moduulin tuottamaa ”palvelua”. Samalla on kiinnitettävä huomiota hierarkiaan, jotta moduuli ei joudu käyttämään useita alimoduuleja, muutoin yksittäisen ohjelman rakenteista tulee monimutkaisia. Suunnittelun dokumentaatiossa olisi mainittava tiivistetysti ohjelmiston tarkoitus ja minkälaisessa ympäristössä se toimii. Dokumenteissa olisi hyvä mainita myös toteutukseen vaikuttavista reunaehdoista, liittymistä ja kuvaus käytetyistä arkkitehtuureista sekä kuvaus teknisistä ratkaisuista. (Pohjonen 2002, 33.)

Toteutusvaiheessa ohjelmisto luodaan tehdyn suunnitelman mukaisesti valitulla ohjelmointikielellä. Ohjelmisto koostuu useista moduuleista ja toteutusvaiheen lopuksi ne integroidaan yhteen, jolloin niistä muodostuu toimiva kokonaisuus. Jos edelliset vaiheet on toteutettu huolellisesti, niin toteutus itsessään on suoraviivainen vaihe, koska rakenne ja toiminnallisuus on määritetty jo aiemmin. Toteutuksen onnistuminen ei ole kiinni valitusta teknologiasta, vaan siitä miten hyvin toteutus vastaa sille asetettuja vaatimuksia ja on lisäksi toiminnallisesti sekä teknisesti määrittelyjen mukainen. Onnistumiseen vaikuttaa edellisten vaiheiden huolellisen suunnittelun lisäksi myös toteutuksen aikana ilmenneet uudet vaatimukset ja se, miten ne kyetään toteuttamaan. (Pohjonen 2002, 34.)

Toteutuksen aikana tulisi ratkaista siirrettävyydestä ja ylläpidosta johtuvat kysymykset. Ohjelmistot voivat toimia elinkaaren aikana useissa erilaisissa laite- ja käyttöjärjestelmäympäristöissä, joten ylläpitovaihe on huomioitava jo toteutuksen aikana. Näiden syiden vuoksi ohjelmiston rakenne ja toteutus on tehtävä kurinalaisesti. Itse ohjelmoinnissa kurinalaisuudella tarkoitetaan ohjelmointityyliä, jossa ohjelmoija pitää mielessään sen, että tuotettavan koodin on oltava sellaista, että muutkin sitä ymmärtävät. Hyvässä ohjelmointityylissä nimeämiset suoritetaan kuvaavasti ja järjestelmällisesti. Lisäksi ohjelman koodi on kommentoitava selkeästi. Mahdollisuuksien mukaan kannattaa käyttää automaattista dokumentointia itse lähdekoodista, koska jos se on tehtävä manuaalisesti, niin se jää helposti hoitamatta. Useimmiten tämä vaatii kehittäjiä käyttämään määrämuotoista dokumentaatiota. (Pohjonen 2002, 35; Haikala & Mikkonen 195.)

Toteutuksen jälkeen ohjelmisto on testattava. Testaamisen tarkoituksena on löytää virheet ja se voidaan toteuttaa monelle eri tasolla. Testaus on tällöin jaettu ns. V-mallin mukaisesti moduuli-, integrointi- ja järjestelmätestaukseen. (Pohjonen 2002, 36). V-mallissa kuvataan testaamisen tasot V-kirjaimen oikeaan sakaraan alkaen alhaalta edeten ylös, vasemmassa sakarassa on suunnittelun tasot alkaen ylhäältä alas (Wikipedia 2021a). Testaamiseen liittyvät työvaiheet, kuten suunnittelu, testiympäristön luominen, testaamisen suorittaminen ja testaamisen analysointi sekä debuggaus ja korjaaminen voivat viedä yli puolet ohjelmistoprojektin resursseista. Debuggauksella tarkoitetaan virheen jäljittämistä. Moduulitestaus testaa pelkästään yksittäistä ohjelmamoduulia, kun integraatiotestauksessa testataan niiden välistä toimintaa. Järjestelmätestaus käsittää koko järjestelmän toiminnan ja suorituskyvyn testauksen. Testaamisen ja siinä käytettävän testiaineiston on pohjauduttava määrittelyn ja suunnittelun dokumenteissa määritettyihin tuloksiin, koska testaamisen pohjautuessa koodiin, saadaan testaus menemään aina läpi. Täydellisen kattavan testauksen avulla voitaisiin paljastaa kaikki virheet ja puutteet, mutta käytännössä tähän ei päästä, koska asioihin vaikuttavien tekijöiden määrä on suuri. (Pohjonen 2002, 36). Testaamalla ei voida siis osoittaa ohjelman virheettömyyttä edes pienissä ohjelmissa. Siksi ohjelman toimivuuteen ei voi luottaa liikaa, vaikka testitulokset olisivatkin hyvät. (Haikala & Mikkonen 2011, 205.)

Testaamisen jälkeen on vuorossa käyttöönotto. Käyttöönotossa mukana on tekijöitä, jotka on otettava huomioon. Tällaisia tekijöitä voivat teknisten ja fyysisten ympäristöjen muutokset. Käyttöönottoa on valmisteltava huolellisesti ennakkoon myös tietojen siirron takia. Usein käyttöönottoon liittyy tietojen siirtoa vanhasta ohjelmistosta uuteen. Samalla on otettava huomioon käyttäjien ja ylläpidon kouluttaminen sekä mahdollisten laiteympäristöjen vaihdokset. (Pohjonen 2002, 37.)

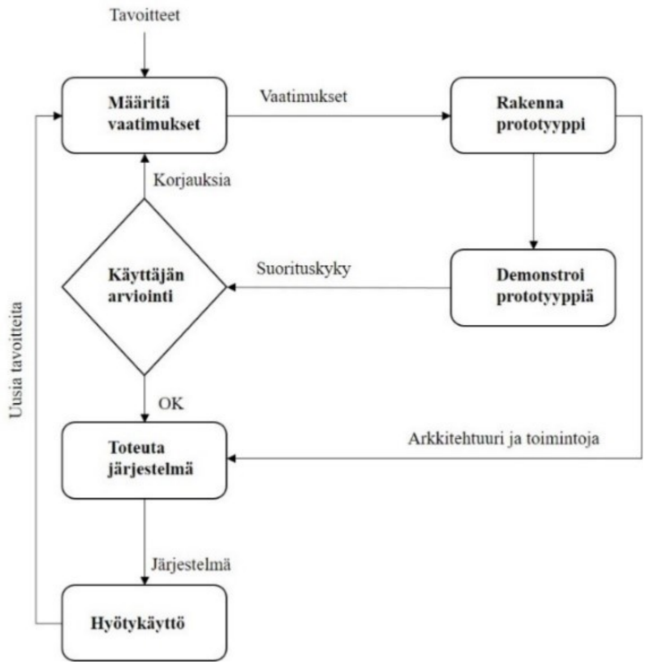
Käyttöönoton jälkeen ohjelmisto siirtyy ylläpitovaiheeseen, joka on sen elinkaaren pisin vaihe. Kyseisessä vaiheessa ohjelmisto on tuotantokäytössä, jolloin ylläpito keskittyy pitämään sen toimintakunnossa, joka usein tarkoittaa virheiden korjauksia. Ohjelmisto voi vaatia myös jatkokehitystä tai muita toimenpiteitä. Arvioiden mukaan ohjelmistoyritysten ajasta noin 70 % kuluu vanhojen ohjelmien ylläpitoon ja loput noin 30 % jäävät siten uusien ohjelmien tekemiseen. Luultavasti ylläpitoon käytetty aika tulee vain kasvamaan, koska ohjelmien ikä ja määrä lisääntyy ja samalla ylläpidettävien ohjelmien koko kasvaa. Ohjelmistojen ja järjestelmien käyttöikä voi olla hyvinkin pitkä, jopa yli 20 vuotta, vaikka niitä suunniteltaessa niiden käyttöiäksi on oletettu 5 – 10 vuotta. Vaikka uusien järjestelmien investointi on yrityksille suuri kustannus, niin on kuitenkin järkevää miettiä uuden järjestelmän käyttöönottoa ja sitä, milloin vanhan järjestelmän tekohengitys kannattaa lopettaa. (Pohjonen 2002, 37; Harsu 2003, 18; Koistinen 2002, 29.)

Ohjelmiston elinkaarimallit

Ohjelmistoprosessia voidaan mallintaa kokonaisvaltaisesti erilaisilla elinkaarimalleilla. Ne ovat kehittyneet perinteisistä ja pitkälle systematisoiduista aloista, kuten autoteollisuudesta. Niiden mukaisesti ohjelmiston elinkaarimalleissa toiminnot on organisoitu integroiduiksi ja järjestelmällisiksi kokonaisuuksiksi. Mallit eivät anna yksityiskohtaisia ohjeita rakentamiseen, ja ne eivät välttämättä sovi käytettäväksi kohteena olevaan sovellusalueeseen. (Pohjonen 2002, 39.)

Vesiputousmalli on kehitetty 1960-luvulla ja sitä pidetään ensimmäisenä elinkaarimallina. Siinä kehittämisprosessi nähdään aina etenevä prosessina, jota on hankala peruuttaa taaksepäin. Vesiputousmallissa elinkaaren vaiheet seuraavat toisiaan alkaen esitutkimuksesta päättyen ylläpitoon. Ideaalitilanteessa näin malli voisikin toimia, mutta todellisuudessa seuraava vaihe voi paljastaa edellisessä vaiheessa tehtyjä virheitä, jolloin prosessissa on pystyttävä peruuttamaan taaksepäin ja korjaamaan kyseinen virhe. Mallin huonona puolena voidaan pitää sitä, että se olettaa edellisen vaiheen loppudokumentin olevan syöte seuraavalle vaiheelle. Näin ollen, jos myöhemmässä vaiheessa havaitaan virhe, niin se edellyttää kaikkien edeltävien vaiheiden uudelleen suorittamisen, joka nostaa kehittämisprosessin kustannuksia. Onkin esitetty arvio, että jos ongelmat tulevat esiin vasta testausvaiheessa, niin silloin ohjelmistoprojektin kustannusarvio voi jopa kaksinkertaistua. Toisena huonona puolena pidetään sitä, että tuloksia voidaan esitellä asiakkaalle vasta varsin myöhäisessä vaiheessa. Näistä seikoista huolimatta vesiputousmalli on tunnetuin ja sovelletuin elinkaarimalli. Sallimalla siinä vaiheiden iterointi ja seuraavien vaiheiden käynnistämisen ennen edellisen loppumista, voidaan mallia soveltaa useisiin eri tilanteisiin. (Pohjonen 2002, 40; Haikala & Mikkonen 2011, 37.)

Prototyyppimallissa pyritään tuottamaan asiakkaalle nopeasti konkreettista arvioitavaa, vaikka se ei olisikaan yksityiskohdiltaan toimiva. Tässä mallissa ei edetä samalla tavalla kuin vesiputousmallissa, jossa asiakkaalle näytetään tuloksia vasta monien vaiheiden jälkeen. Prototyyppimallissa on kaksi päävaihtoehtoa: evoluutio ja poisheitettävä. Evoluutiomallissa prototyyppi kehitetään valmiiksi tuotteeksi. Siinä asiakkaan antaman arvion jälkeen prototyyppiä parannellaan. Arviointi ja parantelu prosessia jatketaan siihen saakka, että asiakas on siihen tyytyväinen. Tämän jälkeen voidaan toteuttaa koko ohjelmisto saadun karkean prototyypityksen mukaisesti, kuten kuvassa 2 on kuvattu. Huonoina puolina voidaan pitää sitä, että se vaatii paljon resursseja, koska prototyyppiä rakennetaan useasti. Lisäksi liiat pelkistykset voivat johtaa siihen, että ohjelmistoon jää piileviä ongelmia. (Pohjonen 2002, 41 - 42; Haikala & Mikkonen, 2011, 38.)



Kuva 2. Prototyyppimalli Pohjosen mukaan (Pohjonen 2002, 41).

Poisheitettävissä prototyypeissä mallinnetaan usein pelkkää käyttöliittymää. Tämä voidaan mallintaa yksinkertaisimmillaan piirtämällä ruutumalleja, joita sitten esitetään asiakkaalle. Niiden avulla voidaan myös kuvata ohjelmistossa tapahtuvaa navigointia. Joskus näyttöjä voidaan generoida erityisillä ohjelmilla ja niihin voidaan liittää toimintalogiikkaa. Tämä voi kuitenkin johtaa asiakasta harhaan, koska prototyypin nähdessään asiakas voi luulla, että ohjelmisto on jo valmis. (Haikala & Mikkonen, 2011, 39.)

Spiraalimallissa keskeisenä ajatuksena on toimia iteratiivisesti. Siinä on erona muihin malleihin verrattuna prosessin riskien jatkuva riskianalysointi ja sen tulosten mukainen prosessin uudelleen ohjaaminen. Spiraalimallissa on neljä toistuvaa vaihetta, joita toistetaan siihen saakka, kunnes järjestelmä on valmis. Ensimmäisenä vaiheena on suunnittelu, jossa määritetään tavoitteet, vaihtoehdot ja rajoitteet. Toisena vaiheena on riskianalyysi, jossa määritetään eri vaihtoehtoihin liittyvät ongelmat. Kolmantena vaiheena on tuotanto ja neljännessä vaiheessa asiakas suorittaa arvioinnin. Tuotantovaiheessa voidaan käyttää haluttuja menetelmiä, kuten esimerkiksi vesiputousmallia. Mallin ongelmana nähdään se, että asiakkaat on saatava mukaan prosessiin ja samalla olisi hallittava myös riskianalyysien tekeminen. Lisäksi mallissa esitetty vaiheiden iteratiivinen toistaminen voi johtaa aikavievään kehitystyöhön. (Pohjonen 2002, 42 - 43.)

Ketterät menetelmät ovat joukko erilaisia kevyitä kehitysmenetelmiä. Niiden periaatteina on, että tärkeintä on tyytyväinen asiakas ja toimivat ohjelmisto. Ketterien menetelmien peruskirjassa Agile Manifestissa on mainittu 12 kohtaa, jotka konkretisoivat ketterien menetelmien ajatuksen. Niissä on kuitenkin omat rajoitteensa, eikä niitä ole helppoa soveltaa laajemmissa kokonaisuuksissa. (Haikala & Mikkonen 2011, 43 - 46.) Seuraavaksi on listattu Agile Manifestissa julistetut periaatteet:

- Tärkeintä on asiakkaan tarpeet täyttävän ohjelmistoversion toimittaminen aikaisessa vaiheessa ja säännöllisesti.
- Muuttuvat vaatimukset eivät ole ongelma, koska ketterät menetelmät mahdollistavat asiakkaan kilpailukyvyn edistämisen.
- Ohjelmistosta toimitetaan toimiva versio lyhyellä aikavälillä, noin kuukauden välein.
- Asiakkaan edustajan on oltava mukana koko projektin ajan ja työskenneltävä ohjelmistokehittäjien kanssa päivittäin.
- Projektit rakentuvat motivoituneiden tekijöiden ympärille, joille annetaan tarvittavat resurssit.
- Kehitystiimin ja sen jäsenten tehokkain tiedon välittämistapa on kasvokkain käytävät keskustelut.
- Edistymistä osoittaa toimiva ohjelmisto.
- Kestävä toimintapa edellyttää jäsenten ylläpitävän työtahtiaan nyt ja tulevaisuudessa.
- Ohjelmiston hyvän rakenteen ja teknisen laadun huomiointi edistää ketteryyttä.

- Yksinkertaisuus.
- Itseohjautuvat tiimit löytävät parhaat ratkaisut.
- Tiimi käy läpi toimintaansa säännöllisesti ja pyrkii parantamaan tehokkuuttaan.

(Agile Manifesto 2020.)

Yleisin käytetty ketterä menetelmä on Scrum, joka on peräisin Japanista. Scrum-menetelmän periaatteet on esitelty alun perin 1980-luvun puolivälissä, mutta ohjelmistonkehityskäyttöön se on tullut 1990-luvun alkupuolella. Scrumia pidetään yksinkertaisena, ja sen perusperiaatteet on helppo selvittää nopeasti. Scrum ei ota kuitenkaan kantaa kaikkiin ohjelmiston elinkaareen liittyviin tehtäviin, eikä se yksinään riitä. (Haikala & Mikkonen 2011, 46 - 47.) Scrum voi täydellisesti omaksuttuna johtaa koko yrityskulttuurin muutokseen, jossa työntekijät muuttavat suhtautumista ja sitoutumista työhönsä (Haikala & Mikkonen 2011, 54).

Scrumin uusin versio on julkaistu marraskuussa 2020. Uuden version suomenkielinen versio julkaistiin helmikuussa 2021. Scrum on oppaassa määritelty niin, että siinä Scrum master luo ympäristön, jossa toteutetaan seuraavat vaiheet:

1. Tuoteomistajalla on monimutkainen ongelma, joka asetetaan kehitysjonoon.
2. Scrum-tiimi tuottaa sprintissä arvoa tuottavan tuloksen.
3. Scrum-tiimi ja sidosryhmät tarkastavat sprintin tuloksia ja niiden perustella muotoilevat sen pohjalta seuraavan sprintin toimintaa.
4. Toistetaan vaiheet 1 – 3.

(Schwaber & Sutherland 2020, 3).

Scrum-tiimi muodostuu yhdestä Scrum Masterista, yhdestä tuoteomistajasta ja joukosta kehittäjiä. Tiimi muodostu yhtenäisestä joukosta, joka keskittyy aina yhteen tavoitteeseen kerrallaan. Tiimit ovat pieniä, enintään 10 hengen kokoisia ja itseohjautuvia, sekä niiden jäsenillä on kaikki tarvittavat taidot, joita tarvitaan tuotteen arvon tuottamiseen. Sprintti on tapahtuma, jossa ideat tuottavat arvoa. Niiden pituus on vakio ja enintään kuukauden mittainen, lisäksi uusi sprintti alkaa aina edellisen päättymisen jälkeen. (Schwaber ym. 2020, 5 - 7).

Lean menetelmä on käytetty ohjelmistotuotannossa 2000-luvulta lähtien. Lean ei ole varsinaisesti menetelmä, vaan tapa ajatella, jota käytetään prosessin kehityksen apuna. Keskeisimmät ajatukset liittyvät siihen, että ei tehdä mitään sellaista mikä ei tuota asiakkaalle lisäarvoa. Toisena keskeisenä ajatuksena on ihmiskeskeisyys, jolloin keskitytään niihin ihmisiin, jotka tuottavat lisäarvon. Tähän liittyy ihmisten arvostus, koulutuksen ja toiminnan tehostaminen sekä työympäristö, jossa kommunikoidaan tehokkaasti hyvässä työilmapiirissä. (Haikala & Mikkonen 2011, 54 - 55.)

Lean menetelmän pääperiaatteet voidaan jakaa viiteen eri vaiheeseen. **Ensimmäisenä** vaiheena on asiakkaan arvon miettiminen. Tällöin organisaation on tiedettävä mitä asiakas haluaa ja mistä asiakas on valmis maksamaan. **Toisena** vaiheena on arvoketjun tunnistaminen. Arvoketjun tunnistamiseksi yrityksen arvoketju on kuvattava, jotta siitä tunnistetaan asiakkaalle arvoa tuovat toiminnot. Toiminnot, jotka eivät tuota lisäarvoa poistetaan. Arvoketjuun sisällytetään koko ketju alkaen suunnittelusta ja raaka-aineista mahdolliset alihankkijat on otettava tarkasteluun. **Kolmas** vaihe on tuotannon virtauksen sujuvoittaminen. Fyysisten materiaalien virtausten lisäksi on kiinnitettävä huomioita myös informaatiovirtoihin. **Neljäs** vaihe on imuohjauksen toteuttaminen, jota päästään toteuttamaan silloin kun, asiakasarvoa parhaiten toteuttava arvoketju on tunnistettu. Imuohjauksessa tuotteet valmistetaan asiakkaan tilauksen perusteella. **Viides** vaihe on täydellisyyteen pyrkiminen, johon päästään prosessien jatkuvalla kehittämisellä. (Vuorinen 2013, 72 - 75.)

Kanban on työnhallintatapa, jossa on kolme periaatetta. Ensimmäinen periaate koskee työn kulun visualisointia. Siinä jokaiselle vaiheelle on sarakkeet, joita voivat olla mm. ei-aloitettu, aloitettu ja valmis. Työ on merkitty kortille, ja kortti liikkuu työn vaiheiden mukaisesti sarakkeesta toiseen. Kanbanin toisen periaatteen mukaisesti samassa vaiheissa olevien töiden määrää on rajoitettu, jotta vaiheista ei tule pullonkaulaa. Kolmantena periaatteena on läpimenoaikojen mittaaminen, jotta prosessia voidaan optimoida. Kanbanin periaatteita ei voi kuitenkaan suoraan tuoda teollisuudesta it-alalle, vaan kanban on tällöin enemminkin joukko erilaisia konsepteja. (Haikala & Mikkonen 2011, 55; Leopold & Kaltenecker 2015, 13.)

Ohjelmiston ylläpidettävyys

Ohjelmiston ylläpidettävyyttä voidaan pitää eräänä ohjelmiston laatutekijänä. Laatutekijöitä ei voi mitata suoraan mittareilla, jolloin niiden yhteydessä puhutaan arvioinnista. Jotta arviointia voidaan suorittaa, on määritettävä millaisista mitattavista ominaisuuksista laatutekijät koostuvat. Ominaisuudet voidaan mitata joko suorasti tai epäsuorasti. Suoraan mitattavat ominaisuudet ovat sellaisia, joiden tulokset eivät riipu muiden ominaisuuksien mittaamisten tuloksista. Esimerkkinä suoraan mitattavasta ominaisuudesta voidaan pitää ohjelman sisältämän koodirivien määrää. Epäsuora mittaaminen liittyy ominaisuuksiin, jotka liittyvät yhden tai useamman muun ominaisuuden mittaamisen, josta esimerkkinä voidaan pitää ylläpidettävyyttä. Ylläpidettävyyden liittyviä suoraan mittavia muita ominaisuuksia olisi tällöin esimerkiksi koodirivien määrä ja dokumentointi. (Harsu 2003, 50 - 51.) Ylläpidettävyyden ennakointiin vaikuttaa ohjelmiston tunteminen ja ylläpito henkilöstö. Keskimääräisen korjausajan määrittämiseen vaikuttavat tekijät, kuten esimerkiksi ongelman tunnistamiseen kuluva aika, hallinnollinen viivästys ja ongelman analysointiin kuluva aika. Ylläpidettävyys laatutekijällä tarkoitetaan, sitä miten helppoa muutosten tekeminen ohjelmistoon on. Ylläpidettävyys laatutekijään vaikuttavat muutkin laatutekijät, kuten virheettömyys, luotettavuus ja käytettävyys. (Grubb & Takang 2003, 260 - 261; Harsu 2003, 57.)

Seuraavat seikat vaikuttavat ylläpidettävyyteen heikentävästi:

- Huonosti tehty suunnittelu ja toteutus.
- Vanhan laitteiston käyttö.
- Määrittelyiden puutteellisuus.
- Eri ohjelmointikielien käyttö samassa järjestelmässä.
- Ohjelmiston koon paisuminen suureksi.
- Puutteellinen dokumentaatio tai sen puuttuminen kokonaan.
- Käyttöliittymien huono käytettävyys.
- Henkilöstön taitojen puutteet ohjelmoijissa ja ylläpitäjissä.

(Harsu 2003, 58.)

Pigoskin (1997, 290 - 291) mielestä ohjelmiston ylläpidettävyys on otettava huomioon jo ohjelmiston kehitysvaiheessa. Hän on laatinut listan suositeltavista käytännöistä, jotka ovat osin varsin yksityiskohtaisia. Ohjeita tulisi hänen mielestään noudattaa niin ohjelmiston kehitys-, kuin ylläpitovaiheessakin. Suositeltuja käytäntöjä ovat esimerkiksi:

- Varmistaa, se että jokainen ohjelmarivi sisältää ainoastaan yhden komennon.

- Ohjelmakommenteissa on oltava käyttökelpoista tietoa.
- Pyritään yleisesti käytettyihin ohjelmointityyleihin.
- Pyrkimys yksinkertaiseen koodiin.
- Tunnistetaan ohjelman heikot kohdat.

(Pigoski 1997, 290 - 291.)

Ylläpito ja ohjelmat

Yrityksille hyvin toimiva ylläpito voi olla yksi sen menestyksen kulmakivistä, koska hyvin toimivat järjestelmät mahdollistavat päivittäisten toimintojen sujuvan toiminnan. Yritysten liiketoimintaympäristöissä voi kuitenkin tapahtua muutoksia nopeastikin, jolloin myös ohjelmistojen ja järjestelmien on joustettava toiminnan muutosten mukana. Ylläpidon kehittäminen tulisikin huomioida tietojärjestelmien kokonaiskehityksessä ja strategiassa. Ylläpidossa ei kuitenkaan tulisi niputtaa kehittämistä ja vikojen korjausta yhteen. Muuten se sotkee molempia toimintoja ja johtaa virheisiin arvioidessa muutosten tekemiseen tarvittavaa aikaa ja rahaa. Pahimmillaan kuvitellaan, että ylläpito on vain vikojen ja virheiden korjausta. On tärkeää ymmärtää, että ylläpito on normaali osa ohjelmiston elinkaarta. Hyvin toteutettu ylläpitoprosessi auttaa vähentämään vaivaa ja kustannuksia. Jotta ylläpitoprosessi voidaan toteuttaa hyvin, on prosessi määritettävä kunnolla ja hyödynnettävä tarjolla olevia työkaluja sekä tekniikoita. (Koistinen 2002, 29 - 30; Pigoski 1997, 17; 53.)

Ylläpidettävät ohjelmistot ovat yleensä vanhoja ja niiden rakenne voi olla haasteellinen, koska ne voivat koostua ohjelmista, joita useat eri henkilöt muuttaneet. Näissä tapauksissa ohjelmat ovat kehittyneet vuosien saatossa, ja niihin on tällöin tehty lisäyksiä, korjauksia ja muita ylläpidollisia muutoksia. Ohjelmien pitkä käyttöaika osoittaa, että ne ovat yrityksille tärkeitä, ja niiden korvaaminen voi olla vaikeaa tai joskus mahdotonta. (Harsu 2003, 65.)

Ohjelmien muutostarve voi johtua käyttäjien tarpeiden muutoksesta, jolloin muutos ei välttämättä ole merkki siitä, että ohjelmat olisi suunniteltu tai toteutettu huonosti. Ohjelmien evoluutiota tutkineet tutkijat ovat muodostaneet havaintojensa pohjalta ns. **Lehmanin lait**, joissa on käsitelty ohjelmien muutoksia:

- **1. Lehmanin** laki on jatkuva muutos, jonka mukaan muutoksia ei voi välttää, jos halutaan, että järjestelmä pysyy käyttökelpoisena.

- **2. Lehmanin** lain mukaan ohjelmistorakenne monimutkaistuu muutosten yhteydessä. Yksinkertaisuuden säilyttämiseksi on käytettävä resursseja, jotka voivat lisätä kustannuksia kehittämisen aikana, mutta yksinkertaisuuden säilyttäminen voi kuitenkin vähentää ylläpitokustannuksia tulevaisuudessa.

- **3. Lehmanin** laki on suurien järjestelmien evoluutio, jonka mukaan niiden kasvua voivat rajoittaa organisatoriset tai psykologiset syyt. Näissä tapauksissa muutoksia ei haluta tehdä, koska sen pelätään aiheuttavan uusia tuntemattomia virheitä aiheuttaen koko järjestelmän toiminnan heikkenemisen.

- **4. Lehmanin** laki on kehittämisnopeuden muuttumattomuus, jonka mukaan koko elinkaaren aikana kehittämisvauhti pysyy samana riippumatta siihen saatavilla olevista resursseista.

- **5. Lehmanin** laki käsittelee samankaltaisuuden säilymistä, jonka mukaan versiojulkistusten välillä ei ole suuria eroja ohjelmiston elinkaaren aikana.

- **6. Lehmanin** laki on jatkuvan kasvun laki. Sen mukaan toiminnallisuuden on kasvettava koko ohjelmiston elinkaaren ajan.

- **7. Lehmanin** laki on heikkenevän laadun laki. Ohjelmistot on rakennettu tiettyjen olettamusten varaan ja jos toimintaympäristön muutoksiin ei varauduta eikä toimita, niin ohjelmiston laatu heikkenee ajan myötä.

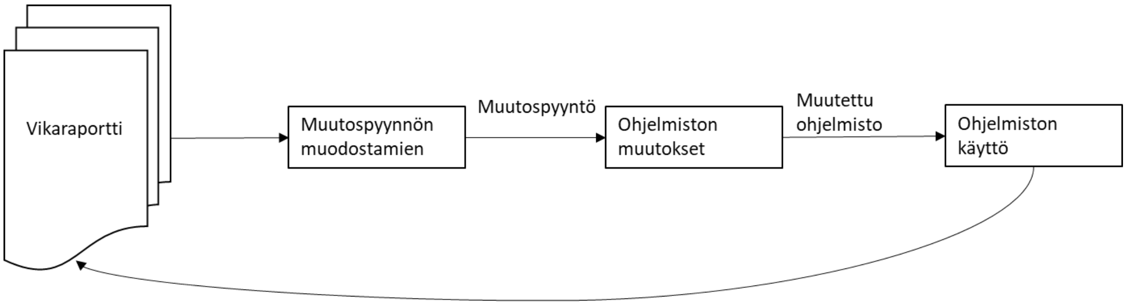
- **8. Lehmanin** laki on palautejärjestelmän laki. Ohjelmistojen evoluutio on monitasoista ja palautejärjestelmällä on roolinsa jokaisessa edellä käsitellyssä laissa.

(Harsu 2003, 66 - 67; Grubb ym. 2003, 45 - 46.)

Ohjelmien käyttökelpoisena pitäminen vaatiikin edellä mainitun Lehmanin ensimmäisen lain mukaan jatkuvaa muutosta. Muutoksen ajureina voivat olla lakimuutokset, kansainväliset säädökset, hallinnolliset muutokset tai organisaation rakenteissa tapahtuneet muutokset. Jatkuva muutos kuitenkin muuttaa ohjelmien rakennetta huonommaksi, johon voi olla syynä mm. ohjelmiston kehityksessä mukana olleet useat eri ohjelmoijat, jotka eivät omaa tarkkaa käsitystä koko ohjelmiston toiminnasta. Uuden ohjelmiston käyttöönotto voi olla riskialtista, jolloin vanhojen ohjelmien käyttöä on jatkettava. Vanhasta ohjelmasta ei ole aina olemassa tarkkaa määrittelyä. Vaikka määrittely olisikin dokumentoitu, niin tehtyjä muutoksia ei ole välttämättä päivitetty dokumenttiin. Tämän vuoksi uuden ohjelmiston kehittäminen vanhan tilalle on vaikeaa. (Harsu 2003, 67 - 68.)

Ohjelmiston ylläpito on haastavaa, jos ohjelmistosta on käytössä puutteellinen dokumentaatio. Puutteellinen dokumentaatio vaikeuttaa ohjelmiston kehityksen ymmärtämistä, koska tällöin ei voida ymmärtää kaikkea suunnittelu- ja toteutusratkaisuja. Dokumentoinnin tulisi olla jatkuvasti päivittyvää ja kattaa kaikki osa-alueet. Itse dokumentointi tulisi tehdä standardimuotoisena, jotta se olisi helposti löydettävissä. Lisähaasteen ylläpitoon tuo saman ohjelmiston poikkeavat versiot, esimerkiksi asiakaskohtaisesti räätälöidyt ohjelmaversiot. Näissä tilanteissa yhteen versioon tehty päivitys voi jossakin toisessa versiossa tuottaa ei-toivottuja tuloksia. (Pohjonen 2002, 37 - 38.)

Koposen (2007, 15) mukaan käytössä on kolme kansainvälistä standardia, jotka määrittelevät ohjelmiston ylläpitoprosessin. Näitä standardeja ovat IEEE 1219, ISO/IEC 14764 ja ITIL. Standardeissa on määritetty tarvittavat toiminnot. Niiden avulla voidaan päätellä mitä tehtäviä on tehtävä. (Koponen 2007, 15.) Esimerkiksi IEEE 1219 määrittelee ylläpitoprosessiin seitsemän eri vaihetta. Siinä ensimmäisenä vaiheena on ongelman luokittelu ja tunnistaminen. Sen jälkeen suoritetaan analysointivaihe, jossa analysoidaan mm. muutosten vaikutuksia ja kustannuksia. Näiden tietojen pohjalta päätetään muutoksen tekemisestä. Korjaavissa ylläpitotoiminnoissa tätä kuitenkaan harvoin käydään läpi, vaan korjaus tehdään ilman analysointia. Suunnitteluvaiheessa kaikki kerätty tieto käydään läpi ja tietoa käytetään suunnittelun aikana. Toteutus vaiheessa tehdään konkreettinen työ, joka sisältää koodaamista eri tason testaamista. Tässä vaiheessa laaditaan suunnitelma käyttöönottamisesta ja päivitetään dokumentaatiot. Järjestelmä testauksessa testataan kokonaisuutta ja sitä, että ohjelmisto toteuttaa alkuperäisen vaatimusmäärittelyn mukaiset toiminnot. Lisäksi siinä on otettava huomioon uudet lisätyt ominaisuudet. Hyväksymistestaus tehdään täysin integroidussa ympäristössä, ja usein sen toteuttaa asiakas. Toimitus vaiheessa asiakkaalle toimitetaan uusi versio käyttöönotettavaksi. (Pigoski 1997, 42 - 46.) Koponen on määritellyt ylläpidon prosessin kuvan 3 mukaiseen malliin, jossa vikaraportti on toiminnan laukaiseva tekijä. On otettava kuitenkin huomioon, että ohjelmiston ylläpidollinen prosessi voi käynnistyä myös muiden, kuin ohjelmistossa olevien vikojen vuoksi.



Kuva 3. Ohjelmiston ylläpidon prosessi Koposen mukaan (Koponen 2007, 16).

Ohjelmien sisään on koodattu organisaation toimintasäännöt ja prosessit. Se voikin olla ainoa dokumentti, joka niistä on tehty. Ohjelmaan on voitu koodata erilaisia säännöstöjä, mutta niiden erottaminen muusta ohjelmakoodista voi olla haasteellista. Vanhan ohjelmiston muuttaminenkin on haastavaa mm. siksi että eri ihmiset ovat toteuttaneet ohjelmiston eri osia, jolloin ohjelmistossa voi olla kirjavia joukko erilaisia ohjelmointityylejä. Muita haasteita muutokselle voivat olla mm. se, että ohjelmisto on voitu toteuttaa vanhentuneella ohjelmointikielellä, jolloin sitä taitavia ohjelmoijia on vaikea löytää. Haasteina on ohjelmiston dokumentointi, joka ei ole ajantasainen, ja joskus ainoa dokumentti voi olla lähdekoodi. Huonoimmassa tapauksessa pelkkä suoritettava ohjelma voi olla ainoa dokumentti ohjelmasta. Suorituskyvyn optimointi on voinut myös johtaa koodin selkeyden ja luettavuuden heikkenemiseen. Tämän takia kokemattomat ohjelmoijat eivät ymmärrä koodia. Lisäksi ohjelmiston data on hajallaan ja niiden rakenne on epäyhtenäinen. (Harsu 2003, 68 - 69.)

Useimmiten yhden virheen korjaus aiheuttaa sivuvaikutuksena muita virheitä, jos ohjelmiston eri osa-alueet ovat voimakkaasti sidoksissa toisiinsa. Virheiden paikantamisen ja korjaamisen vuoksi arkkitehtuurisuunnittelussa olisi otettava huomioon, että toiminnot on kapseloitu omiin moduuleihinsa. Ylläpidon toiminta onkin sidoksissa voimakkaasti siihen, kuinka ylläpito huomioidaan määrittely- ja suunnitteluvaiheessa. Ylläpidon helppoudella säästetään helposti ohjelmiston elinkaaren aikana merkittäviä summia rahaa, vaikka suunnitteluvaiheessa siihen käytetyt resurssit voivat tuntua rahan haaskaamiselta. (Pohjonen 2002, 38.)

Ylläpitovaihe voidaan luokitella neljään eri luokkaan, joita ovat korjaava ylläpito, sopeuttava ylläpito, täydentävä ylläpito ja ennakoiva ylläpito. Näistä korjaava ylläpito on virheiden korjaamista, sopeuttava ylläpito on uusien ympäristöjen käyttöönottoa ja täydentävässä ylläpidossa toteutetaan uusia ominaisuuksia sekä ennakoiva ylläpito. Ennakoivassa ylläpidossa parannetaan dokumentaation tasoa tulevaisuutta silmällä pitäen. (Pohjonen 2002, 37.)

Karkeasti jakaen ohjelmiston ylläpito voi olla joko virheiden korjaamista tai uusien omaisuuksien lisäämistä. Ennen kuin muutos tehdään, on ohjelman koodista löydettävä sopiva kohta, johon se tehdään. Lisäksi on huomioitava vaikutukset, joita tehdyillä muutoksilla voi olla muuhun koodiin. Vaikutukset voivat jäädä kuitenkin huomaamatta heti, jolloin joudutaan tekemään virheen korjaus, joka voivat aiheuttaa lisää virheitä. Tätä aaltomaista muutostarpeiden etenemistä kutsutaan väreilyvaikutukseksi. (Harsu 2003, 77.)

Ohjelmiston ylläpito koetaan vaikeammaksi kuin uuden koodin kirjoittaminen. Tämä johtuu siitä, että vaikka siinä on samoja elementtejä kuin uuden ohjelmiston kirjoittamisessa, niin siinä on lisäksi myös usein toisten kirjoittamien koodien tarkastelua. Vanhaa toisten luomaa koodia voi olla haasteellista ymmärtää. Muita ylläpitoon liittyviä ongelmia ovat koodin huono rakenne, ohjelmiston tai sovellusalueen riittämätön tuntemus, huono dokumentaatio ja ylläpitotehtäviin liittyvä huono maine. Huono rakenteinen koodi johtaa siihen, että ylläpitäjien on vaikea saada selkoa ohjelmiston ja sovellusalueen toiminnasta. Huono dokumentaatio johtaa taas siihen, että ylläpitäjät eivät voi luottaa muihin dokumentteihin kuin koodiin. Kaikkien edellä mainittujen ongelmien myötä ylläpitotehtävillä on huono maine. Usein tehtävissä joutuu tekemisiin asiakkaiden kanssa, jotka eivät ole tyytyväisiä ohjelmistoon. Ylläpitotehtävissä tulisi käyttää ainoastaan kokeneita ja parhaita ohjelmoijia, koska heillä on usein laaja-alaista kokemusta ohjelmistoista. (Harsu 78 - 79.)

Kehittäjien käyttäminen ohjelmiston ylläpitoon voi tuntua hyvältä idealta ja usein niin on tehtykin. Taulukossa 1 on vertailtu kehittäjien käyttämistä ylläpitoon puoltavia ja vastustavia seikkoja. Taulukkoon 2 on kuvattu erillisen ylläpito organisaation käyttämisen hyviä ja huonoja puolia.

Kehittäjien käyttämistä ylläpitoon puoltavat seikat:	**Kehittäjien käyttämistä ylläpitoon vastustavat seikat:**
Kehittäjällä on paras tietotaito järjestelmästä.	Henkilöstö voi jättää kehitysorganisaation, jos työt ovat pelkkää ylläpitoa.
Dokumentaatiota ei tarvitse kehittää.	Uudet työntekijät voivat olla tyytymättömiä ylläpitotehtävien määrään.
Kehittäjien ja ylläpitäjien välille ei tarvitse muodostaa kommunikointi kanavaa.	Jos kehittäjä ekspertti lähtee työstää, voi käydä niin, että ylläpitäjiä ei ole koulutettu riittävästi.

Käyttäjät ovat tekemisessä vain yhden organisaation kanssa.	Kehittäjät parantelevat kehitettyä järjestelmää tarpeettomasti.
Työtehtävät ovat monimuotoisia, joten henkilöstä on tyytyväisempää.	Alkuperäiset kehittäjät saavat usein uusia työtehtäviä korkeamman tason projekteista.

Taulukko 1. Kehittäjien käyttämisen hyvät ja huonot puolet ohjelmiston ylläpidossa. (Pigoski 1997, 21 -22.)

Erillisen ylläpito organisaation käyttämistä puoltavat seikat:	**Erillisen ylläpito organisaation käyttämistä vastustavat seikat:**
Dokumentaatio on parempaa.	Ohjelmistoon siirtyminen voi olla hidasta.
Ylläpitäjät oppivat järjestelmän vahvat ja heikot kohdat.	Kouluttamisen tarve kasvaa.
Tehdään muodolliset menettelyt muutosten toteuttamiseksi.	Voi tulla rahoitusongelmia.
Moraali on parempaa, koska ihmiset, jotka pitävät ylläpito tehtävistä tekevät luultavasti parempaa työtä.	Uuden ohjelmiston oppiminen ja ylläpito-organisaation kehittäminen voi kestää pitkän ajan.
	Käyttäjätuki voi kärsiä ja siitä seuraa tukiorganisaation luotettavuuden kärsiminen.

Taulukko 2. Erillisen ylläpito organisaation hyvät ja huonot puolet. (Pigoski 1997, 23 - 25.)

Ylläpitoon liittyvät ongelmat voidaan ratkaista eri tavoin. Samalla pyritään vähentämään ylläpitokustannuksia. Korjaavan ylläpidon kustannuksia voidaan vähentää parantamalla koodin laatua, kehittämällä testausta ja yhtenäistämällä dokumentointitapoja. Käytäntöjen on oltava vakiintuneita ja standardoituja niin ohjelmoinnissa kuin dokumentoinnissa. Täydentävässä ylläpidossa otetaan käyttäjä paremmin huomioon käyttämällä esim. prototyyppimallia ja osallistuttamalla käyttäjät analyysi- ja suunnitteluvaiheisiin. Ennakoimalla muutoksia jo määrittely- ja suunnitteluvaiheessa voidaan ylläpitoa tehostaa, koska ohjelmista saadaan yleisempiä ja useampiin tilanteisiin sopivia. Edellä mainitut tavat liittyvät kuitenkin varsinaiseen ohjelmistokehitykseen ja on huomioitava, että kaikkea ei siinäkään voida huomioida. Yleensä ylläpidettävyyttä voidaan parantaa takaisinmallinnuksen kautta, joka johtaa uudelleenrakentamiseen ja uudistamiseen. Toinen tapa on parantaa ylläpidettävyyttä hallinnollisilla tavoilla, jolloin käyttöön otetaan yhteiset tavat niin suunnittelussa, ohjelmoinnissa kuin dokumentoinnissa. (Harju 2003, 81.)

Ohjelmien ylläpidossa tehdään muutoksia ohjelmiin. Muutosten tarkoituksena on joko havaitun virheen korjaaminen, toiminnallisuuden lisääminen tai muuttaminen. Ohjelmistoon tehtyjä muutoksia hallinnoidaan muutosten hallinnan kautta. Sen avulla voidaan ohjata ja korjata ohjelmistossa tapahtuvia muutoksia. Muutoksia voidaan esittää muutospyyntödokumenttien kautta. Siinä kuvataan jokin ongelma tai muutostarve ja sen vaikutukset käyttöön. Muutospyyntödokumentin käyttö parantaa kommunikointia ylläpitäjien ja eri sidosryhmien välillä. Muutospyyntödokumentissa on raportoitava tarpeeksi tarkalla tasolla, jotta sen lukijat saavat oikean käsityksen ongelmasta ja tarvittavasta muutoksesta. (Harsu 2003, 82.)

Tunnistusnumero	8603-002
Muutoksen pyytäjä	L.J. Green, NASA
Havaintopäivämäärä	04.03.86
Tavoitepäivämäärä	04.03.86
Ylläpitotyyppi	Korjaava
Muutoksen vakavuus	1
Järjestelmä	Titan Launch
Muutoksen kuvaus	Selostetaan mitä on tehtävä ja miksi on tehtävä.
Muutoksen hyödyt	Mitä tehtävä muutos hyödyttää.
Ongelman alkuperä:	
Järjestelmä	Titan
Ohjelma	Ascent
Moduuli	Initialization
Virheen tyyppi	Virheelliset alkuarvot
Ratkaisu	
Vaikutukset:	
Järjestelmät	Titan
Ohjelmat	Ascent
Moduulit	Initialization
Dokumentointi	Initialization-moduulin määrittelydokumentti
Vaihe	
Aloituspäivämäärä	05.03.86
Hyväksyntä	Tom Prive
Henkilöt	Jay Arthur
Arvioidut resurssit	2 henkilötyöpäivää
Todelliset resurssit	1.5 henkilötyöpäivää

Kuva 4. Muutospyyntödokumentti Harsun mukaan (Harsu 2003, 83).

Kuvassa 4 on kuvattu muutospyyntödokumentti. Se sisältää tietoja, joiden avulla se voidaan tunnistaa, kuten esimerkiksi tunnistusnumero. Muutokselle on ilmoitettava pyytäjän nimi mahdollisten lisätietojen antamisen takia. Lisäksi dokumentissa on oltava ongelman havaintopäivämäärä ja tavoitepäivämäärä, jolloin muutos pitäisi olla valmis. Ylläpito olisi tyypitettävä sen mukaan minkälaista ylläpidollista tehtävää se vaatii, jolloin vaihtoehtoina voivat olla korjaava, mukauttava, täydentävä tai ehkäisevä. Vakavuudella ilmoitetaan se, kuinka kiireellisesti muutos on tehtävä. Vakavuudessa asteikko voi olla esim. neliportainen, jossa yksi (1) tarkoitta kiireellisintä muutostarvetta ja neljä (4) vähäisintä muutostarvetta. Järjestelmä kohtaan merkitään mihin järjestelmään muutos vaikuttaa. Ohjelma kohtaan merkitään mitä ohjelmaa muutos koskee. Muutoksen kuvaus kohdassa annetaan joko toiminnallinen tai tekninen kuvaus halutusta muutoksesta. Kuvauksen on oltava niin tarkka, jotta sen avulla voidaan arvioida tarvittava työ, sen vaikutukset muualle ja laskea tarvittavat resurssit. Muutoksen hyödyt kohdan avulla muutoksia voidaan priorisoida ja järjestää muutosten toteutusjärjestys. (Harsu 2003, 83 - 84.)

Muutospyyntödokumentissa on mainittu myös ongelman alkuperä, jonka avulla ongelmia ja muutoksia voidaan jäljittää. Ratkaisu kohdassa on kerrottu mitä, miten ja miksi muutoksia on tehty. Vaikutukset kohdassa kerrotaan mihin muutokset vaikuttavat ohjelmissa ja se voi sisältää useita eri järjestelmiä sekä ohjelmia. Tämä tieto on tarpeen työmäärän ja resurssien arvioinnissa. Vaiheessa kerrotaan vaiheen aloituspäivämäärä ja se kuka on hyväksynyt muutoksen tekemisen. Henkilöt kohdassa on lueteltu muutokseen osallistuvat henkilöt. Arvioiduissa resursseissa ilmoitetaan alkuperäiset ja tarkistetut kustannukset ja aikataulut. Todelliset resurssit kohdassa nähdään, miten kustannukset ja aikataulut toteutuivat. Näiden tietojen avulla voidaan arvioida myöhemmin samankaltaisia muutostarpeita. (Harsu 2003, 84.)

Muutosvaikutusanalyysin avulla voidaan saada selville muutosten aiheuttamat vaikutukset ja tarvittava työmäärä. Ilman sitä yksinkertaiselta vaikuttava muutos voi osoittautua sellaiseksi, jonka vaikutukset ulottuvat tietokantaan, dokumentaatioon tai muihin ohjelmiin. Tämän vuoksi muutosvaikutusten selvittämisen on oltava laaja-alaista. Selvitys on aloitettava muutoksenpyyntödokumentista, jota ylläpitäjä selittää käyttäen ohjelmistoon liittyviä termejä. Tämän jälkeen käyttäjältä on tarkistettava, että muutoksen kuvaus on ymmärretty oikein. Sen jälkeen voidaan aloittaa muutosten vaikutusten tarkastelu ohjelmiin ja moduuleihin, joissa on kiinnitettävä huomiota rajapintoihin. Muutos voi aiheuttaa uuden tietokentän lisäämisen, joka vaikuttaa ohjelmiston tietorakenteisiin, tiedostoihin tai tietokantoihin. Jos useampi ohjelmisto tai järjestelmä käyttää samaa tietokantaa on myös niiden rakenteet tarkistettava. Muutoksen myötä dokumentit on päivitettävä, jotta ne vastaavat muuttunutta tilannetta. Muutosvaikutusanalyysissa on käsiteltävä myös tarvittavat resurssit. Tämän myötä voidaan laatia aikataulu muutosten toteuttamiseksi, jossa on otettava huomioon kiireellisyysjärjestys. (Harsu 2003, 84 - 86.)

Korjausten tekeminen on aiheellista silloin kun, ohjelmisto ei toimi määrittelyn mukaisesti. Korjaavaa ylläpitoa tarvitaan silloinkin, kun dokumentit eivät vastaa ohjelmiston toimintaa tai käyttöohjeet ovat harhaanjohtavia. Korjaaminen voi kuitenkin aiheuttaa uusia virheitä, jotka jäävät huomaamatta. Usein näin käy silloin, kun testataan ainoastaan sitä ohjelmaa, johon korjaus tehtiin eikä koko ohjelmistoa. Huolellinen testaus koko ohjelmiston osalta vaatii paljon aikaa. Korjaustoiminnot ovat jaettavissa kahteen luokkaan, kiireellisiin ja vuorollaan tehtäviin korjauksiin. Kiireelliset korjaukset ovat sellaisia, joissa korjataan virheitä mitkä voivat aiheuttaa esim. suuria taloudellisia menetyksiä tai sellaisia, joiden korjaamista joukko ihmisiä joutuu odottamaan, jotta he pääsevät jatkamaan töitään. Kiireelliset korjaukset joudutaan tekemään usein nopeasti, joten niiden suunnitteluun ja analysointiin ei voida käyttää tarvittavaa aikaa. Vuorollaan tehtävät korjaukset kohdistetaan niihin virheisiin, joihin ei vaadita välitöntä korjausta. Ne voivat olla myös kiireellisten korjausten uudelleenkäsittelyä, jolloin niiden analysointiin voidaan käyttää tarvittava aika. (Harsu 2003, 86 - 87.) Kustannuksiltaan virheen korjaus on ylläpitovaiheessa huomattavasti kalliimpaa kuin esimerkiksi suunnitteluvaiheessa. Tyypillisesti kustannukset kasvavat mitä myöhemmässä vaiheessa virhe huomataan (Koistinen 2002, 42).

Ennen kuin havaittua virhettä korjataan, niin on tarkasteltava muutoksenpyyntödokumenttia. Siinä oleva ongelma on ymmärrettävä oikein, jotta osataan korjata oikea ongelma. Tämän vuoksi on tärkeää selvittää missä ja milloin ongelma ilmenee. Ongelman etsinnässä käytetään apuna erilaisia dokumentteja, lähdekoodia ja tulosteita sekä ohjelman suorittamista ns. debuggerin avulla. (Harsu 2003, 87.)

Muutos voi johtua myös ominaisuuksien tai toiminnallisuuden lisäämisestä ohjelmistoon. Syyt muutokseen voivat johtua mm. toimintojen automatisoinnista tai uusista vaatimuksista. Tässä tapauksessa ennen muutosta on tarkasteltava muutospyyntödokumenttia ja sitä miten hyvin haluttu muutos sopii vaatimusmäärittelyyn. Tämä vaiheen aikana on tarkasteltava toiminnallisuuteen liittyviä vaatimuksia. Toiminnallisuuden lisääminen vaatii koko ohjelmiston tarkastelua ja mitä muutoksia se vaatii muihin ohjelmiin ja moduuleihin. Kun muutettavat ohjelmat ja moduulit on selvitetty, niin sen jälkeen tarkistetaan niihin liittyvät suunnitteludokumentit, jotka sitten päivitetään vastamaan muuttuvaa tilannetta. Ominaisuuksien lisääminen ylläpidettävään ohjelmistoon on hyvin samankaltaista kuin uuden ohjelmiston kehittäminen. Ylläpidettävässä ohjelmistossa on kuitenkin otettava huomioon käytössä olevan ohjelmiston aiheuttamat rajoitukset ja se, että muutosten on sovittava yhteen keskenään yhteen. (Harsu 2003, 88 - 89.)

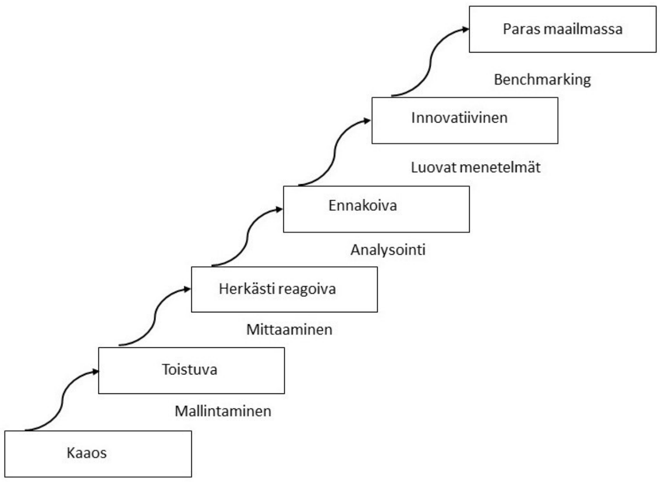
Yksittäiseen ylläpitotehtävään voi kulua aikaa, niin että kokonaisajasta 50 % menee koodin ymmärtämiseen, tällöin ylläpitäjä etsii korjattavaa kohtaa ja tekee sitten muutokset. Muutoksen testaamiseen menee 25 % ja suunnitteluun kuluu aikaa 10 %. Lopuksi toteutukseen kuluu aikaa 10 % ja dokumentointiin 5 %. (Koistinen 2002, 43.)

Muutosten hallinta voi tuottaa vaikeuksia ylläpitäjille, koska muutosvaatimuksia voi tulla eri puolilta käyttäjäorganisaatiota. Tämä voi johtaa siihen, että kehittämistarpeet eivät ole nipussa, vaan ne on toteutettu yksitellen. Muutostarpeet voivat olla myös ristiriidassa keskenään. Jos vaatimuksia toteutetaan yksitellen, niin myöhempi muutos voi muuttaa tai jopa poistaa aiemmin tehdyn muutoksen. Nämä tilanteet voivat aiheuttaa käyttäjissä hämmennystä. (Koistinen 2002, 51.) Ylläpitäjien olisi myös tunnettava käyttäjien toimintaa ja ymmärrettävä toiminnan tavoitteet ja sen prosesseja (Koistinen 2002, 53).

2.2 Prosessit ja niiden kehittäminen

Prosessi on määritelty niin, että siinä on toisiinsa loogisesti liittyvä toimintoja ja siihen liittyvän toiminnan tuloksen aikaan saavat resurssit sekä tulos (Laamanen 2001, 19). Projektia ja prosessia ei pidä sekoittaa keskenään, koska projekti on ainutkertainen ja prosessi on toistuva. Prosessi on ulkopuolelta katsottuna laatikko, jossa syötteitä jalostamalla saadaan tuloksia. (Lecklin 2006, 124.) Harrington (1991, 9) jakaa prosessit tuotanto- ja liiketoimintaprosesseihin. Tuotantoprosessiksi käsitetään ne prosessit, joissa tuote toimitetaan ulkoiselle asiakkaalle. Liiketoimintaprosesseiksi käsitetään kaikki tuki- ja muut prosessit, jotka koostuvat joukosta tehtäviä, jotka käyttävät resursseja organisaation tavoitteiden saavuttamiseksi. (Harrington 1991, 9). Toisaalta Lin, Yang ja Pai ovat sitä, mieltä että erityisesti liiketoimintaprosessi käsittää viisi kohtaa. Prosessilla on asiakkaita, ja se koostuu toiminnoista. Toimintojen on tuotettava arvoa asiakkaille ja toiminnon ohjaamisesta vastaavat joko ihmiset tai koneet. Prosessi ulottuu useille eri osastoille, jotka ovat vastuussa prosessista. (Lin, Yang & Pai, 2002, 3.) Prosessin asiakkaat voivat olla joko ulkoisia tai sisäisiä asiakkaita. Ulkoinen asiakas on yrityksen ulkopuolinen asiakas, joka voi olla joko välitön tai välillinen. Välitön asiakas on asiakas, joka on suoraan yhteydessä organisaatioon ja tilaa tuotteen tai palvelun. Välillinen asiakas käyttää organisaation tuotteita tai palveluita ja on usein välittömän asiakkaan asiakas. Sisäinen asiakas on prosessiketjussa toimiva prosessivaiheen käsittelijä. (Lecklin 2006, 79 - 80.) Tässä yhteydessä prosessilla tarkoitetaan liiketoimintaprosessia.

Prosessien avulla voidaan järjestää asioita. Niiden tunnistaminen ja kuvaaminen auttaa ymmärtämään kokonaisuutta. Kuvaamisen avulla voidaan esittää organisaatiossa tehtävää käytännön työtä. Prosessien jäsentämisen kautta voidaan varmistaa, että kehittämistyö kohdistuu niin, että se hyödyttää organisaatiota. (Laamanen 2001, 23.) Prosessien avulla voidaan kehittää organisaatiota, joka toimii tukirankana toimintaan kohdistuville vaatimuksille, tukirankaa on kuvattu kuvassa 5 (Laamanen 2001, 39).



Kuva 5. Prosessien kehittyminen kohti maailman parasta prosessia Laamasen mukaan (Laamanen 2001, 44).

Prosessien kehittymistä on kuvattu, siten että matalimmalla tasolla toiminta on kaoottista ja asioita tehdään sitä mukaa kuin niitä tulee eteen. Tehdyt ratkaisut ovat ainutlaatuisia ja sovittuja sääntöjä on vähän. Tämä mahdollistaakin yksilöiden toiminnan omien intressiensä mukaan. Se ei kuitenkaan ole organisaation kannalta hyvä asia, koska koko organisaation laajuiset tulokset vaativat yhteistyötä. (Laamanen 2001, 44.)

Toisessa tasossa tunnistetaan toimintamalleja, jotka ovat toistuvia. Niiden myötä luodaan sääntöjä ja kuvataan eri prosesseja. Toiminnan toistuvuutta on vaikea käsitellä ja analysoida ilman niistä tehtyjä kuvauksia. Vaarana on kuitenkin, se että kuvauksia, ohjeita ja sääntöjä tehdään, vaikka ne eivät liittyisi mitenkään toiminnan tuloksiin. Siitä voi seurata se, että ongelmia yritetään ratkaista tekemällä lisää sääntöjä. Toisena ongelmana Laamanen näkee sen, että prosessien kehittäminen jää tähän tasoon. (Laamanen 2001, 44 - 45.)

Kolmannessa tasossa mitataan prosessien suorituskykyä, joka on tunnettava ennen kuin mittaamista voidaan suorittaa. Mittaustuloksista on hyötyä vain, jos prosessi on tunnettu, koska silloin tiedetään mitä on parannettava. Neljännessä tasossa voidaan tehdä ennakointia, koska mittaaminen mahdollistaa kehityskulun analysoinnin. Tarvittaessa tällöin voidaan tehdä reagointia tilanteisiin, ilman että organisaatiolle vakavia kriisejä pääsee syntymään. (Laamanen 2001, 45.)

Innovatiivisessa tasossa prosesseista saatua informaatiota osataan käyttää aikaisempaa paremmin hyväksi, koska analysointitaidot kehittyvät. Analysoinnilla voidaan edistää joustavuutta, herkkyyttä ja uusien mahdollisuuksin hyödynnettävyyttä. Innovatiivisilla prosesseilla voi olla hyvä tai jopa ainutlaatuinen tuloksentekokyky ja parhaimmillaan nämä prosessit voivat maailman parhaita. Organisaatiot, joiden prosessit ovat parhaita tunnetaan alalla hyvin ja niiden toimintaa voidaan yrittää kopioida. Organisaatiot toimivat siten, että jokaisella osastolla on omat tehtävänsä ja tavoitteensa. Osastot pyrkivät tehostamaan ja kehittämään omaa toimintaansa, vaikka erityisesti liiketoimintaprosessit vaativat osastojen välistä yhteistyötä. Osastojen väliset rajapinnat ovat siis tärkeitä, jotta prosessit olisivat tehokkaita läpi koko organisaation. (Laamanen 2001, 45; Lecklin 2006, 124 - 125.)

Prosessien rajaus ja luokittelu

Prosessin tunnistamiseen ei välttämättä kiinnitetä paljoakaan huomioita. Tunnistamisen yhteydessä päätetään kuitenkin prosessin kannalta merkittäviä ratkaisuja, kuten mistä prosessi alkaa ja mihin se loppuu, miten se luokitellaan, nimetään sekä mitä elementtejä siitä kuvataan. Prosessin alkamisen ja loppumisen yleisenä periaatteena pidetään, että se alkaa asiakkaasta ja päättyy asiakkaaseen, jotta prosessiketju säilyy eheänä organisaation sisällä. Prosessi on rajattava siten, että se alkaa suunnittelusta ja päättyy arviointiin, jotta organisaatio toimii jatkuvan kehittämisen periaatteen mukaisesti. Usein organisaatiot eivät kuitenkaan sisällytä suunnittelu- ja arviointivaiheita prosessiin, koska niitä vaiheita pidetään irrallisena työnä. (Laamanen 2001, 52 - 53.)

Prosessin luokittelussa tehdään valinta, siitä onko prosessi ydin- vai tukiprosessi. Ydinprosessin tunnistaa siitä, että se synnyttää organisaation jalostusarvon ja siitä on suora yhteys ulkoiseen asiakkaaseen. Organisaation ydinprosesseja ovat tällöin mm. tuotteen kehittäminen, asiakkaan vakuuttaminen, tuotteen toimittaminen ja asiakkaan tyytyväisyyden ylläpitäminen. Ydinprosessi kulkee läpi organisaation ja tällainen prosessi on esimerkiksi tilaus-toimitusketju. Lisäksi ydinprosessi sisältää enemmän ydintoimintoja, kuin niitä on sitä tukevissa tukiprosesseissa. (Laamanen 2001, 56; Kiiskinen, Linkoaho & Santala 2002, 28.)

Organisaation toiminta tarvitsee ydinprosessien lisäksi tukiprosesseja, jotka luovat edellytykset tehokkaalle toiminnalle. Ne ovat organisaation sisäisiä ja niihin kuuluvat esim. henkilöstöhallinto, taloushallinta sekä tietohallinto. Prosessien nimeämisessä on kiinnitettävä huomiota nimeämiskäytäntöön, sillä kuvaukset ja nimet ovat tärkeitä viestinnän välineitä. Niiden avulla ymmärretään toiminnan tarkoitus, tavoitteet ja tuloksia. (Laamanen 2001, 56 - 58.)

Prosessien tunnistaminen aloitetaan analysoimalla joko toimintaa, menestystekijöitä tai asiakkaan prosesseja. Toiminnan analysoinnissa tutkitaan organisaation toimintaa, josta nousee helposti esiin erilaiset toiminnalliset prosessit kuten tuotekehitys. Tämä voi johtaa kuitenkin harhaan, koska sisäisten prosessien tutkiminen ei ratkaise osastojen välisiä yhteistyövaikeuksia. Menestystekijöiden analysointi on vaikeaa, koska niitä ei tunnisteta. Lisäksi menestystekijöiden analysointi tuottaa listan asioista, jotka vaikuttavat tehokkuuteen ja niiden pohjilta onkin haastavaa hahmottaa prosesseja. Asiakkaiden prosessien analysoinnin pohjalta voidaan organisaation prosessit asettaa palvelemaan mahdollisimman hyvin asiakkaan toimintaa. Tällöin asiakkaan ja organisaation prosessit niveltyvät toisiinsa. (Laamanen 2001, 64 - 65.) Prosessien kohdalla on otettava huomioon, että asiakkaat voivat olla joko organisaation sisäisiä tai ulkoisia (Kiiskinen ym. 2002, 28).

Prosessikartan tarkoituksena on olla viestinnän väline, joka auttaa ymmärtämään toimintaa. Prosessikartassa kuvataan mistä palvelut ja tuotteet tulevat. Prosessikartassa tulisi esittää asiakkaan toiminta, jotta organisaation jäsenillä olisi ymmärrystä asiakkaan toiminnasta. Sen tulisi kuvata oman organisaation toimintaa siten, että toiminnan luonne ymmärretään ja prosessit vaikuttavat toisiinsa. Tämän takia keskeiset vaikutussuhteet on kuvattava prosessien verkkona. (Laamanen 2001, 59 - 61.)

Prosessin nimi ei välttämättä kerro paljoa prosessista, koska ihmiset voivat käsittää nimen eri tavoin. Tällöin rajauksella voidaan sopia mitä prosessi sisältää. Rajauksessa on esitettävä asiakkaat, tuotteet, prosessin vaiheet, syöte ja toimittajat. Rajauksen myötä opitaan ymmärtämään prosessikarttaa. Rajauksia arvioitaessa on kiinnitettävä huomiota prosessikartan eheyteen, eli syötteille ja tuotteille on löydettävä vastineet eri prosessien välillä ja prosessien tulisi alkaa ja loppua aina asiakkaaseen. (Laamanen 2001, 66 - 67.)

Prosessin kuvaaminen

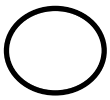
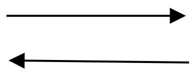
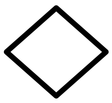
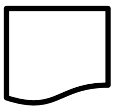
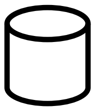
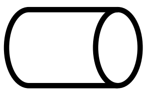
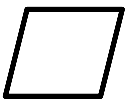
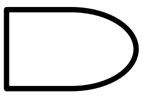
Prosessin kuvaamisella voidaan kuvata organisaation toimintaa ja sitä tarvitaan kriittisten vaiheiden tunnistamiseen. Laamasen mukaan hyvässä prosessi kuvauksessa esitetään kriittiset asiat ja asioiden väliset riippuvuudet. Lisäksi se auttaa ymmärtämään kokonaisuutta ja sitä miten oma rooli auttaa tavoitteiden saavuttamisessa. Se edistää ihmisten välistä yhteistyötä ja antaa mahdollisuuden toimia joustavasti tilanteiden vaatimusten mukaan. Teknisesti kuvauksen on oltava lyhyt, joka sisältää tekstiä ja vuokaavion sekä tunnistetiedot. Sen oltava sovitun rungon ja prosessikaavion mukainen, ymmärrettävä, looginen eikä se saa sisältää ristiriitoja. (Laamanen 2001, 75 - 76.)

Ennen prosessien kuvaamista, valitaan prosessien kuvaamistapa ja mitä prosesseja kuvataan. Kuvattavaksi valitut prosessit nostavat niiden esittämien aiheiden merkitystä organisaatiossa. Kuvaaminen sisältää organisaation menestykselle tärkeät seikat. Prosessikuvauksessa käydään Laamasen mukaan läpi soveltamisala, asiakkaat, tavoitteet, syötteet, tuotteet ja palvelut sekä prosessikaaviot ja vastuut. (Laamanen 2001, 77 - 78.)

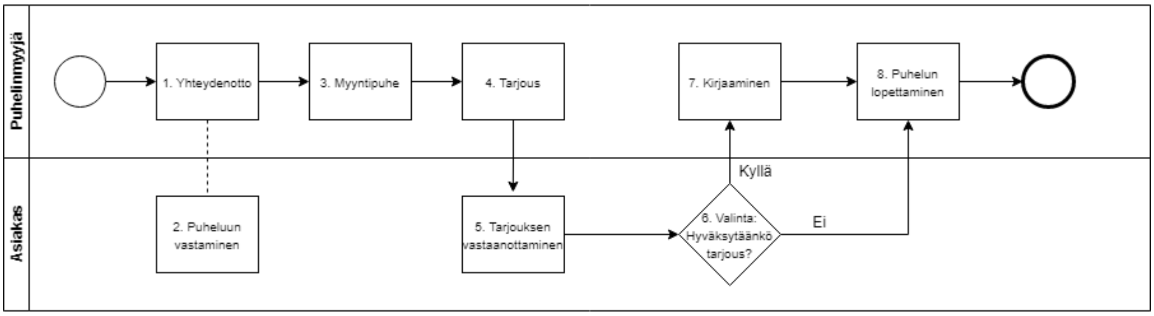
Prosessikaavion malleja on erilaisia ja niiden kaikkien tarkoituksena kuvata toimintaa. Toiminnan ymmärtämiseen käytetään karkeaa kuvausta. Jos halutaan parantaa toimintaa, niin se vaatii yksityiskohtaisen kuvauksen parannettavasta kohteesta. Kaaviossa on kuvattava asiakas, jottei kehitystyö käänny sisäänpäin. Ydinprosessien osalta asiakkaat ovat helpommin tunnistettavissa kuin tukiprosesseissa. Asiakkaan valinta tukiprosessin osalta vaikuttaa merkittävästi prosessin suunnitteluun ja Laamasen mukaan asiakkaaksi kannattaa valita osaston tms. sijaan rooli. (Laamanen 2001, 79 - 80.)

Yleisesti käytetty malli on kuvata toimijoiden roolit hierarkkisesti, jolloin ylimmäksi on sijoitettu toimitusjohtaja ja alimpana on sitten varsinaiset työn suorittajat. Laamasen mielestä tämä mallin huonona puolena on se, että siinä kuvataan prosessia hyvin esimieslähtöisesti ja se voi vähentää organisaatiossa itseohjautuvuutta, aloitteellisuutta ja joustavuutta. Tehtävät kuvataan neliöillä ja etenemistä nuolella. Asiakkaan toimintaa kuvataan soikiolla. Prosessikaaviota kuvatessa on tärkeää, että kuvataan organisaation toimintaa, eikä pelkkää tiedonkulkua. Toiminta on kuvattava sellaisella tarkkuudella, että siitä käy prosessin toimintalogiikka selville. Prosesseihin voi liittyvä paljon erilaisia tehtäviä, ja jos kaaviolla pyritään ymmärtämään toimintaa, niin silloin kaikkia tehtäviä ei ole tarpeen kuvata. Jos kyse on sen sijaan esim. tietojärjestelmän kehittämisestä, niin silloin voidaan tarvita yksityiskohtaisia kuvauksia tehtävistä. Hyvänä sääntönä onkin tunnistaa kriittiset toiminnot ja edistää niiden toimintaa. (Laamanen 2001, 80 - 81.)

Ihmisen hahmotuskyvyn rajallisuuden vuoksi toimintoja ei kannata sijoittaa liian montaa yhteen kaavioon, maksimissaan toimintoja voi olla korkeintaan 15 - 20. Liian yksityiskohtaisesti kuvattu prosessi aiheuttaa katsojassa ymmärryskyvyttömyyttä. Sen vuoksi prosessi on kuvattava ensin karkeasti. Prosessin kuvauksessa on neljästä seitsemään vaihetta. Karkean kuvauksen avulla saavutetaan ymmärrystä toiminnoista. Tämän jälkeen voidaan edetä yksityiskohtaisempiin kaavioihin. Prosessikaaviossa kuvattujen tehtävien tulisi olla samantasoisia. Tiedonkulkukuvauksen avulla voidaan, josta nähdään mitä tietoa kuhunkin vaiheeseen kuuluu. (Laamanen 2001, 81; Lecklin 2006, 141.) Yksityiskohtainen prosessikuvaus on mahdollista tehdä usealla eri kuvaustavalla. Yleisimpiä ovat vuokaavio, tehtävämatriisi ja uimaratakaavio sekä prosessin tekstimuotoinen kuvaus. Vuokaaviossa ja uimaratakaaviossa käytetään varsin vakiintuneita merkintätapoja, joita on esitetty taulukossa 3. Uimaratakaaviossa jokainen rooli esitetään omalla radallaan, jossa kyseisen roolin tehtävät suoritetaan. Esimerkkikuvaus uimaratakaaviosta on kuvassa 6. (Martinsuo & Blomqvist 2010, 11 - 12.)

Merkintä	**Merkitys**
	Aloitus tai lopetus
	Tehtävä tai prosessi
	Materiaali- tai tietovirta
	Päätös
	Dokumentti
	Tietojärjestelmä / Varasto
	Varasto
	Data
	Viive, odotus

Taulukko 3. Vuokaavion ja uimaratakaavion keskeiset merkintätavat (Martinsuo ym. 2010, 11).



Kuva 6. Esimerkki uimaratakaaviosta, jossa rooleina on puhelinmyyjä ja asiakas.

Näkökulmia prosessien kehittämiseen

Kehitettävät prosessit valitaan joko keskustelujen pohjalta, joissa todetaan yhdessä mitkä prosessit ovat tärkeitä tai valintamatriisin kautta. Valintamatriisista valitaan kehitettäväksi ne prosessit, joilla nähdään olevan suurin kehittämispotentiaali. Valinta voidaan tehdä myös siten, että valinta perustuu joko strategiaan tai menestystekijöihin, jolloin valitaan ne prosessit, jotka ovat kriittisiä joko strategian tai menestystekijän toteutuksessa. (Laamanen 2001, 83.)

Lecklinin mukaan asiakkaat ovat tärkeitä palautteen antajina, jolloin asiakkaan antaman palautteen perusteella voidaan valita kehitettävä prosessi (Lecklin 2006, 141). Lisäksi kehitettävän prosessin valintaan voi vaikuttaa resurssien rajallisuus, jolloin kehitettävä prosessi on valittava esimerkiksi erityisen kuntotestin avulla. Kuntotestin mittarit on ominaisuuksiltaan painotettava erilaisiksi ja sen jälkeen arvioidaan prosessit arvosanoin, jolloin huonoimman arvosanan saanet prosessit nousevat esiin. (Lecklin 2006, 146 - 147.) Näistä prosesseista voidaan valita sitten ne, jotka ovat tärkeitä ja joissa voidaan saavuttaa huomattavia parannuksia kohtuullisilla panostuksilla (Kvist, Arhonmaa, Järvelin & Räikkönen 1995, 72).

Prosessien kehittäminen voidaan aloittaa myös silloin, kun organisaatioihin hankitaan esim. tietojärjestelmä. Se vaatii usein toiminnan määrittelyä, jotta hankittu järjestelmä voidaan sovittaa organisaation toimintaan. Samalla se tuo mahdollisuuden parantaa organisaation prosesseja, mutta joskus voi käydä myös toisin. Prosessien kehittäminen voidaan aloittaa myös silloin, kun on havaittu ongelma. Tällöin ongelmasta tehdään prosessikuvaus, joka voi puutteellisesti tehtynä johtaa siihen, että tehdään vain ns. osaoptimointi eikä paranneta koko prosessin suorituskykyä. Kolmantena vaihtoehtona tietoisasti parantaa organisaation suorituskykyä. Silloin organisaatiosta tunnistetaan ne prosessit, joiden suorituskykyä halutaan parantaa. (Laamanen 2001, 202.) Harringtonin (1991, 37) mukaan on viisi seikka, jotka kehittävien prosessien valinnassa olisi huomioitava:

- Asiakasvaikutus: Hyötyykö asiakas prosessin kehittämisestä?
- Muutettavuus: Onko kehittäminen realistista?
- Suorituskyky: Miltä prosessin suorituskyky vaikuttaa tarkasteluhetkellä, onko se huono?
- Vaikutus liiketoimintaan: Miten tärkeää prosessi on liiketoiminnan kannalta?
- Vaikutus työhön: Onko kehittämiseen käytettävissä resursseja ja mitä ne ovat?

(Harrington 1991, 37.)

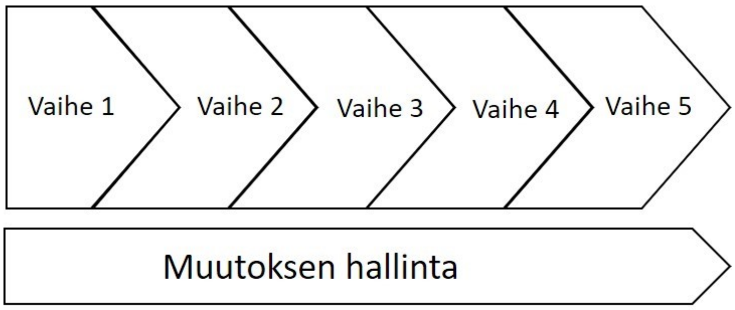
Prosessien parantaminen voidaan jakaa niiden luonteen perusteella reagoivaan, ennakoivaan ja innovatiivisen parantamiseen. Reagoivassa parantamisessa joku huomaa jotakin tapahtuvan ja reagoi siihen ryhtymällä toimenpiteisiin. Käytännössä tämä voi olla ero asetetun ja käytännössä olevan suorituskyvyn välillä. Ennakoivassa parantamisessa prosesseja parannetaan tutkimalla trendejä sekä arvioimalla tulevaisuutta. Tässä mallissa kehitystyöhön ryhdytään ennen kuin prosessin suorituskyky vaikuttaa organisaation tulokseen. Innovatiivisessa parantamisessa pyritään löytämään jatkuvasti uusia ratkaisuja. Tässä mallissa tavoitteet voidaan asettaa korkeiksi, jolloin organisaatio on pakotettu löytämään uusia keinoja toimintojen hoitamiseksi. (Laamanen 2001, 205 - 206.) Haasteellista on myös saada henkilöstö ja organisaatioin johto uskomaan omiin mahdollisuuksiinsa prosessien kehittämisessä, koska virheellisesti uskotaan, että prosessin parantamien ei ole mahdollista (Kvist ym. 1995, 65).

Prosessien kehittämismalleja

Prosessien kehittäminen voidaan tehdä joko ihmisten tai järjestelmien näkökulmasta. Kehittämisessä on kuitenkin hyvä huomioida molempia näkökulmia. (Laamanen 2001, 209.) Toiminnan kehittäminen keskitetään niihin prosesseihin, joista yrityksen tuotteet ja palvelut syntyvät (Lecklin 2006, 134). Resurssien asettamien rajoitteiden takia ei ole kuitenkaan mahdollista kehittää kaikkia prosesseja täysipainoisesti yhtä aikaa. Alussa on syytä keskittyä muutamaan prosessiin, joilla saadaan näkyviin konkreettisia parannuksia. Tämän avulla myös muutosvastarinta organisaatiossa pienenee. (Kvist ym. 1995, 72.)

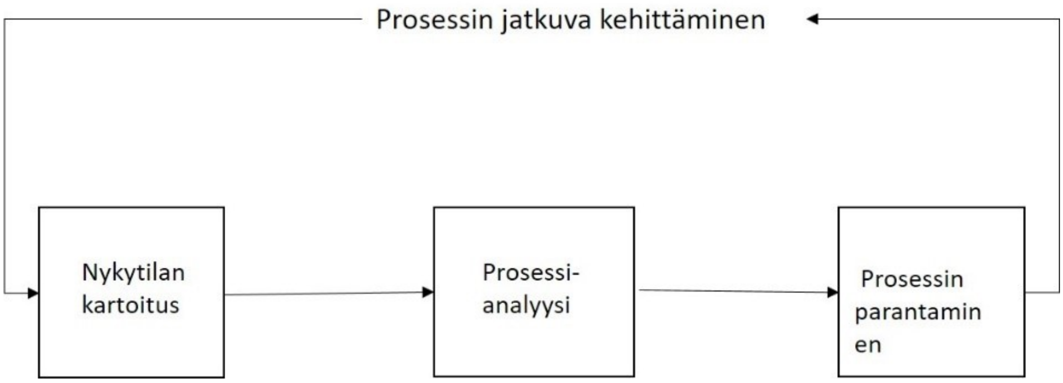
Laamasen mukaan on tunnistettavissa kolme erilaista perustyyppiä, joiden avulla prosesseja voidaan kehittää. Ensimmäisenä tyyppinä on Laamasen mukaan prosessin suunnittelu ja suorituskyvyn parantaminen. Toisena tyyppinä on ongelman ratkaisu ja kolmantena tyyppinä on benchmarking, eli parhaiden käytäntöjen jakaminen. Jokainen näistä tyypeistä sisältää kuvaamisen, mittaamisen ja analysoinnin sekä ratkaisujen testaaminen. (Laamanen 2001, 209.)

Kiiskisen, Linkoahon ja Santalan (2002, 37 - 39) mukaan prosessin kehittäminen voidaan jakaa viiteen eri vaiheeseen, jossa ensimmäisessä vaiheessa asetetaan odotukset organisaation johdon puolesta ja annetaan kehittämiselle hyväksyntä. Vaiheet on kuvattu kuvassa 7, jossa nähdään, että muutoksen hallinnan on oltava jatkuvaa läpi koko prosessin kehittämisen. Ensimmäisessä vaiheessa on perusteltava syyt muutokselle. Toisessa vaiheessa analysoidaan nykyiset prosessit ja samalla täsmennetään muutostarve ja sen kohteet. Kolmannessa vaiheessa verrataan nykytilaa visioon. Neljännessä vaiheessa määritellään uusi toimintamalli ja viidennessä vaiheessa toteutetaan muutostoimenpiteet. (Kiiskinen ym. 2002, 37 - 39).



Kuva 7. Prosessin kehittämisen vaiheet mukaillen Kiiskistä, Linkoahoa ja Santalaa (Kiiskinen ym. 2002, 38).

Toisena prosessin kehittämisen mallinna on ns. kolmevaiheinen kehittämismalli. Siinä kehittäminen aloitetaan nykytilan kartoituksella. Sen aikana tehdään prosessityön organisointi, laaditaan niin prosessikuvaukset kuin prosessikaaviot sekä tehdään arviointi prosessin toimivuudesta. Prosessianalyysi vaiheessa selvitetään ongelmat ja ratkaistaan ne. Samalla pohditaan myös erilaisia kehittämisvaihtoehtoja sekä lopuksi valitaan kehittämistapa. Prosessin parantamisen vaiheessa uudistettu prosessi otetaan käyttöön parannussuunnitelman mukaisesti. Jatkuva kehittäminen palauttaa tilanteen alkuun ja prosessin toimivuutta voidaan arvioida säännöllisesti. Tarvittaessa kehittäminen voidaan käynnistää uudelleen ja aloittaa kehittäminen uudelleen. Kuvassa 8 kuvataan 3-vaiheinen kehittämismalli. (Lecklin 2006, 134 - 135.)



Kuva 8. Prosessien jatkuva kehittäminen Lecklinin mukaan (Lecklin 2006, 134).

Adesola ja Baines (2005, 10) ovat kehittäneet prosessin kehittämiseen seitsemänvaiheisen mallin, joka on nimetty integroiduksi ja mallipohjaiseksi prosessinparannus metodiksi. Ensimmäisessä vaiheessa on ymmärrettävä tarpeet. Siinä on muodostettava ymmärrys tilanteesta ja käytettävä siihen esim. SWOT-analyysiä. Toisessa vaiheessa on ymmärrettävä prosessi mallintamalla se sellaisena kuin se on nykytilassa. Kolmannessa vaiheessa nykyistä prosessia on analysoitava ja mitattava sen tehokkuus. Neljännessä vaiheessa prosessia kehitetään siihen suuntaan kuin sen halutaan tuottavan tuloksia. Tässä vaiheessa on arvioitava uudelleen suunnitellun prosessin tehokkuus. Viidennessä vaiheessa suunnitellaan uuden mallin käyttöönottoa, tässä vaiheessa on mm. henkilöstöä on koulutettava. Kuudennessa vaiheessa arvioidaan uuden prosessin suorituskykyä erilaisin mittauksin. Seitsemännessä vaiheessa arvioidaan uutta prosessia, jotta saavuttiko se asetetut tavoitteet. Jatkuvan kehittymisen mallin mukaan lopuksi palataan vaiheeseen yksi, jolloin tämä malli toimii samalla tavalla kuin kolmevaiheinen malli. (Adesola & Baines 2005, 10.) Tässä mallissa vaiheiden määrä on suurempi verrattuna aiemmin esiteltyyn kolmevaiheiseen malliin, ollen näin tarkemmin eritelty versio kolmivaiheisesta kehittämismallista.

2.3 Asiakassuhteiden hallinta

Tässä yhteydessä keskitytään asiakassuhteiden hallintaan prosessin näkökulmasta, jolloin käsitellään asiakaspalveluun liittyviä näkökohtia. Tässä yhteydessä asiakkaat ovat ulkoisia asiakkaita.

Asiakassuhteiden hallinnasta ja hoidosta puhutaan usein käsitteellä CRM – Customer Relationship Management, joka on käännetty suomen kieleen usein muotoon asiakashallinta, asiakassuhteen johtaminen tai asiakkuuksien johtaminen. CRM on kuitenkin termi, jolla on monta eri ulottavuutta. Se voi olla käsite toimintatavoille ja tietojärjestelmille, joiden avulla hallitaan organisaatioiden asiakkuuksia. CRM on myös prosessi, jolla hallitaan asiakaskohtaamisia, eli mm. palvelutapahtumia, kuten markkinointia, myyntiä ja asiakaspalvelua. (Oksanen 2010, 21 - 22.) Peppers ja Rogers määrittelevät CRM:n niin, että se edellyttää eri asiakkaita kohdeltavan eri tavalla, jotta jokaisen asiakkuuden arvoa saadaan nostettua (Peppers & Rogers 2004, 6). Grönroos esittelee asiakkaan määritelmän kahdella eri tavalla, joista vaihtokeskeisessä mallissa asiakas on asiakas vain silloin kun hän on markkinointi- tai myyntitoimenpiteiden kohteena. Toisessa mallissa asiakkuus on jatkuva prosessi ja sen myötä suhde on voimassa myös silloin kun siihen ei liity vastikkeellisia tapahtumia. (Grönroos 2009, 63.) Asiakassuhteen hallinnan myötä yritys saa lisää ymmärrystä asiakkaiden ostojen syistä, samalla se vahvistaa myös myynnin ja markkinoinnin vaikuttavuutta (Mäntyneva 2000, 12). Peppersin ja Rogersin kirjassa Kotlerin näkemys asiakassuhteen hallinnasta on se että, tarkoituksena ei ole pelkästään tarjota erinomaisia tuotteita tai palveluita vaan, hankkia, pitää ja kasvattaa erinomaisia asiakkuuksia. Lisäksi tarkoituksen mukaista on keskittyä asiakkuuksien säilyttämiseen ja niiden arvojen nousuun, eikä haalia kaikenlaisia asiakkaita suurilla kustannuksilla. (Peppers ym. 2004, 13.)

Asiakassuhde

Liiketoimintaan kuuluu asiakas ja palveluntarjoaja. Asiakkaan ja palveluntarjoajan välistä toimintaa kutsutaan suhteeksi. Suhteeseen liittyy palvelutapahtumia, jossa asiakas käyttää palveluntarjoajan tuotteita tai palveluita. (Grönroos 2009, 30.) ITIL määrittelee palveluntarjoajan organisaatioksi, joka tarjoaa palveluitaan yhdelle tai useammalle sisäiselle tai ulkoiselle asiakkaalle (Hunnbeck, Rudd, Lacy & Hanna 2011, 15). ITIL on kokoelma IT-palveluiden hallintaan ja johtamiseen tarkoitetuista käytännöistä. Se peräisin Isosta-Britanniasta ja sen kehitys on aloitettu 1980-luvulla. (Wikipedia 2021b). Palvelutapahtuma on prosessi, jossa molemmat osapuolet ovat vuorovaikutuksessa. Vuorovaikutus voi tapahtua, niin että palveluntarjoaja on siinä läsnä kuten esimerkiksi ravintolassa, jossa tarjoilija on suorassa vuorovaikutuksessa asiakkaan kanssa. Vuorovaikutus voi tapahtua myös siten, että palveluntarjoaja ei ole läsnä, mutta tarjoaa prosessin perusedellytykset puhelinsoitolle, kuten esimerkiksi matkapuhelinoperaattori. Asiakkaan ja palveluntarjoajan välinen suhde kehittyy yksittäisten tapahtumien kautta. Useammat tai jatkuvat tapahtumat luovat edellytyksiä suhteen jatkumiselle ja syventymiselle. Asiakassuhde lujittuu, jos asiakas kokee saavansa palvelutapahtumista jotain erityisen arvokasta. Asiakkaat ovat kuitenkin harvoin valmiita maksamaan siitä enempää kuin he kokevat sillä olevan arvoa. Tämän vuoksi palveluntarjoajan onkin yritettävä löytää ratkaisu palvelutapahtumien personointiin ilman lisäkustannuksia. (Grönroos 2009, 30; Mäntyneva 2000, 10.)

Erälinnan on väitöskirjassaan lainannut Zablahia, jonka mukaan kestävien asiakasliikesuhteiden kehittäminen ja säilyttäminen ovat yritykselle tärkeitä prosessiajattelussa. Sen mukaan yrityksen toiminnot voidaan pilkkoa ja kasata erilaisiksi prosessiketjuiksi asiakassuhteen elinkaaren mukaan. Tällöin on huomioitava asiakkaan muuttuvat asiakastarpeet, jotta niihin voidaan vastata asiakaskohtaisesti optimoiduilla prosessiketjuilla. Näin toimiessa organisaatio takaa asiakassuhteen jatkumisen ja oman tuloksensa. (Erälinna 2013, 19.) Yritykset, jotka ovat omaksuneet asiakassuhteiden hallinnan ja siihen liittyvät tietojärjestelmät ovat saaneet siitä etua parantuneen asiakaskuvan ja työntekijöiden tuottavuuden kautta. Sen myötä asiakastyytyväisyys ja yritysten oma liikevaihto on kasvanut. (Abdul-Muhmin 2012, 94 - 95).

Asiakkaan tarpeiden tunnistaminen

Asiakkaiden odotukset muuttuvat jatkuvasti. Tällöin esimerkiksi aiemmin asiakkaalle riittänyt palvelutapahtuman taso ei välttämättä riitäkään enää tänä päivänä tyydyttämään asiakkaan odotuksia. Asiakkaan kokemaa odotusarvoa muuttaa tiedon saatavuus ja läpinäkyvyys sekä teknologian kehitys. Läpinäkyvyyden myötä eri kanavat tarjoavat asiakkaalle tietoa muiden asiakkaiden kokemuksista ja asiakas vertaa niitä omiin kokemuksiinsa. Asiakkaan odotukset ylittävä kokemus, eli ns. ylipalvelu voi johtaa siihen, ettei yritys voi pitää haluttua palvelutasoa yllä, koska asiakkaat jakavat kokemuksiaan keskenään. Tämä voi tuottaa siten asiakkaille pettymyksiä, vaikka aiempi kokemus olisi ollut riittävä. (Gerdt & Eskelinen 2018, 71.) Yrityksen menestyminen johtuu Erälinnan mukaan Zablahia mukaillen siitä, miten hyvin yritys osaa havaita asiakkaan muuttuvat tarpeet ja vastata sitten niihin (Erälinna 2013, 20). Asiakkaat ovat olleet aina hyvin innokkaita tarjoamaan parannusehdotuksia ja uusia ideoita yrityksille (Evans & Mckee 2010, 230).

Asiakkaan tunnistamisen lisäksi yrityksen on tunnistettava myös asiakkaan tarpeet. Asiakas tekee hankintansa tarpeensa tyydyttämiseksi. Asiakas tekee hankinta päätöksen sen perusteella mitä myyntilupauksia tuotteesta on annettu, ja vertaa niitä omiin tarpeisiinsa. Samalla asiakas huomioi myös omat varansa ja sen, miten tuotteesta saadaan hyötyä. Valintatilanteessa vaikuttaa myös muut tekijät, kuten odotukset, vaatimukset, toiveet ja aiemmin saadut kokemukset sekä arvot. Lecklinin mukaan asiakkaalla on oma arvomaailma, johon hankittava tuote tai palvelu sijoittuu. (Lecklin 2006, 84.) Niinpä yrityksen olisi luovuttava yksipuoleisesta viestinnästä ja pyrkiä asiakkaan kanssa kaksisuuntaiseen dialogiin, jossa asiakas voi kertoa omista tarpeistaan (Mäntyneva 2000, 10).

Asiakkaiden arvojen tunteminen ja niiden käyttäminen mahdollistaa yrityksen oman liiketoiminnan menestyksen. Asiakkaiden arvoja voidaan määrittää arvojen määritysprosessin kautta, jonka tuloksena laaditaan toimintasuunnitelma. Toimintasuunnitelman tarkoituksena on saada yrityksen tuotteet sopimaan paremmin asiakkaiden arvomaailmaan. (Lecklin 2006, 85 - 87.)

Asiakasarvot voivat kuitenkin muuttua, koska ne ovat dynaamisia ja ne voivatkin muuttua eri tahtiin ja eri suuntiin. Muutos voi johtua mm. yhteiskunnallisista syistä, kilpailijoiden takia, innovaatioista ja sen myötä uusista teknologioista sekä asiakkaan muuttuneista käyttö- ja kulutustottumuksista. Yrityksen kilpailukykyä parannetaan nopeuttamalla prosesseja ja ennakoimalla asiakkaiden arvojen muutoksia. (Lecklin 2006, 90.)

Asiakkailla on odotusarvoja yrityksen tarjoamia tuotteita ja palveluita kohtaan. Odotusarvoihin vaikuttavia tekijöitä ovat mm. aiemmat kokemukset, asiakkaiden tarpeet, imago ja kilpailevien toimijoiden tuotteet, joihin organisaation tuotteita verrataan. Yrityksen toiminnan perusteella asiakkaalla on odotuksia tuotteen laadun lisäksi, myös ratkaisujen toimivuuteen, ammattimaisuuteen, asiantuntemukseen, uskottavuuteen ja osaamisen sekä yhteiskykyyn. Lisäksi asiakkaan vaatimustaso ja odotukset kasvavat asiakassuhteen vanhenemisen myötä. Jotta asiakastyytyväisyyteen voidaan vastata, yrityksen on pystyttävä täyttämään tai jopa ylittämään omat annetut lupaukset ja asiakkaan odotukset. Mielikuvien avulla yritys voi muokata asiakkaan odotusarvoja, tällöin voidaan valita joko korkea tai matala laatumielikuva. Korkean laatumielikuvan käytössä voi käydä niin, että jos toiminta ei vastaa sille asetettuja odotuksia ja lupauksia, niin asiakastyytyväisyys laskee. Matalaa mielikuvaa käyttämällä asiakkaiden odotukset ovat helpommin täytettävissä, mutta kiinnostus yritystä kohtaan on vähäinen. Parasta olisikin yhdistää mielikuvat, jotta yritys saa houkuteltua asiakkaita ja vastaamaan asiakasodotuksiin niihin toiminnalla, joka täyttää asiakkaan matalat odotukset ja jopa ylittää ne. (Lecklin 2006, 91 - 92.)

Markkinoinnissa ja myyntitilanteissa tehdyt toimet ovat tärkeitä asiakkaille ja ne vaikuttavat heidän laadunarviointiinsa (Grönroos 2000, 63). Asiakkaalle muodostuu odotusarvo yrityksen palvelun laadusta, johon vaikuttavat asiakkaiden tarpeiden lisäksi yrityksen markkinaviestintä ja muu tarjottu informaatio. Asiakkaan kokema koottu laatu syntyy, kun tekninen laatu ja toiminnallinen laatu, eli se kuinka hyvin palvelu toteutetaan ja mitä palvelu sisältää, yhdistetään. Asiakas vertaa kokemaansa laatua omiin odotuksiinsa, ja vaikka palvelu olisikin kelvollisesti toteutettu, niin siitä voi kaikesta huolimatta syntyä eroa odotetun ja koetun laadun välille. Näin voi käydä silloin kun, palvelua tai tuotetta on mainostettu ylisanoin ja luotu siten asiakkaalle korkea odotusarvo, eikä asiakkaan odotukset täyty. (Lecklin 2006, 94.)

Tiedonhallinta

Asiakassuhteiden hallintaan liittyy asiakastietojen hallinta, jotta yritys tuntee asiakkaansa. Tämän vuoksi yrityksen kannattaakin käyttää asiakassuhteiden hallintaan asiakastietojärjestelmää. Tietosisältönä voi olla asiakkaan osoite- ja toimitustietojen lisäksi mm.:

- asiakasprofiili ja -segmenttitiedot,
- tietoja toimintaympäristöstä,
- liiketoiminnan tunnuslukuja,
- ostojen kehitys,
- asiakastyytyväisyys,
- kontaktihistoria ja -suunnitelmaa.

Tietosisältöä ei kuitenkaan kannata tehdä liian laajaksi, vaan siinä tulee keskittyä asiakassuhteen kannalta olennaisiin asioihin. Hiljaisen tiedon kerääminen on tärkeää ja sen muuttaminen näkyvämpään muotoon kaikkien käyttöön on tärkeää. Hiljainen tieto on kokemusperäistä ja henkilökohtaista, mutta sitä voidaan jakaa henkilöiden välillä kollektiivisesti ja se on abstraktia. Hiljainen tieto vaikuttaa yksilön toimintaan. Asiakassuhteiden hallinnassa hiljaista tietoa kertyy mm. tilaus-, toimitus- ja yhteyskäytännöistä. Näissä tapauksissa asiakkaiden suosimat käytännöt ovat yhteyshenkilöiden tiedossa, ja he pyrkivät toimimaan niiden mukaan. Yrityksessä sovittujen dokumentointikäytäntöjen avulla tiedot ovat helposti dokumentoitavissa ja muiden käytettävissä. Toinen tärkeä hiljaisen tiedon lähde on asiakassuhteen historian ja tulevaisuuden yleiset suuntaviivat. (Lecklin 2006, 99; Oksanen 2010, 150.)

Yhteydenpito asiakkaisiin on tärkeää, eikä sen tulisi rajoittua pelkkään myyntitapahtumaan. Asiakkaalle nimetyt yhteyshenkilöiden tulisi huolehti asiakkaan hyvinvoinnista ja asiakassuhteen kehittymisestä koko asiakassuhteen ajan. Yhteyshenkilöitä voi olla useampiakin riippuen toimialasta tai kaupankäynninlaajuudesta. Tällöin yhteyshenkilöitä voivat olla lisäksi erilaiset asiantuntijat ja huollosta vastaavat henkilöt. Tällä tavalla asiakassuhdetta saadaan syvennettyä ja ne helpottavat yhteydenpitoa. (Lecklin 2006, 101.)

Asiakaspalvelu ja asiakastyytyväisyys

Asiakaspalvelun tehtävänä on kohdata asiakas ja sen merkitys on riippuvainen toimialasta. Palvelualoilla sen merkitys on suurin, mutta se on tärkeää myös teknisiä laitteita myyvien yritysten kohdalla. Ohjelmistoalan yrityksissä, jotka tekevät ohjelmistojen kehittämistä yhdessä asiakkaan kanssa, joutuvat vastaamaan asiakkaan odotuksiin jatkuvasti. Ohjelmiston käyttöönoton ja ylläpidon aikana tämän merkitys korostuu lisää. Asiakaspalvelun kehittämiseen olisi panostettava heti alusta alkaen, ottaen huomioon sen myös henkilöstön valinnassa ja koulutuksessa. Tavoitteena on, että asiakas saa otettua yhteyttä mahdollisimman helposti ja sujuvasti, kun asiakas tarvitsee apua tai neuvoja, palvelun tai tuotteen käyttämiseksi. (Lecklin 2006, 93, 101.)

Palvelukuvausten avulla koko palveluprosessi puretaan osiin, joista tehdään kuvaukset. Tämän avulla palveluprosessi voidaan tuotteistaa ja vakioida. Työvaiheiden tehtävät kuvataan ja niille määritetään standarditasot ja tavoitteet. On kuitenkin huomioitava, että kaikkea palveluita ei voi standardoida, jolloin kuvaukset toimivat ohjeellisena kehyksenä, ja asiakaspalvelussa toimivien henkilöiden on sitten pystyttävä joustamaan tilanteen mukaan. (Lecklin 2006, 101.)

Yhteistyön avulla yrityksen tuotteet ja palvelut saadaan vastaamaan paremmin asiakkaiden tarpeita ja sitä paremmin ne vastaavat asiakkaan tarpeisiin mitä tiiviimpää yhteistyö on. Tuotekehityksessä asiakkaan ottaminen mukaan jo aikaisessa vaiheessa säästää kustannuksia. Tuotekehitystä tekevä henkilöstö ei osaa useinkaan asetettua asiakkaan asemaan. Lisäksi asiakkaan tekemä koekäyttö tai beta-testaus mahdollistaa virheiden löytymisen ennen laajempaa julkistamista ja näin ollen estää yritystä mahdolliselta imagon menetykseltä. (Lecklin 2006, 102 - 103.)

Asiakassuhteiden hallintaan kuuluu myös asiakasvalitusten käsittely. Niiden käsittely on hoidettava hyvin, koska huonosti käsiteltynä ne heikentävät yrityksen mainetta. Tyytymätön asiakas kertoo saamastaan huonosta palvelusta tai tuotteesta monelle. Tyytyväinen asiakas sen sijaan on tyytyväinen omalla tahollaan. Asiakasvalitusten käsittely on oma prosessinsa, jossa valitukset ja syyt kohdistetaan toimintaprosessiin. Sen jälkeen analysoidaan, mitä on tehtävä. Valitukseen on vastattava henkilökohtaisesti mahdollisimman pian, jotta asiakassuhde säilytetään. Yrityksessä on huomioitava, että valittava asiakas ei ole vielä katkaissut asiakassuhdetta, joten asiallinen ja nopea valituksen käsittely voivat ylittää asiakkaan odotukset. Sen myötä asiakas voi olla tyytyväinen saamansa kohteluun ja saa hänet jatkamaan asiakassuhdetta. (Lecklin 2006, 103 - 104.)

Asiakastyytyväisyys on tärkeää, koska toiminta voi jatkua vain, jos asiakkaat ovat valmiita maksamaan yrityksen tuotteista tai palveluista käyvän hinnan (Lecklin 2006, 105). Asiakkailla voi olla odotuksia siitä, että palveluntarjoaja ratkaisee jonkin ongelman, mutta asiakkailla ei ole käsitystä siitä mitä pitäisi tehdä. Jos palveluntarjoaja havaitse sumeita odotuksia, eikä saa niitä esille ja näin ollen ratkaistua ongelmaa, niin silloin asiakas pettyy palveluun, ja se voi johtaa asiakastyytymättömyyteen. Samalla tavalla myös epärealistiset ja asiakkaiden implisiittiset odotukset voivat johtaa asiakastyytymättömyyteen. Asiakastyytyväisyyden selvittämisen myötä voidaan kehittää toimintaa. Tämän vuoksi asiakkaan näkemystä ja tyytyväisyyttä on selvitettävä eri toimintojen osalta. Tätä tieto voidaan saada normaalista asiakaspalautteesta ja lisätietoa saadaan tekemällä tyytyväisyystutkimuksia. Mittaustekniikat asiakastyytyväisyydelle ovat joko kvantitatiivisia tai kvalitatiivisia. Kvantitatiivisissa tutkimuksissa asiakkaita pyydetään arvostelemaan tyytyväisyyttään antamalla toiminnasta numero. Kvalitatiivisissa tutkimuksissa menetelminä ovat asiakaspaneelit, joissa voidaan haastatella tai käydä ryhmäkeskusteluja. (Grönroos 2009, 133 - 134; Lecklin 2006, 106 - 107.)

Asiakastyytyväisyyttä olisi seurattava jatkuvasti. Aluksi on kuitenkin tehtävä laaja-alainen kertatutkimus, jossa selvitetään mihin asiakkaat ovat tyytyväisiä ja mitkä ovat heidän mielestään ongelma-alueita. Tämän jälkeen jatkuvaan seurantaan voidaan valita mitattavat asiat ja kohteet. Mittaukset voivat olla erilaisia, mutta niiden olisi tuettava toisiaan. Asiakastyytyväisyyden lisäksi on selvitettävä myös asiakastyytymättömyys. Asiakkaat, jotka ovat tyytymättömiä, ovat organisaation kannalta riskiryhmä. Tämän vuoksi tyytymättömät asiakkaat olisi tunnistettava ja sen myötä jatkotutkimuksen avulla tunnistettava tyytymättömyyden syyt ja sitten tehtävä korjaavat toimenpiteet. Asiakasuskollisuus ei ole automaattista, vaikka asiakkaat olisivat tyytyväisiä tuotteisiin tai palveluun. Asiakkaiden mielestä liian korkea hinta tai hidas tuotekehitys voi johtaa siihen, että asiakkaat vaihtavat hankintalähdettä. Uskolliset asiakkaat eivät kuitenkaan aina ole kannattavia, varsinkaan silloin kun prosessit ovat yhtä raskaita, niin suurille kuin pienillekin asiakkaille. Tällöin pienet asiakkaat voivat käydä tappiollisiksi. Pitkät asiakassuhteet ovat kuitenkin yleisesti ottaen kannattavampia, koska tällöin asiakkaan ja myyjäorganisaation toimintatavat ovat hioutuneet paremmin yhteen. (Lecklin 2006, 112 - 115.) Mäntynevan (2000, 23) mukaan asiakasuskollisuus ja sen myötä pidemmät asiakkuuksien kestot saavutetaan panostammalla asiakastyytyväisyyteen sekä toimiin, joilla asiakkuuksia säilytetään (Mäntyneva 2000, 23). Asiakkaan säilyttämisen toimissa on tunnettava asiakkaat ja heidän tarpeensa syvällisesti. Tämän perusteella voidaan mallintaa ne asiakkaat, jotka ovat lopettamassa asiakkuuksiaan. Sen jälkeen voidaan tehdä potentiaalianalyysi, kannattaa kyseisiä asiakkuuksia säilyttää ”hinnalla millä hyvänsä”. (Mäntyneva 2000, 22.)

Asiakaspalveluhenkilöstö on tärkeässä roolissa yrityksen toiminnan kannalta, koska he ovat tekemisissä asiakkaan kanssa. Heidän toiminnastaan syntyy asiakkaalle mielikuva yrityksestä ja sen toiminnasta. Tuotteiden ja palvelujen tuntemus on oltava riittävää, jotta asiakaspalvelua voi tehdä menestyksekkäästi. Näiden lisäksi asiakaspalveluhenkilöstön on omaksuttava yrityksen arvomaailma ja omattava kommunikointikykyjä. Usein asiakaspalvelutilanteissa asiakkaat voivat antaa negatiivista palautetta, jolloin asiakaspalvelijan on kuitenkin hallittava hermonsa ja yritettävä ratkaista ongelma sekä sen myötä kääntää tilanne positiiviseksi. Tämän kautta asiakas kokee saaneensa hyvää palvelua ja jatkaa tyytyväisenä asiakassuhdetta. (Lecklin 2006, 118.)

3 Tutkimusstrategia

Tässä luvussa käydään läpi tämän tutkimuksen tutkimusstrategiaa. Ensin käydään läpi teoriaa, jota on käsitelty ns. menetelmäkirjallisuudessa. Tämän jälkeen perustellaan tutkimukseen tehdyt valinnat. Tutkimusasetelma pitää sisällään tutkimusongelman ja siitä johdetut tutkimuskysymykset. Siinä käydään läpi myös työn tavoite, miten tutkitaan ja aineisto kerätään sekä analysoidaan. Tutkimusasetelmassa otetaan myös kantaa luotettavuuteen ja itse kohteeseen (Kananen 2019, 21).

Tutkimusongelman määrittely on tärkeää, jotta tutkimus voi onnistua. Sen määrittäminen on kriittistä myös siksi, koska se ohjaa koko tutkimusprosessia. Tutkimuskysymyksellä voidaan etsiä syitä, mutta se ei kuitenkaan paljasta koko tutkimusongelmaa. Tutkimusongelman löytämiseksikin voi joutua tekemään tutkimusta ja sen löytäminen voi olla haasteellista. Tutkimusongelma on esitettävä mahdollisimman selkeästi ja tarkasti rajaten, sen tulisi esittää johtoajatus, johon tutkimuksen pääongelma voidaan täsmentää. Ongelman muotoileminen voi osoittautua kuitenkin joskus vaikeammaksi kuin sen ratkaiseminen. (Kananen 2015a, 41; Kananen 2019, 21; Hirsjärvi, Remes & Sajajärvi 2009, 125 - 126.)

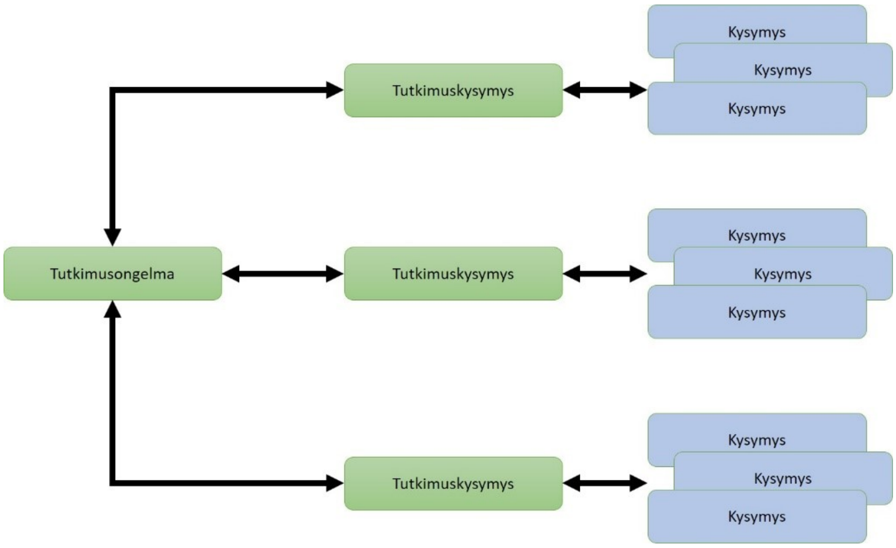
3.1 Tutkimuskysymys

Tutkimusongelman määrittämisen jälkeen siitä voidaan johtaa tutkimuskysymykset (Kananen 2019, 23). Tutkimusongelman ratkaisu löytyy vastauksista, joita saadaan, kun tutkimuskysymykset esitetään. Haasteena on kuitenkin muuttaa ongelma ymmärrettäväksi kieleksi. (Kananen 2015a, 55.) Tämän helpottamiseksi ensin on muotoiltava peruskysymys ”mitä?”. Sen jälkeen voidaan esittää apukysymyksiä, kuten:

- Miten jokin ilmiö toimii?
- Miksi ilmiö tapahtuu?
- Milloin ilmiö tapahtuu?

(Kananen 2019, 22.)

Tutkimuskysymyksiä ei voi suoraan esittää, sillä niiden lisäksi tarvitaan tarkennettuja kysymyksiä, joilla tuotetaan aineisto, josta saadaan vastaus tutkimusongelmaan, kuten kuvassa 9 on kuvattu. Kvalitatiivisessa tutkimuksessa kysymyksiä ei esitetä tutkittavalle, vaan ilmiötä tutkitaan haastattelemalla tutkittavia niiden teemojen kautta, jotka liittyvät aiheeseen (Kananen 2015a, 57 - 59).



Kuva 9. Tutkimusongelma purkautuu joukoksi tutkimuskysymyksiä, jotka taas muodostavat apukysymyksiä (Kananen 2015a, 58).

3.2 Tutkimusote

Tutkimusongelmien ratkaisemiseksi käytetään menetelmien kokonaisuuksia, joita voidaan kutsua tutkimusotteiksi. Otteen valinta on tärkeä päätös, koska valinta on perusteltava sekä sen lisäksi sovittava tutkittavaan aiheeseen että siitä johdettuun tutkimusongelmaan. Tutkimusotteet voidaan jakaa kahteen eri ryhmään, joita ovat kvantitatiivinen eli määrällinen tutkimus ja kvalitatiivinen eli laadullinen tutkimus (Kananen 2015a, 63; Hirsjärvi ym. 2009, 135 - 137).

Kvantitatiivinen tutkimus on lähtöisin luonnontieteistä ja siinä korostetaan yleisiä syyn ja seurauksen lakeja. Lähtökohtana tälle voidaan pitää ns. loogista positivismia, jossa korostetaan sitä, että tiedot ovat suorista aistihavainnoista johdettuja loogisia päätelmiä. Kvantitatiivisen tutkimuksen keskeisiä piirteitä ovat mm. johtopäätökset edellisistä tutkimuksista ja aiemmat teoriat. Siinä esitetään tutkimukselle hypoteesit ja määritellään tutkimuksen käsitteitä. (Hirsjärvi ym. 2009, 139 - 140.) Kvantitatiivinen tutkimus edellyttääkin ilmiön tuntemista, jotta voidaan kysyä oikeita kysymyksiä tutkimusongelman selvittämiseksi (Kananen 2019, 25).

Kvalitatiivisen tutkimuksen lähtökohtana on kuvata todellista elämää, näitä asioita ei voi mitata yksinkertaisella tavalla määrällisesti, kuten esimerkiksi kysyttäessä: Onko olut hyvää? Yhtenä piirteenä kvalitatiivisessa tutkimuksessa on se seikka, että todellisuus on vaihteleva, eikä sitä voida hajottaa osiin. Tutkimuksen kohdetta tutkitaankin tällöin kokonaisvaltaisesti. Kvalitatiivisen tutkimuksen pyrkimys on siis löytää tosiasioita enemmin kuin pyrkiä todistamaan aiemmin löydettyjä väittämiä. (Hirsjärvi ym. 2009, 161.)

Tapaustutkimus on tutkimusstrategia liiketaloustieteissä. Se soveltuu hyvin kehittämistyön lähestymistavaksi erityisesti silloin, kun tarkoituksena muodostaa kehittämisehdotuksia tai kehittämisideoita. Itse kohde voi olla yritys tai pelkästään joku sen osa, palvelu, toiminta tai prosessi. Tutkimuksen kohteen on oltava ilmiö, joka tapahtuu nykyajassa ja sitä ei voi tehdä aiemmin tapahtuneesta ilmiöstä. Teoreettinen viitekehys pohjautuu kuitenkin aiemmin tapahtuneisiin ilmiöihin ja niistä tehtyihin dokumentteihin. (Ojasalo, Moilanen & Ritalahti 2014, 52; Kananen 2013, 54.) Tapaustutkimusta voidaan kutsua myös case-tutkimukseksi.

Tapaustutkimuksessa tarkoituksena on tuottaa syvällistä ja yksityiskohtaista tietoa tutkimuksen kohteesta. Sen myötä tutkimuksessa ymmärretään ilmiötä sen oikeassa toimintaympäristössä, jolloin voidaankin kysyä, miten jokin on mahdollista tai kuinka jokin tapahtuu. Tällöin tutkimus pyrkiikin vastamaan kysymyksiin miten ja miksi. Tapausta tutkittaessa on otettava huomioon luonnollisen ympäristön ajalliset ja sosiaaliset muuttujat. (Ojasalo ym. 2014, 52 - 53.)

Tapaustutkimuksessa tutkimus aloitetaan usein tutkittavasta tapauksesta, ei vain yleisestä teoriasta. Tapauksen tutkijalla on usein tutkimuksen kohteena olevasta ilmiöstä jonkinlaista tietoa, joka mahdollistaa alustavan määrittelyn tutkimukselle. Lisäksi voi käydä myös niin, että aiheeseen perehtymisen jälkeen selviää vasta, mikä on todellinen kohde. Tästä seuraa se, että voidaan kehittää täsmentäviä kysymyksiä, joiden avulla löydetään tutkimuksen tausta-aineistoa. Tämän myötä voi käydä niinkin, että prosessin edetessä kohdekin voi muuttua tietojen tarkentuessa. (Ojasalo ym. 2014, 54.)

Tapaustutkimus luo syvällisen kuvan ilmiöstä keräämällä aineiston erilaisista tietolähteistä, joita voivat olla esimerkiksi dokumentit, havainnointi ja haastattelut. Tietolähteiden monipuolisuus ja hajanaisuus tekevät tutkimuksesta haastavan hallita. Useiden erilaisten tiedonkeruumenetelmien takia tutkija joutuu hallitsemaan useita erilaisia menetelmiä. Lisäksi analysointivaiheessa on otettava huomioon kunkin tiedonkeruumenetelmän mukaiset analysointimenetelmät. (Kananen 2013, 77 - 79.)

Kehittämistutkimuksessa pyritään muutokseen. Se voi olla kvantitatiivisen ja kvalitatiivisen tutkimuksen yhdistelmä. Se voi olla myös pelkästään kvalitatiivinen tutkimus, jonka tavoitteena on muutos. Yleensä siinä kehitetään tuotetta, menetelmää tai organisaatiota muutoksen aikaansaamiseksi. Tavallisesta kehittämistyöstä se eroaa siinä, että kehittämistutkimus vaati tutkimuksellista otetta. (Kananen 2015a, 76.) Kehittämistutkimuksessa käytännön ongelmat ja kysymykset määrittävät tiedontuotantoa, jolloin tietoa tuotetaan aidoissa toimintaympäristöissä käyttäen tutkimuksellisia asetelmia ja menetelmiä (Toikko & Rantanen 2009, 22).

Kehittämistutkimuksessa tutkija ei osallistu muutoksen läpivientiin toisin kuin toimintatutkimuksessa. Kehittämis- ja toimintatutkimuksen välillä ei nähdä usein eroja ja sen englannin kielisiä vastineita ei oikein ole. Kanasen mukaan kehittämistutkimus onkin suomalaisten itsensä kehittämä versio toimintatutkimuksesta. (Kananen 2014, 29 - 30.) Se on kehitetty 1980- ja 1990-luvuilla muutosstrategiaksi, jossa yhdistetään tutkimus, käytännön kehitystyö ja koulutus (Toikko ym. 2009, 31). Kehittämistutkimus eroaa perinteisestä laadullisesta ja määrällisestä tutkimuksesta siinä, että siinä pyritään poistamaan ongelma. Perinteisessä tutkimuksessa ongelmaa ei poisteta, vaan sitä analysoidaan ja siihen esitetään ratkaisu. (Kananen 2015b, 40.)

Tutkimuksellisessa kehittämisessä on ymmärrettävä kuinka toimijat hahmottavat työnsä, sen tavoitteet ja sitä ohjaavat periaatteet. Näitä tietoja ei välttämättä ymmärretä tutkimalla organisaation virallisia organisaatiokaavioita. Tutkimuksellinen kehittämistyö on uuden tiedontuotannon mukaista toimintaa, jossa tieto syntyy käytännönyhteyksistä. Tiedon on oltava käyttökelpoista. Siinä on otettava myös kantaa, kenen etua kehittäminen palvelee. Kehittäminen ei perustu ennalta tiukasti määritettyihin tavoitteisiin ja prosesseihin, vaan se etenee prosessimaisesti. Kehittäminen vaatii tällöin toimintatapojen tarkastamista koko ajan, ja se vaatii reflektiivistä asiantuntijuutta. (Toikko ym. 2009, 54 - 55.)

Kehittämistutkimus on aloitettava nykytilan kartoittamisella, jossa määritetään poistettava ongelma. Ongelman määrittely ja siihen liittyvät tekijät on analysoitava ja määritettävä huolellisesti, koska tässä vaiheessa vaikutetaan siihen, että löydetään oikea ongelma. Sen määrittämiseen voi joutua käyttämään runsaasti aikaa. Aikaa kuluu myös vaihtoehtoisten ratkaisumallien tuottamiseen. Muutoksen onnistuminen riippuu siis siitä, kuinka ongelma pystytään määrittämään ja miten se voidaan muuttaa tutkimuskysymyksiksi. Tutkimuskysymykset tuottavat tietoa ongelman ratkaisuun. Kun ongelman syyt on löydetty, niin sen jälkeen on arvioitava keinot, joiden avulla ongelma on poistettavissa. Itse ongelman poistaminen on oma prosessinsa. Joskus voi käydä, niin että vaikka ongelman syyt tiedetään, niin ongelmalle ei voida tehdä mitään. (Kananen 2015b, 41 - 42.)

Toimintatutkimuksessa pyritään ratkaisemaan yhdessä käytännön ongelmia ja saamaan samalla aikaan muutosta. Tutkimuksen kohteena olevat ongelmat voivat olla niin teknisiä, sosiaalisia, eettisiä kuin ammatillisia. Tavoitteena onkin ratkaista käytännön ongelma. Tutkimus tuottaa samalla myös uutta tietoa ilmiöstä. Lähestymistavaltaan siinä ollaan kiinnostuneita siitä, miten asioiden pitäisi olla eikä pelkästään siitä, miten ne ovat. Siinä tavoitteena on nykyisen tilan muuttaminen. Toimintatutkimus yhdistääkin usein käytännön ja teorian. Toimintatutkimus edellyttää useita vaiheita. Näiden vaiheiden välillä käytäntö ja reflektio vaihtelevat ja myös suunnittelu ja toteutus vaihtelevat. (Ojasalo ym. 2014, 58; Toikko ym. 2009, 30.)

Toimintatutkimus on sekoitus erilaisia tutkimusmenetelmiä eikä se sulje pois mitään tiedonkeruu- ja analyysimenetelmiä, olivat ne sitten kvalitatiivisia tai kvantitatiivisia (Kananen 2015a, 28). Itse tutkimusprosessi etenee vaiheittain suunnittelun, havaintojen ja arvioinnin ympyränä. Vaiheet on suhteutettava toisiinsa järjestelmällisesti ja kriittisesti. Tutkimuksen alussa selvitetään kirjallisuudesta ja muusta lähteistä onko vastaavaa tutkittu aiemmin. Sen jälkeen itse kehittämistehtävää voidaan tarkentaa. (Ojasalo ym. 2014, 60 - 61.)

3.3 Aineiston kerääminen

Yleensä kvalitatiivisen tutkimuksen aineistonkeruumenetelminä käytetään haastatteluja, kyselyitä, havainnointia ja dokumenteista saatava tietoa. Erilaisia aineistonkeruumenetelmiä voidaan käyttää rinnakkain tai erilaisina yhdistelminä riippuen tutkimusongelmasta ja millaiset resurssit ovat käytettävissä. Edellä mainittuja aineistonkeruumenetelmiä voidaan käyttää myös kvantitatiivisessa tutkimuksessa. (Tuomi & Sarajärvi 2013, 71.)

Haastattelujen tavoitteena on selvittää mitä mieltä tutkittava on ilmiöstä. Haastattelu on keskustelua, jota käydään tutkijan aloitteesta ja johdattelemana. Yleensä haastattelu on ennalta suunniteltua ja siinä on motivoitava haastateltavaa sekä pidettävä haastattelua yllä. (Eskola & Suoranta 1998, 85.) Haastattelut on hyvä yhdistää muihin aineistonkeruumenetelmiin, jolloin ne tukevat toisiaan (Ojasalo ym. 2014, 106).

Haastattelutyypit voidaan jaotella sen mukaan kuinka muodollisia ja jäsenneltyjä eli strukturoituja ne ovat. Ääripäinä tällöin on täysin strukturoitu haastattelu, jossa ennalta laaditut kysymykset esitetään aina tietyssä järjestyksessä ja toinen ääripää on strukturoimaton haastattelu, jossa aiheesta keskustellaan vapaasti. (Hirsjärvi ym. 2009, 208.)

Strukturoitu keskustelu on ns. lomakehaastattelu, jossa kysymysten muoto ja esittämisjärjestys on määrätty ennakolta (Hirsjärvi ym. 2009, 208). Tällöin kysymyksillä on sama merkitys kaikille haastateltaville. Strukturoitu haastattelu vastaa kyselylomakkeen täyttämistä ohjatusti. Jos strukturoidussa haastattelussa ei ole valmiita vastausvaihtoehtoja, vaan haastateltava vastaa omin sanoin, niin silloin kyseessä on puolistrukturoitu haastattelu. (Eskola ym. 1998, 86.)

Teemahaastattelu on haastattelumuoto, joka on strukturoidun ja strukturoimattoman haastattelun välissä. Siinä haastattelija on määrittänyt haastattelussa läpikäytävät aihepiirit, mutta itse kysymykset ja niiden järjestys puuttuu. (Hirsjärvi ym. 2009, 208.) Teemahaastattelu on mahdollista toteuttaa niin yksilö- kuin ryhmähaastatteluna (Kananen 2015a, 148).

Yksilöhaastattelua voidaan käyttää silloin, kun tutkimuksen aiheena on jokin henkilökohtainen tai arkaluonteinen aihe. Ryhmähaastatteluun osallistuu samalla kertaa useita henkilöitä, jolloin haastatteluaikaa tarvitaan vähemmän. Ryhmähaastattelussa on haasteena tilanteet, joissa on mukana esimies-alaissuhteessa olevia henkilöitä, koska tällöin kaikki eivät välttämättä tuo näkemystään julki esimiehen läsnä ollessa. (Kananen 2015a, 148 - 149.) Haastattelujen avulla saadaan tietoja siitä, mitä henkilöt aiheesta tai ilmiöstä ajattelevat eli sen, miten henkilöt havaitsevat ilmiön (Hirsjärvi ym. 2009, 212).

Havainnointia käytetään täydentämään haastatteluja tai niitä voidaan käyttää itsenäisesti. Sen avulla päästään tapahtumien luonnollisiin ympäristöihin. (Ojasalo ym. 2014, 114.) Havainnointi ei sovi menneeseen eikä tulevaan aikaan, vaan ilmiön on tapahduttava reaaliajassa (Kananen 2015a, 136). Osallistuvassa havainnoinnissa havainnoija on paikan päällä ja osallistuu toimintaan eri asteisesti. Osallistuvan havainnoinnin etuna pidetään, sitä että tällöin päästään syvälle itse tutkittavan ilmiön olemukseen, vaikka tutkija ei olisikaan varsinainen yhteisön jäsen. (Kananen 2015a, 137.) Osallistava havainnointi eroa osallistuvasta havainnoinnista siinä, että tutkimukseen osallistuvat henkilöt osallistutetaan siihen, jolloin hyvin alkanut toiminta jatkuu myös tutkimuksen loputtua (Tuomi ym. 2013, 82).

3.4 Aineiston käsittely

Analyysin tarkoituksena on selkeyttää kerättyä aineistoa ja tuottaa tällä tavalla uutta tietoa tutkimuksen kohteesta. Kvalitatiivisen tutkimuksen haastavin osa voikin olla aineistolle tehtävä analyysi. Se kuinka analyysi tehdään ei määräydy minkään säännön mukaan, ja joskus ei ole selvillä sekään, milloin ja kuka analysoinnin tekee. Kvalitatiivisessa tutkimuksessa aineistoa kerätään usein eri menetelmin ja eri vaiheissa, tällöin analyysia tehdään myös aineiston keräämisvaiheessa. (Eskola ym. 1998, 137; Hirsjärvi ym. 2009, 224.)

Kerätty tutkimusaineisto on voi olla useassa eri muodossa, koska se voi sisältää haastatteluaineistoja, havainnointiaineistoja sekä erilaisia dokumentteja. Aineistojen erilaisuus aiheuttaa sen, että niiden käsitteleminen yhdellä kertaa ei ole mahdollista. Jokainen tutkimuksessa eri tavoin kerätty aineisto vaatii sen, että tutkija käsittelee kaikki aineistot erikseen niiden vaatimilla tavoilla. (Kananen 2015a, 160.)

Aineistot on litteroitava, jolla tarkoitetaan erilaisten audiovisuaalisten dokumenttien muuttamista tekstimuotoon. Tutkijan on valittava millä tarkkuudella litterointi tehdään, koska se on hidas työvaihe. Sanantarkassa litteroinnissa kirjataan jokainen äännähdys ja joskus jopa eleetkin. Yleiskielisessä litteroinnissa teksti muutetaan kirjakielelle, jolloin siitä poistetaan murre- ja puhekielen ilmaisut. Propositiotason litteroinnissa kirjataan ainoastaan ydinsisältö. (Kananen 2015a, 160 - 161.)

Yhteismitallistamisen avulla erilaisia aineistoja voidaan käsitellä samalla analyysimenetelmällä. Dokumenteista voidaan ottaa mukaan vain tarvittava osa. Haastattelut on purettava tekstimuotoon, jotta niiden käsittely on mahdollista tietokoneohjelmilla. Aina yhteismitallistamista ei ole tarpeen tehdä, varsinkaan silloin kun käsiteltävä aineisto on suppea, ja se on hallittavissa lukemalla. (Kananen 2015a, 161.)

3.5 Aineiston analysointi ja toiminnan kehittäminen

Aineiston analysointi vaatii sen lukemista useampaan kertaan, jotta siitä saa hahmotettua aineiston sisällön. Aineiston jatkojalostamista ei ole aina tarpeen tehdä, vaan pelkkä lukeminen ja sen pohdiskelu voi riittää. Jos aineistoa on paljon, voidaan käyttää aineiston tiivistämistä asiasisällöksi. Asiasisällön löytämisen jälkeen sitä kuvataan sisältöä kuvaavalla sanalla. (Kananen 2015a, 163.)

Aineiston tulkinta voi olla käytetyn päättelylogiikan mukaan joko induktiivista, jossa edetään yksittäisestä yleiseen tai deduktiivista, joka on induktiivisen vastakohta. Näiden yhdistelmää kutsutaan abduktiiviseksi päättelyksi. Aineistolähtöisessä tulkinnassa aineistoa analysoidaan ilman teoreettista etukäteistietoa tai olettamusta. Tässäkin tapauksessa tutkijalla on tietoa ja olettamuksia tutkittavasta ilmiöstä, mutta se ei saa vaikuttaa aineistosta nouseviin teemoihin. Teorialähtöisessä tulkinnassa hyödynnetään jotakin teoreettista näkökulmaa ja aineistoa pidetään esityksenä valitusta teoriasta. Tällöin sitä myös tarkastellaan sen näkökulman kautta. (Tuomi ym. 2013, 95; Eskola ym. 1998, 151 - 152.)

Kehittämisen on oltava luonteeltaan kehittävää toimintaa ja siihen on liityttävä toiminnan arviointia. Kehittämisen kautta pyritään luomaan ja levittämään uusia tuotteita ja palveluita. (Toikko ym. 2009, 56 - 57.)

Prosessianalyysin avulla tutkitaan ja kehitetään prosesseja. Sen avulla voidaan selvittää mitä kukakin toimija prosessissa tekee, ja mitkä ovat sen kriittisiä vaiheita. Samalla voidaan myös tunnistaa missä ja miksi ongelmia ilmenee. Prosessin eri vaiheita havainnollistetaan luomalla prosessikaavio, josta käy ilmi ongelmat sekä niihin tehdyt ratkaisuvaihtoehdot. (Ojasalo ym. 2014, 178.)

3.6 Tutkimukseen valitut menetelmät

Tutkimusongelman muotoilu on kehittynyt menetelmäkirjallisuutta tutkittaessa ja sen vaikutuksesta. Tutkimusongelma on muotoutunut muotoon ”Millainen Datala Oy:n ohjelmiston ylläpitoprosessi on ja miten ylläpitoprosessia voidaan kehittää”. Tutkimusongelmasta on johdettu tutkimuskysymys: ”Miten ohjelmiston ylläpitoa tehdään?”. Tutkimusongelman selvittämiseksi tarvitaan lisäksi joukko muita kysymyksiä, joiden avulla tarkennetaan tutkimusongelman ratkaisua. Tällaisia kysymyksiä tutkimusongelman selvittämiseksi ovat seuraavat kysymykset:

- Mikä laukaisee ylläpitotoiminnon käsittelyn, ts. mikä on signaali toiminnan aloittamiselle?
- Miten ylläpitäjä määräytyy?
- Millainen ohjelmiston ylläpidon prosessi on?
- Miten prosessi voi kehittyä?

Tutkittava ilmiö on tuttu tutkijalle ja työyhteisölle, mutta ilmiössä esiintyviä vaiheita ei ole tutkittu tutkimuskohteessa. Tutkimus on kvalitatiivinen ja siinä tutkitaan ilmiötä ja tunnistetaan sen vaiheet. Aineistonhankinta on toteutettu teemahaastatteluin ja havainnoin. Haastattelut toteutettiin teemahaastatteluina, koska henkilöstön määrä yrityksessä on pieni. Havaintojen avulla todennetaan, onko ilmiö koetun kaltainen. Ilmiön tuntemisen myötä jatkotutkimus voi olla tutkimusotteeltaan kvantitatiivinen. Jatkotutkimuksessa voitaisiin tällöin mitata eri vaiheiden läpimenoaikoja.

Tutkimus painottuu teorialähtöiseen tulkintaan, tyypillisesti laadullisessa tutkimuksessa käytetään kuitenkin induktiivista analyysia. Pohjan teorialähtöiseen tulkintaan tuo tutkimukseen liittyvän teorian käsittely. Tutkimuksen ja tulosten analysoinnin edetessä voi olla mahdollista, että tulkinta malli voi muuttua myös teoria- ja aineistolähtöisen tulkinnan välimuodoksi.

Tutkimusstrategialtaan tutkimus on tapaus- ja kehittämistutkimuksen mukainen. Tapaustutkimus tämä on sen vuoksi, että tässä keskitytään yhteen ja tiettyyn ilmiöön, jota pyritään kuvamaan aineiston avulla. Kehittämistutkimukseksi tämän tutkimuksen tekee se, että työssä kuvataan kehitysehdotus toimeksiantajalle. Kehittämisvaiheessa tutkimuksen aineiston pohjalta laadittiin prosessikuvaus tapahtumasta. Kyseinen prosessikuvaus on toimeksiantajan toiminnan nykykuvaus. Tämän jälkeen siitä laadittiin analyysien perusteella kehitysehdotus. Kehitysehdotuksessa prosessin toimintaan lisättiin uusi toimija ja vaiheistusta muutettiin. Vaikka asiakassuhteiden hallintaan kuulukin olennaisena osana asiakastyytyväisyyden mittaaminen, niin tässä työssä sitä ei tehty.

4 Ohjelmiston ylläpitoprosessin kuvaaminen ja kehittäminen

Tässä luvussa kuvataan tarkemmin tutkimuksen ja kehittämistoiminnan kohdetta. Lisäksi tässä luvussa esitetään tutkimuksen tekeminen ja sen pohjalta tehty kehittämisehdotus.

Toimeksiantaja on Datala Oy, joka on kajaanilainen ohjelmistoalan yritys. Yritys on perustettu 1983. Yrityksen toimialueena on koko Suomi. Asiakkaat koostuvat pääosin elintarvikealan yrityksistä, erityisesti meijereistä, joille Datala Oy on toimittanut ohjelmistoja vuosikymmeniä. Ohjelmistot muodostavat laajoja kokonaisuuksia. Ohjelmistojen toiminta käsittää maidon keräilyyn liittyvää hallinnointia, laboratoriotulosten käsittelyä, maitotilien käsittelyä ja tuottajille tapahtuvaa myyntiä. Lisäksi toimintaan voidaan liittää meijerien tuottajille tarjoamia erilaisia lisäpalveluita. Tällaisia lisäpalveluita ovat tuottaja-extranet palvelut ja koetulosten tekstiviestipalvelut. Datala Oy tarjoaa asiakkailleen ohjelmistoja myös palveluna, jolloin Datala Oy hallinnoi palvelimia ja ohjelmistoja. Tällaisessa palvelussa asiakkaan itsensä ei tarvitse huolehtia kuin työasemasta ja toimivasta internet-yhteydestä. Osa asiakkaista pitää itse järjestelmiään yllä, jolloin Datala Oy:n vastuu rajoittuu ohjelmiston ylläpitoon.

Työssään Datala Oy:n työntekijät joutuvat usein tekemään ohjelmiston ylläpitoa, joka tarkoittaa käytännössä sitä, että asiakkaan kohdatessa ongelmia asiakkaat ottavat yhteyttä Datalaan ja kertovat kohtaamastaan ongelmasta. Tämän jälkeen työntekijät pyrkivät ratkaisemaan asiakkaan kohtaaman ongelman mahdollisimman joutuisasti, jotta asiakkaan oma työ voisi jatkua viiveettömästi. Tapaukset, jotka ovat Datala Oy:n hallinnoimilla palvelimilla ovat ”helpompia” kuin ne tapaukset, joissa joudutaan toimimaan asiakkailta saadun tiedon varassa eikä pääsyä asiakkaan järjestelmään ole.

Tutkimus koostui nykytilan selvittämisestä, josta muodostui opinnäytetyön tutkimuksellinen osa. Tutkimus oli kvalitatiivinen tapaustutkimus, joka toteutettiin teemahaastatteluin ja havainnoin. Kehittämistyö muodostui nykytilan prosessikuvauksesta. Nykytilan prosessikuvauksen jälkeen tehtiin prosessianalyysi, jonka perusteella luotiin kehitysehdotus toimeksiantajalle. Kehitysehdotus on kuvattu myös prosessikuvauksena. Kehittämistyö liittyi prosessien ja niiden kehittämisen viitekehykseen.

4.1 Tutkimuksen toteutus

Tutkimus oli kvalitatiivinen tapaustutkimus ja se toteutettiin toimeksiantajan tiloissa, jolloin tutkimus tapahtui ilmiön luonnollisessa ympäristössä. Teoriaa tapaustutkimuksesta on käsitelty alaluvussa 3.2 Tutkimusote. Tutkimus keskittyi ns. korjaavaan ohjelmiston ylläpitoon, koska tämän kaltaista ylläpitoa ei voi ennakoida. Korjaava ylläpito toiminto on luonteeltaan yllättävää ja vaatii usein nopeaa reagointia. Siinä on usein myös ohjelman toimintaan liittyvää korjausta, muutosta tai ohjelman toiminnan selvittämistä. Ohjelmiston ylläpitoon liittyviä muotoja ja teoriaa on käsitelty tarkemmin kappaleessa Ylläpito ja ohjelmat. Tutkimussuunnitelmaa laadittaessa prosessikuvauksesta ei tehty hypoteesia, jolloin ilmiötä tarkasteltiin mahdollisimman objektiivisesti tutkimuksen aikana. Tältä osin tutkimustyö noudattaa aineistolähtöistä analyysiä, mutta tutkimuksessa on sovellettu tutkimuksen edetessä myös teorialähtöistä analyysiä. Analyysien eroja on kuvattu tarkemmin alaluvussa 3.5 Aineiston analysointi ja toiminnan kehittäminen.

Toimeksiantajan tiloissa tehtiin yksilö- ja ryhmämuotoisia teemahaastatteluja sekä tutkimukseen liittyvää havainnointia. Tutkimuksen primääriaineisto koostui siten teemahaastatteluista, jotka toteutettiin kahtena haastatteluna ja havainnoinnista. Haastatteluihin osallistui Datala Oy:n koko henkilöstö, eli toimitusjohtaja ja kaksi työntekijää. Paikaksi se valikoitu myös siksi, että se on neutraali tila, ”ei kenenkään maa”. Tällä järjestelyllä pyrittiin välttämään esimies-alaissuhteessa olevien henkilöiden välinen hierarkkinen suhde, josta on mainittu alaluvussa 3.3 Aineiston kerääminen. Ensimmäinen haastattelukierros toteutettiin yksilöhaastatteluna, johon osallistui kaksi työntekijää, joista tutkija oli toinen. Toisen haastattelukierroksen kysymyksiä muutettiin ensimmäisestä haastattelukierroksesta saatujen kokemusten perusteella. Haastattelujen paikkana oli Datala Oy:n neuvottelutila, joka on rauhallinen ja häiriötön ympäristö. Toiselle haastattelukierrokselle osallistui toimitusjohtaja ja kaksi työntekijää, joista toinen oli tutkija. Haastattelut toteutettiin tammikuun 2021 aikana, niin että haastattelukierrosten välillä oli neljä päivää. Haastatteluaineistoa kertyi tallenteina yhteensä noin 50 minuuttia.

Haastattelun teemat olivat ohjelmiston ylläpito ja asiakassuhteiden hallinta. Kyseiset teemat valikoituivat aiheiksi, siksi että tutkimuksen ne on koettu tärkeiksi kehitystyön kannalta. Haastatteluissa teemojen alle tutkija muodosti kysymyksiä, joihin haastatteluissa haettiin vastauksia. Kysymykset muotoutuivat substanssiteoriassa esitettyjen asioiden pohjalta. Kysymyksiä ei esitetty suoraan, vaan asioista keskusteltiin yhdessä. Haastatteluissa käsitellyt teemat ja niihin liittyvät kysymykset on esitetty erillisessä liitteessä yksi. Haastattelut nauhoitetiin analysointia varten. Seuraavaksi nauhoitukset purettiin yleiskieliselle tasolle. Analysointivaiheessa keskustelut ensin segmentointiin. Sen jälkeen segmentistä haettiin avainsana, jotta prosessin erilaiset vaiheet saatiin tunnistettua. Kolmannessa vaiheessa segmentistä ja avainsanasta muodostettiin kuvaus prosessi kuvausta varten.

Haastattelun myötä ilmiön olemus ja kuinka haastateltavat kokevat ilmiön saatiin selville. Haastatteluissa todettiin esimerkiksi se, että ohjelmien ominaisuuksien ja koon kasvaminen ovat tehneet ohjelmista monimutkaisempia ja siten haastavampia ylläpitää:

” Onhan ne monimutkaistuneet ja niissä on enemmän ominaisuuksia, jos jotain vanhaa käyttöliittymää katsoo, niin siellä on käyttäjän pitänyt tarkkaan tietää, että mitä nappia painaa missäkin kohtaa. Nykyään ohjelmat ovat sellaisia, että ne ohjaavat käyttäjää tekemään oikein ja varoittaa virheistä. Että kyllä tämä lisää ohjelmien monimutkaisuutta. ”

Lisäksi haastatteluissa koettiin tärkeänä asiakkaan osallistuminen vaatimusmäärittelyn luomiseen, jotta ohjelmat toimisivat alusta alkaen tarkoituksen mukaisesti. Tämän katsottiin pienentävän niin ohjelmistotoimittajan kuin asiakkaankin kustannuksia, kuten haastattelusta poimitussa keskustelun katkelmassa on todettu:

*”**H1:** Mutta joka tapauksessa tämän voi summata, että yhteydenpito asiakkaisiin on tärkeää ja se että asiakkaat saisi osallistettua määrittelyyn, jotta ne muutokset ja korjaukset saataisiin menemään kerralla oikein.**H2**: Kyllä **H1**: Ja näin ollen asiakkaalle koituvat kustannukset voisivat pienentyä myös, koska työ tulisi kerralla valmiiksi. **H2**: Ja varmasti meidän kustannuksemme myös.”*

Seuraavaksi tutkimuksessa suoritettiin havainnointia. Havainnointi oli luonteeltaan osallistuvaa ja siinä tutkija toimi yhtenä toimijana, havainnointia tutkimusmenetelmänä on käsitelty tarkemmin alaluvussa 3.3 Aineiston kerääminen. Havainnot kerättiin päiväkirjatyyppisesti, niin että siitä kirjattiin päivämäärä, kellonaika ja tapahtuman lyhyt kuvaus. Havaintopäiväkirjaa on havainnollistettu liitteessä kaksi olevalla esimerkki kuvauksella. Havaintoaineistoa kerättiin yhden kuukauden ajalta. Tapauksia kertyi tutkijalle 21 kappaletta. Kuvausten kirjaamista vaikeutti se, että havaintojen tekeminen oli tehtävä varsinaisen tapahtuman jälkeen, jotta varsinaisen työn tekeminen ei häiriinny ja hidastu havaintojen kirjaamisen takia.

Havaintojen avulla varmistettiin ilmiön olemus. Haastattelujen analysointi ja havainnointi tapahtui limittäin. Haastattelujen analysoinnin myötä tutkija havaitsi yhtäläisyyksiä havaintojen analysoinnin kanssa. Havainnointi toisti itseään, jolloin saturaatio saavutettiin ja sen varsinainen käsittely voitiin tehdä. Kirjausvaiheessa tiedot olivat tapauskohtaisessa järjestyksessä. Jokainen tapaus muodosti yhden kokonaisuuden, vaikka se jakautuikin useammalle päivälle. Aineisto järjestettiin tämän jälkeen tapauksen aloittavan päivämäärän mukaiseen järjestykseen nousevasti eli alkaen vanhimmasta tapauksessa päättyen uusimpaan.

Sen jälkeen aineistosta haettiin avainsanat tapahtumille samalla tavalla kuin haastatteluja analysoitaessa. Havannoista ei laadittu tiivistelmää tapahtumasta, koska itse tapahtuman kuvaus oli sinällään jo tiivis kuvaus siitä. Haastattelujen ja havaintojen luokittelujen avulla tunnistettiin joukko toimintoja, jotka muodostavat ilmiön.

Aineiston perusteella voitiin löytää ilmiössä esiintyvät erilaiset toimijat, kuten esimerkiksi asiakas. Aineistosta etsittiin toimijalle kuuluvia toimintoja lukemalla aineistoa useita kertoja läpi. Lisäksi aineiston litterointi vaihe itsessään auttoi hahmottamaan aineistoa. Toiminnot järjestettiin lopuksi ajalliseen järjestykseen aina kunkin toimijan kohdalla. Vaiheiden ja toimijoiden vähäisen määrän vuoksi työ voitiin tehdä tekstinkäsittelyohjelman taulukkojen avulla. Suuremmalla aineistomäärällä toimiessa pelkkien taulukkojen varassa toimiminen ei olisi välttämättä enää toiminut. Ilmiöstä voitiin laatia sen jälkeen kuvaus, joka esitettiin prosessina.

Prosessin sanallisen kuvauksen tekeminen auttoi hahmottamaan ilmiötä kokonaisuutena ja muodostamaan siitä kokonaiskuvan. Prosessin sanallinen kuvaus ja toimintotaulukko tukevat toisiaan, koska toimintotaulukon avulla voidaan luoda mielikuvakartta tapahtumista ja toimijoista. Toimintotaulukon laatiminen tehtiinkin yhtä aikaa sanallisen kuvauksen kanssa. Prosessikaavio laadittiin vasta sen jälkeen, kun prosessin sanallinen kuvaus ja toimintotaulukko oli valmis. Prosessikaaviossa käytettävä malli oli tuttu YAMK-opintojen yhteydessä suoritetulta opintojaksolta ja sen myötä luonteva valinta prosessin kuvaukseen.

4.2 Nykytilan kuvaus ja kehittäminen

Ilmiö muodostuu joukosta toimintoja, jota voidaan nimittää prosessiksi. Prosessin kuvausta varten laadittiin sanallinen kuvaus, jossa käydään läpi prosessin kulku. Siinä esitellään myös prosessin roolit ja tavoite sekä taulukko, jossa kuvataan prosessin toiminnot.

Prosessista laadittiin sen jälkeen kuvaus nykytilanteesta. Prosessi kuvattiin työn kulku tasolla, koska siinä kuvataan toiminnan työvaiheita ja se näyttää yhden työn vaiheet. Prosessin kuvaaminen tehtiin ns. uimaratamallilla, jossa jokaisella roolilla on oma uimaratansa. Siinä uimaradalle kuvatut prosessiaskeleet kuuluvat ko. uimaradan roolin vastuulle. Uimaratakuvaus auttaa ymmärtämään prosessin toimintaa ja mikä tehtävä on milläkin roolilla. Uimaratakuvaus ja siinä käytettävät merkinnät on kuvattu tarkemmin kappaleessa Prosessin kuvaaminen.

Prosessikaavion jälkeen kehittämistyössä vaiheena oli prosessin kehittäminen, jossa käytettiin ns. kolmevaiheista mallia. Kolmivaiheinen prosessin kehittämismalli on esitelty työhön liittyvässä teoriassa kappaleessa Prosessien kehittämismalleja. Kun mallissa esitetyn vaiheistuksen mukaisesti nykytilan kartoitus oli tehty ja kuvattu prosessina, niin voitiin analysoida itse prosessia. Analyysin perusteella tehtiin kehitysehdotus prosessin tai sen vaiheiden parantamiseksi.

Kehittämistyössä pyrittiin huomioimaan asiakkaiden suuntaan tapahtuvan tiedottamisen, yhteydenottotapojen sekä kommunikaatioväylien parantaminen. Kehitysehdotus pitää sisällään ehdotuksen uudesta teknisestä järjestelmästä. Prosessin edelleen kehittäminen vaatisi mittariston käyttöönottoa. Mittariston avulla toiminnot saataisiin optimoitua ja samalla tuotettua lisäarvoa asiakkaille, kun ohjelmistot ovat nopeasti uudelleen käytettävissä. Tämän kehittämistyön puitteissa mittariston valintaan ja käyttöönottoon ei kuitenkaan voida ottaa kantaa.

Nykytilan kuvaus

Tässä kuvataan prosessin perustiedot ohjelmiston ylläpidossa nykytilan mukaisesti, prosessista kuvataan prosessin omistaja, sille asetettu tavoite ja siinä olevat roolit.

Omistaja: Esimies (Datala Oy)

Tavoite: Asiakkaille tarjotaan nopeaa palvelua ja ongelmien ratkaisua.

Roolit: Asiakas. Asiakkaalla kuvataan tässä asiakasyrityksen edustajaa. Ilmoittaa ongelmasta.

Esimies. Henkilö, joka ottaa useimmiten vastaan asiakkaan ilmoituksen ongelmasta. Jakaa tehtävät työntekijöille.

Työntekijä. Etsii ja korjaa ongelmat, dokumentoi korjaukset.

Prosessi kuvataan sanallisesti seuraavasti:

Asiakas ottaa yhteyttä Datalaan joko sähköpostilla tai puhelimella ja ilmoittaa ongelmastaan. Esimies vahvistaa samaansa viestin asiakkaalle ja jakaa tehtävän työntekijöille. Esimies voi käsitellä ongelmaa myös itse. Työntekijän saatua tehtävän hän kohdentaa ongelman tiettyyn järjestelmään ja ohjelmaan sekä itse ongelman. Kohdentaminen voi tapahtua testaamalla ja toistamalla tapahtumaa samalla tavoin tehtynä, kuin asiakas on sitä tehnyt. Jos kohdentaminen ei onnistu asiakkaalta saaduilla tiedoilla, ottaa työntekijä tällöin yhteyttä asiakkaaseen ja pyytää lisäselvitystä ongelmaan.

Asiakas toimittaa lisäselvityksen, jonka avulla työntekijä voi kohdentaa ongelmaa. Vian paikallistamisen jälkeen työntekijä voi korjata ohjelman. Korjaamisen yhteydessä tai välittömästi sen jälkeen työntekijä dokumentoi korjaamisen ja mahdolliset muutokset lähdekoodiin kommentoimalla. Korjaamisen jälkeen testataan korjausta, jotta se toimisi oikein. Testaaminen voi johtaa palaamisen korjaamisvaiheeseen, jos korjaus ei tuota haluttua lopputulosta, tätä vaihetta toistetaan niin kauan, että korjauksesta saadaan haluttu lopputulos.

Ongelman kohdentamisen aikana voi selvitä myös se, että asiakas ei ole käyttänyt ohjelmaa oikein. Tällöin työntekijä tutkii ohjeita ja vertaa vastaako se ohjelman toimintaan. Jos ohjeet vastaavat ohjelman toimintaan, niin sitten asiakkaalle annetaan opastusta ohjelman toiminnasta. Ohjeiden ollessa ristiriidassa toiminnan kanssa, työntekijä päivittää ohjeistusta. Päivitetty ohjeistus toimitetaan asiakkaalle ja annetaan samalla opastusta.

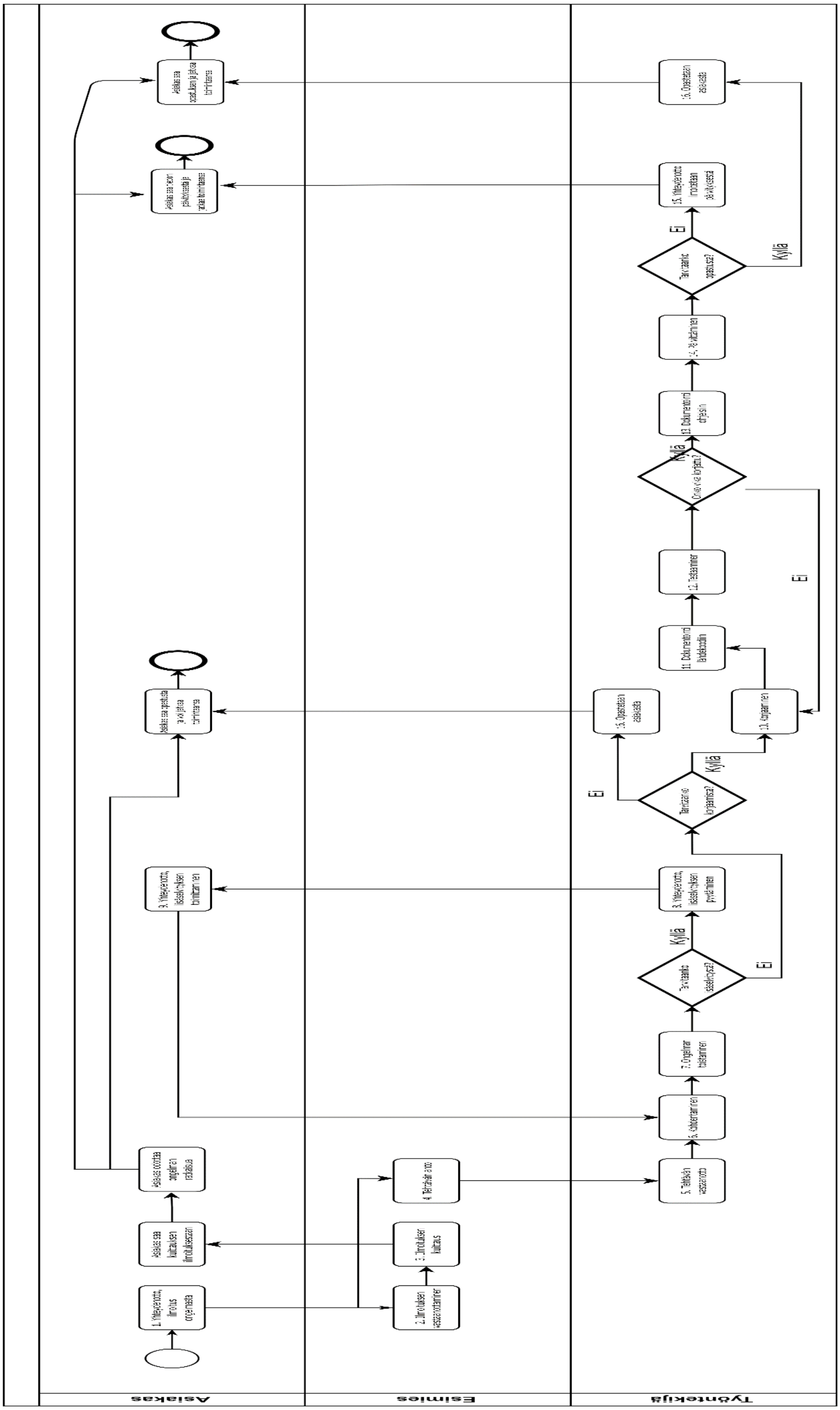
Korjattu ohjelma päivitetään ja asiakkaalle ilmoitetaan päivitysajankohta tai sovitaan siitä asiakkaan kanssa, jos se vaatii ns. käyttökatkon, jolloin ohjelma ei ole asiakkaan käytettävissä. Päivittämisen jälkeen asiakkaaseen otetaan yhteyttä ja kerrotaan päivityksen asentamisesta. Lopuksi asiakkaalle voidaan antaa mahdollisesti opastusta.

Prosessiin kuuluvat vaiheet on kuvattu toimintotaulukkoon, joka on esitelty taulukossa 4. Taulukosta nähdään tarkempi kuvaus toiminnolle ja määritetään rooli toiminnolle. Prosessi kuvataan ns. uimaratamallilla, jossa jokaisella roolille on oma ratansa. Radalle kuvattu toiminto kuuluu silloin tälle roolille.

Toiminto	Toimija	Kuvaus
1. Yhteydenotto	Asiakas	Asiakas ottaa yhteyttä Datalaan joko puhelimella tai sähköpostilla ja kertoo ongelmastaan.
2. Ilmoituksen vastaanottaminen	Esimies	Esimies ottaa vastaan asiakkaan ilmoituksen.
3. Ilmoituksen kuittaus	Esimies	Esimies kuittaa asiakkaalle.
4. Tehtävän anto	Esimies	Esimies jakaa tehtävä.
5. Tehtävän vastaanotto	Työntekijä	Työntekijä ottaa vastaan tehtävän.
6. Kohdentaminen	Työntekijä	Työntekijä kohdentaa asiakkaan antaman ilmoituksen perusteella vian tai ongelman tiettyyn järjestelmään ja ohjelmaan.
7. Ongelman toistaminen	Työntekijä	Työntekijä pyrkii toistamaan ongelman.
8. Yhteydenotto, lisäselvityksen pyytäminen	Työntekijä	Työntekijä ottaa yhteyttä asiakkaaseen ja pyytää lisäselvitystä.
9. Yhteydenotto, lisäselvityksen pyytäminen	Asiakas	Asiakas toimittaa lisäselvityksen.
10. Korjaaminen	Työntekijä	Korjaa paikallistetun vian.
11. Dokumentointi	Työntekijä	Dokumentoidaan korjattu vika lähdekoodiin.
12. Testaaminen	Työntekijä	Testataan korjattua vikaa
13. Dokumentointi	Työntekijä	Dokumentoidaan / päivitetään ohjelmaan liittyvää ohjeistusta.
14. Päivittäminen	Työntekijä	Ohjelma tai ohjelmisto päivitetään.
15. Yhteydenotto	Työntekijä	Asiakkaalle ilmoitetaan päivityksestä.
16. Opastus	Työntekijä	Asiakasta opastetaan ohjelman käytössä.

Taulukko 4. Prosessin toimintotaulukko.

Prosessi on kuvattu kuvaan 10 uimaratakaaviona, jossa on erotettu rooleille omat radat.



Kuva 10. Prosessin kuvaus uimaratakaaviona.

Prosessin kehittäminen

Tässä kuvataan prosessin perustiedot ohjelmiston ylläpidossa kehitysehdotuksen mukaisesti. Prosessista kuvataan prosessin omistaja, sille asetettu tavoite ja siinä olevat roolit.

Omistaja: Esimies (Datala Oy)

Tavoite: Asiakkaille tarjotaan nopeaa palvelua ja ongelmien ratkaisua.

Roolit: Asiakas. Asiakkaalla kuvataan tässä asiakasyrityksen edustajaa. Ilmoittaa ongelmasta.

Järjestelmä. Ottaa vastaan asiakkaan tekemät vikailmoitukset ja tekee niistä saapumisilmoitukset. Toimii myös tietovarastona ongelmatapauksista. Järjestelmää kutsutaan tässä kuvauksessa myöhemmin Tiketti-järjestelmäksi.

Esimies. Henkilö, joka ottaa useimmiten vastaan asiakkaan ilmoituksen ongelmasta. Jakaa tehtävät työntekijöille.

Työntekijä. Etsii ja korjaa ongelmat, dokumentoi korjaukset järjestelmään.

Kehitetty prosessi kuvataan sanallisesti seuraavasti:

Prosessia on kehitetty lisäämällä siihen uusi rata, johon on lisätty järjestelmä. Kuvauksessa sitä kutsutaan Tiketti-järjestelmäksi. Järjestelmän tarkoituksena on automatisoida toimintoja ja tarjota nopeampaa ongelman ratkaisua asiakkaalle sekä kerätä tietoa Datala Oy:n käyttöön. Järjestelmä palvelee myös Datala Oy:tä sillä tavalla, että se tarjoaa tietoa laskutusprosessia varten, joka on Datala Oy:n tukiprosessi. Tässä yhteydessä on otettu huomioon Lean-menetelmän perusajatus, johon sisältyy se, ettei tehdä mitään mikä ei tuota asiakkaalle lisäarvoa. Tiketti-järjestelmän käyttö edellyttää niin asiakkailta kuin muiltakin sen käyttäjiltä määrämuotoista dokumentaatiota, joka voidaan toteuttaa lomakemuotoisena. Järjestelmä voi olla teknisesti toteutettu extranet-tyyppisenä järjestelmänä, johon asiakas kirjautuu. Tiketti-järjestelmän toimintaa voidaan kuvata siten, että se toimii ikään kuin Kanban-kortisto.

Kirjautumisen jälkeen asiakas valitsee järjestelmän, jota asia koskee. Tämän jälkeen asiakas täyttää muutoksenpyyntö-lomakkeen, johon kirjataan vika tai muutos. Asiakas voi liittää lomakkeeseen kuvia tai muita dokumentteja kuten esim. Word-dokumentin, johon asiakas on voinut kerätä muuta tietoa. Tärkeätä on, että asiakas täyttää lomakkeen ohjatusti ja täyttää sen siten, että se tarjoaa tietoa kerralla mahdollisimman paljon.

Järjestelmä lähettää asiakkaalle kuittaus viestin tapauksen vastaanotosta ja generoi ilmoituksen esimiehelle tai suuremmalle vastaanottajajoukolle. Esimies saadessa tiedon hän jakaa tehtävän työntekijöille. Työntekijän saadessa tehtävän hän kirjaa järjestelmään itsenä ko. tapauksen käsittelijäksi ja kirjaa lisää mahdollisia alkutietoja. Seuraavaksi työntekijä etsii järjestelmästä vastaavia tapauksia, jos niitä löytyy, niin sitten hän pyrkii niiden avulla testaamaan, onko asiakkaan ongelma ratkaistavissa niiden dokumentoinnin avulla. Jos järjestelmästä ei löydy vastaavia tapauksia, jatkaa työntekijä ongelman kohdentamista ja prosessia kuten aiemminkin. Ongelman ratkaisun myötä tiketin ratkaisuun johtaneet tiedot ja muut dokumentit linkitetään tikettiin. Lopuksi tiketti suljetaan.

Järjestelmään kerätään tietoa tapauksista. Tapauksia voidaan hakea järjestelmästä siihen tallennettujen metatietojen avulla. Järjestelmään linkitetään tieto ongelmaan liittyvästä lähdekoodista ja muusta dokumentaatiosta. Ajan myötä järjestelmään kerättyjen tietojen avulla ongelmien ratkaisu nopeus paranee ja dokumentaation taso määrämuotoisten lomakkeiden avulla yhtenäistyy.

Prosessin tavoite pysyy samana, mutta prosessin jatkokehittäminen vaatii mittariston käyttöönottoa. Prosessin mittareiden olisi arvioitava laatua, tehokkuutta ja joustavuutta. Lisäksi prosessista pitäisi laatia riskiarvio ja miettiä sen myötä vaihtoehtoiset menettelytavat, sille jos järjestelmässä on häiriö tai se ei toimi.

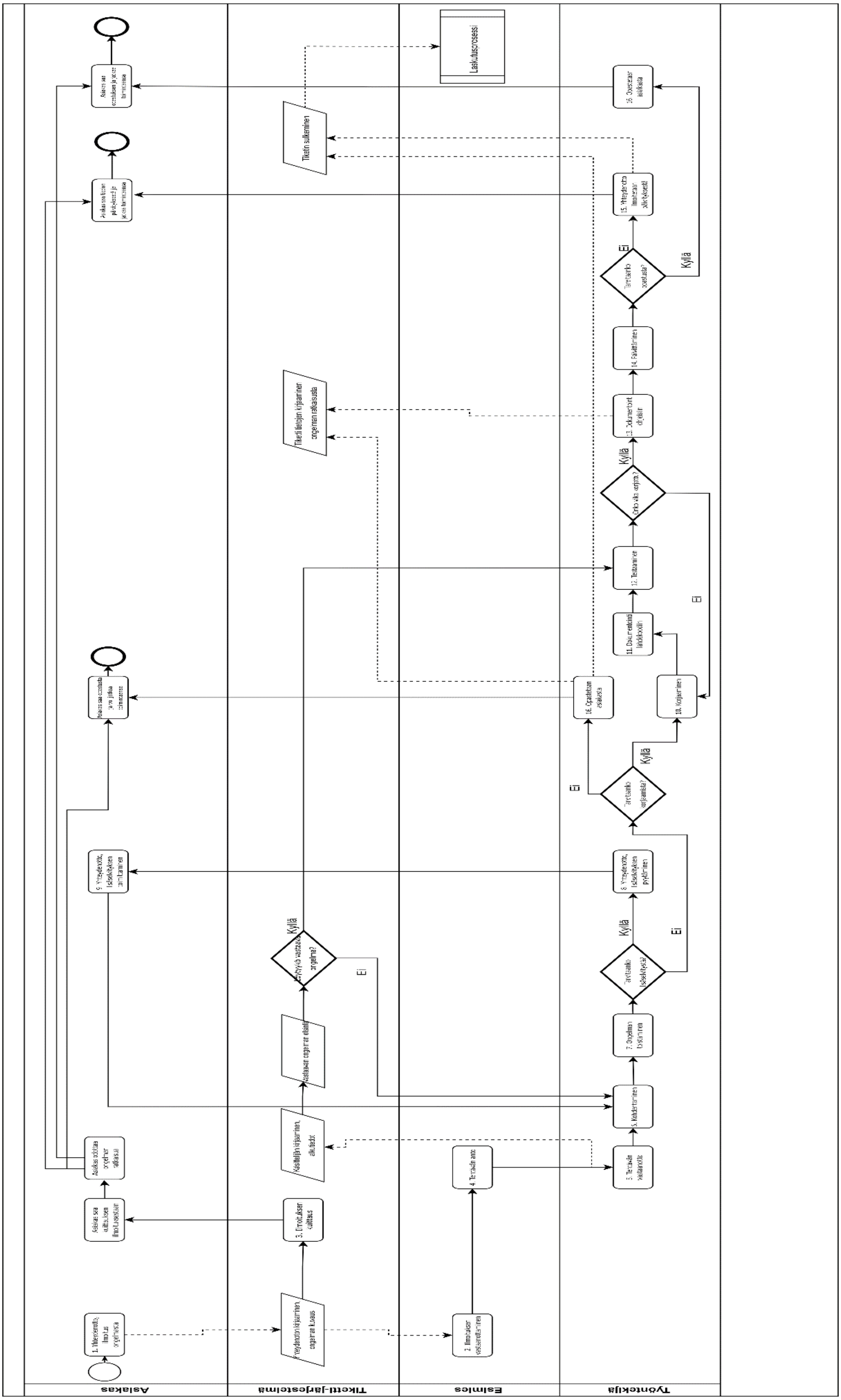
Asiakassuhteiden hallintaan liittyen yhteydenpito asiakkaisiin olisi tehtävä säännölliseksi varsinkin ohjelmistopäivitysten jälkeen, jotta mahdolliset virheet saataisiin selville nopeasti. Järjestelmää voinee kehittää myöhemmin myös niin, että se tekisi asiakkaille asiakastyytyväisyyskyselyn automaattisesti. Tässä yhteydessä voi mainita esimerkkinä autojen huoltojen jälkeen tulevat asiakastyytyväisyysmittaukset, jotka tulevat tekstiviestinä ja joihin asiakkaat voivat vastata nopeasti tekstiviestein.

Ohjelman tehtävän kommentoinnin lisäksi dokumentaation päivittäminen vastamaan ohjelmistoon tehtyjä muutoksia on tärkeää ja siihen tulisi kiinnittää huomioita. Dokumentointia kehitettäessä kannattaisi tehdä menettelyohjeet, jossa kerrotaan mitä dokumentteja päivitetään ja kuinka päivittäminen tehdään. Menettelyohjeiden laadinnassa apuna voisi käyttää esimerkiksi Pigoskin kuvailemia käytänteitä, joita on kuvailtu kappaleessa Ohjelmiston ylläpidettävyys. Käytänteistä sovittaessa kannattanee niiden laatimiseen käyttää koko henkilöstöä ja keskustella niistä samalla niitä kehittäen. Myöhemmin, kun menettelyohjeita on laadittu tarpeeksi, voidaan halutessa edetä kohti laatukäsikirjan laatimista. Näin toimittaessa dokumentointia voidaan yhtenäistää ja sen myötä voidaan varmistua tiedon paikkaansa pitävyydestä ja samalla kehittää toimeksiantajan toimintaa.

Taulukkoon 5 on kuvattu kehitetyn prosessin toimintotaulukko, johon Tiketti-järjestelmä on lisätty toimijaksi. Taulukossa on muotoiltu sinisellä muuttuneet kohdat ja Tiketti-järjestelmään liittyvät kohdat. Kuvaan 11 on kuvattu muuttunut prosessi.

Toiminto	Toimija	Kuvaus
1. Yhteydenotto	Asiakas	Asiakas ottaa yhteyttä Datalan tikettijärjestelmään ja liittää siihen tarvittavat tiedot
2. Ilmoituksen vastaanottaminen	Esimies	Esimies ottaa vastaan tikettijärjestelmän ilmoituksen.
3. Ilmoituksen kuittaus	Tiketti-järjestelmä	Tiketti-järjestelmä kuittaa asiakkaalle.
4. Tehtävän anto	Esimies	Esimies jakaa tehtävä.
5. Tehtävän vastaanotto	Työntekijä	Työntekijä ottaa vastaan tehtävän.
a. Tiketin täydentäminen	Työntekijä	Työntekijä täydentää tiketin tietoja.
b. Ongelman etsintä	Työntekijä	Työntekijä etsii järjestelmästä vastaavaa ongelmaa.
6. Kohdentaminen	Työntekijä	Työntekijä kohdentaa asiakkaan antaman ilmoituksen perusteella vian tai ongelman tiettyyn järjestelmään ja ohjelmaan.
7. Ongelman toistaminen	Työntekijä	Työntekijä pyrkii toistamaan ongelman.
8. Yhteydenotto, lisäselvityksen pyytäminen	Työntekijä	Työntekijä ottaa yhteyttä asiakkaaseen ja pyytää lisäselvitystä.
9. Yhteydenotto, lisäselvityksen pyytäminen	Asiakas	Asiakas toimittaa lisäselvityksen.
10. Korjaaminen	Työntekijä	Korjaa paikallistetun vian.
11. Dokumentointi	Työntekijä	Dokumentoidaan korjattu vika lähdekoodiin.
12. Testaaminen	Työntekijä	Testataan korjattua vikaa
13. Dokumentointi	Työntekijä	Dokumentoidaan / päivitetään ohjelmaan liittyvää ohjeistusta.
a. Tietojen kirjaaminen tiketille	Työntekijä	Työntekijä kirjaa tikettiin liittyvän ratkaisun ja liittää siihen liittyvät dokumentit.
14. Päivittäminen	Työntekijä	Ohjelma tai ohjelmisto päivitetään.
15. Yhteydenotto	Työntekijä	Asiakkaalle ilmoitetaan päivityksestä.
a. Tiketin sulkeminen	Työntekijä	Työntekijä sulkee avoimen tiketin, tapaus valmis.
16. Opastus	Työntekijä	Asiakasta opastetaan ohjelman käytössä.

Taulukko 5. Muuttunut prosessikuvaus, johon lisätty Tiketti-järjestelmä.



Kuva 11. Prosessikuvaus Tiketti-järjestelmän lisäämisen jälkeen.

5 Johtopäätökset ja pohdinta

Opinnäytetyö aloitettiin tutustumalla menetelmäkirjallisuuteen ja sen jälkeen aiheeseen liittyvään substanssikirjallisuuteen. Aiheeseen liittyvä toiminta oli tutkijalle työkokemuksen kautta tuttua, mutta aihekirjallisuutta tutkittaessa ja käsiteltäessä tutkijan näkemys aiheeseen laajeni lisäten uusia näkökulmia aiheeseen. Lisää uusia näkökulmia ja ajatuksia aiheeseen tarjoutui opinnäytetyöseminaarissa, jossa tutkijalle tarjoutui mahdollisuus opponoida opiskelijakollegan tekemää tutkimussuunnitelmaa. Kyseisessä työssä tutkittiin toimeksiantajalle ylläpidon prosessin mittaristoa ja sen kehittämistä.

Laadullisessa tutkimuksessa reliabiliteetti ja validiteetti voivat olla epäselvempiä kuin kvantitatiivisessa tutkimuksessa. Esimerkiksi tapaustutkimuksessa tutkija voi ajatella, että kuvaukset ihmisistä ovat ainutlaatuisia, tällöin perinteisesti käytetyt luotettavuuden ja pätevyyden arvioinnit eivät siihen sovi. Tällöin tarkka selostus tutkimuksen toteuttamisesta nostaa luotettavuuden tasoa. Tällöin kerrotaan olosuhteista ja paikoista, joissa aineisto on kerätty. Samalla tulisi kertoa myös aineiston keräämisen käytetty aika ja häiriötekijät. (Hirsjärvi ym. 2009, 232.) Tapaustutkimuksen luotettavuustarkastelun luonteenomaisin piirre on monilähteisyys. Lähteitä voivat olla kirjalliset dokumentit, havainnointi, tallenteet, kyselyt, teemahaastattelut ja muut tietolähteet. Mitä useammasta lähteestä saadaan vahvistusta esitetylle ilmiölle, sitä luotettavampi annettu kuvaus on. (Kananen 2013, 121 - 122). Aineiston keräämistä ja aineiston käsittelyä onkin pyritty kuvaamaan mahdollisimman hyvin, jotta lukijalla olisi mahdollisuus saada käsitys työssä käytetyistä menetelmistä ja arvioida sen luotettavuutta. Tutkimuksen tulosten luotettavuutta varmistettiin käyttämällä eri lähteitä, joita tässä tutkimuksessa ovat teemahaastattelut ja havainnointi.

Tutkijalla oli työkokemuksen ja substanssikirjallisuuteen tutustumisen myötä aavistus ilmiöstä, mutta tutkimusta tehdessä tutkija pyrki objektiiviseen näkemykseen, eikä hypoteesia ilmiöstä asetettu tutkimuksen alussa. Tutkimuksessa tunnistettiin ensin ilmiö tekemällä teemahaastatteluja ja havaintoja. Sen jälkeen pyrittiin ymmärtämään ilmiötä. Tutkimusmenetelmänä oli teemoittaminen ja tyypittely, joiden avulla tutkittiin mistä vaiheista Datala Oy:n ohjelmiston ylläpitoprosessi muodostuu.

Vaiheiden tunnistamisen myötä ylläpitoprosessista laadittiin sanallinen nykytilan kuvaus, josta tehtiin toimintotaulukko. Taulukossa kuvattiin roolit, joita prosessissa on ja toiminnot, joita prosessin aikana tapahtuu. Taulukon perusteella laadittiin prosessikuvaus. Kehitystyössä nykytilan mukaista prosessia analysoitiin ja pohdittiin, miten sitä voisi kehittää. Analysoinnin perusteella laadittiin sanallinen kuvaus kehitetystä prosessista ja sen myötä toimintotaulukko, kuten prosessin nykytilanteen kuvauksessa tehtiin. Taulukon perusteella laadittiin prosessikuvaus uudesta kehitetystä prosessista. Kehitettyyn prosessiin lisättiin uusi rooli, joka on tietojärjestelmä, jonka kautta asiointi toimii. Samalla tietojärjestelmä toimii myös tietovarastona ja laskutusprosessin osana.

Kehitystyössä kuvattu prosessi voi toimia toimeksiantajalle pohjana toiminnan kehittämiseen. Kehitystyössä kuvatun kaltaisen järjestelmän käyttöönotto edellyttää joko sellaisen hankkimista valmiina markkinoilta tai sitten kuvatun kaltaisen järjestelmän kehittämistä itse. Järjestelmän itse kehittäminen olisi luultavasti nopein tapa saada Datala Oy:n käyttöön soveltuva järjestelmä. Joka tapauksessa siihen olisi kyettävä laittamaan resursseja niin hankintaan kuin järjestelmän edelleen kehittämiseen. Järjestelmän tuottaman lisäarvon arviointi on hankalaa ja päätökset sen hankkimisesta on syytä tehdä huolella.

Haastattelujen ja havaintojen myötä yksi ohjelmistoelinkaaren vaihe nousi erityisesti esiin, joka oli vaatimusmäärittely. Vaatimusmäärittely on ohjelmiston elinkaareen alkuosan vaiheita. Se on ohjelmiston onnistumisen kannalta tärkeimpiä vaiheita, koska silloin usein asiakas ja ohjelmistotoimittaja käyvät läpi toiminnallisuutta ja asiakkaan asettamia vaatimuksia. Jonkun toiminallisuuden tai vaatimuksen puuttuminen tässä vaiheessa vaikuttaa myöhempiin vaiheisiin. Vesiputousmallissa vaatimusmäärittely on kuvattu toiseksi vaiheeksi heti esitutkimuksen jälkeen. Vaatimusmäärittelyyn joudutaan palaamaan aika-ajoin myös ohjelmiston ylläpitovaiheessa, kun tarkistetaan ohjelman toimintaa. Vaatimusmäärittely dokumentin päivittäminen vastamaan ohjelmaan tehtyjä muutoksia edesauttaisi myös sen ylläpitoa. Ylläpidon kehittämiseksi ja hallitsemiseksi kannattaisi pohtia myös Pigoskin kuvailemia käytänteitä, jolloin kehitys- ja ylläpitotyöt yhtenäistyvät.

Opinnäytetyön tekeminen on lisännyt tutkijan tietoisuutta teoriassa käsitellyistä aiheista, vaikka erityisesti teoriassa käsitelty ohjelmiston elinkaari on ollut tutkijalle tuttua. Kuitenkin ohjelmiston ylläpitoon liittyvät näkemykset ja teoriat ovat erityisesti herättäneet jatkokysymyksiä tutkimuksen edetessä. Asiakassuhteiden hallinta ja prosessien kehittäminen aiheina ovat olleet haasteellisia ja niiden teorian soveltaminen työhön on ollut haasteellista. Työn valmiiksi saattaminen tutkijan itsensä asettamassa aikataulussa onnistui. Työssä saavutettiin asetetut tavoitteet, eli saatiin kuvattua nykytila ja tehtyä sen pohjalta toimeksiantajalle kehitysehdotus. Opinnäytetyön myötä tutkija oppi uusia työkaluja ja menetelmiä ongelmien ratkaisemiseen, joten myös sen myötä työn voi sanoa olleen menestyksekäs.

Pohdintaa herätti myös ajoittainen keskustelu vesiputousmallin soveltuvuudesta nykyaikaiseen ohjelmistokehitykseen. Vesiputousmallin rinnalle ja korvaajaksi on kehitetty muita erilaisia vaihtoehtoisia kehitysmalleja. Useimmiten näistä käytetään nimitystä ”ketterät” menetelmät. Niiden etuina vesiputousmalliin nähdään nopea ohjelmistokehitys ja tulosten esittäminen asiakkaalle. Esimerkkinä tällaisesta kehitysmallista voidaan pitää Scrumia, jota on kuvattu kappaleessa Ohjelmiston elinkaarimallit. Datala Oy:n tapauksessa Scrumia ei voine käyttää täydellisesti, koska työt voivat jakautua päivän aikana useaan eri toimintaan. Tämä ei välttämättä edistä Scrumin tyyppistä työn etenemistä, jossa edetään koko ajan eteenpäin ja edellytetään sitoutumista työn eteenpäin saattamiseen. Vesiputousmallissa, sen sijaan voidaan tarvittaessa palata edelliseen vaiheeseen, tarkentaa sitä ja jatkaa työn eteenpäin viemistä. Työ on osoittanut tutkijalle, että kokonaisuuden hallinta on tärkeää, eikä keskittyminen pelkästään yhteen osa-alueeseen, kuten pohdintaan käytettävästä kehitysmallista ole järkevää. Toimintaa yrityksissä ja muissakin organisaatioissa olisi kehitettävä kaikki osa-alueet huomioiden.

Opinnäytetyön tulosten hyödyntämistä ei ollut mahdollista suorittaa tämän työn puitteissa, vaan kehitysehdotuksen sisältämien toimenpiteiden käyttöönotto, seuranta ja mahdollinen parantaminen ovat jatkotutkimuksen aiheita ja toimia. Yhtenä jatkotutkimus aiheena voisi olla tässä tutkimuksessa kuvatun järjestelmän käyttöönotto ja sen kehittäminen. Yhtenä kehittämispiirteenä voitaisiin tutkia mahdollisuutta asiakastyytyväisyyden tutkimiseen. Asiakastyytyväisyyden pohjalta voitaisiin kehittää Datala Oy:n toimintatapoja ja kohdentaa toimenpiteet niihin toimiin, jotka tuottaisivat asiakkaille mahdollisimman suuren lisäarvon.

Olipa jatkotutkimuksen aihe mikä tahansa, niin tavoitteena on kuitenkin pidettävä laadukkaan tuotteen toimittamista asiakkaalle oikeaan aikaan ja tarpeeseen. Samalla asiakkaalle tarjotaan nopeaa ja varmaa käytönaikaista tukea koko ohjelmiston elinkaaren ajan kasvattaen asiakkaan luottamusta ohjelmistotoimittajaan. Näin toimiessa asiakkuudet voivat jatkua pitkiäkin ajanjaksoja.

Lähteet

Abdul-Muhmin, A. (2012). *CRM technology use and implementation benefits in an emerging market. Journal of Database Marketing & Customer Strategy Management* (2012) 19, 82-97. Luettu 1.11.2020. https://link.springer.com/content/pdf/10.1057/dbm.2012.13.pdf

Adesola, B., Baines, T. (2005). *Developing and evaluating a methodology for business process improvement. Business Process Management Journal*, Vol. 11 Iss 1 pp. 37 – 46. Luettu 28.10.2020. https://www.researchgate.net/profile/Sola_Adesola/publication/235291580_Developing_and_evaluating_a_methodology_for_business_process_improvement/links/5c2e9314299bf12be3ab3311/Developing-and-evaluating-a-methodology-for-business-process-improvement.pdf

Agile Manifesto. (2020). *Julistuksen takana olevat periaatteet*. Luettu 17.11.2020. https://agilemanifesto.org/iso/fi/principles.html

Datala Oy. (2020). Datala Oy:n internet-sivut. Luettu 9.11.2020. https://www.datala.fi/datala/joomla3/index.php/datala-oy

Erälinna, L. (2013). *Liikesuhteen ulottuvuuksien väliset vaikutussuhteet maataloustarvikekaupassa*. Väitöskirja. Helsinki: Helsingin yliopisto. Luettu 1.11.2020. https://helda.helsinki.fi/bitstream/handle/10138/38048/liikesuh.pdf?sequence=1&isAllowed=y

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

Evans, D., Mckee, J. (2010). *Social Media Marketing. The Next generation of Business Engagement*. Hoboken: Wiley Publishing Inc.

Gerdt, B., Eskelinen, S. (2018). *Digiajan asiakaskokemus*. Helsinki: Alma Talent Oy. E-kirja. Luettu 1.11.2020. http://kamezproxy01.kamit.fi:2048/login?qurl=https%3A%2F%2Fbisneskirjasto.almatalent.fi%2Fteos%2FDAEBDXDTEB%23kohta%3ADigiajan%28%2820%29asiakaskokemus

Grubb, P., Takang, A. (2003). *Software maintenance: Concepts and Practice*. Second Edition. Singapore: World Scientific Publishing Co. Pte. Ltd.

Grönroos, C. (2009). *Palvelujen johtaminen ja markkinointi*. Helsinki: WSOYpro.

Grönroos, C. (2000). *Service management and marketing*. Chichester: John Wiley & Sons, Ltd.

Haikala, I., Mikkonen, T. (2011). *Ohjelmistotuotannon käytännöt*. Helsinki: Talentum.

Haikala, I., Märijärvi, J. (2004). *Ohjelmistotuotanto*. Helsinki: Talentum.

Harrington, H.J. (1991). *Business process improvement: The Breakthrough strategy for total Quality, Productivity and Competitiveness*. New York: McGraw-Hill. Luettu 21.2.2021. https://issuu.com/davinther/docs/business_process_improvement___the_

Harsu, M. (2003). *Ohjelmien ylläpito ja uudistaminen*. Helsinki: Talentum.

Hirsjärvi, S., Remes, P. & Sajavaara, P. (2009). *Tutki ja kirjoita*. Helsinki: Tammi.

Hunnbeck, L., Rudd, C., Lacy, S., Hanna, A. (2011). *ITIL service design*. Lontoo: TSO (The Stationary Office).

Kananen, J. (2019). *Opinnäytetyön ja pro gradun pikaopas*. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. (2015a). *Opinnäytetyön kirjoittajan opas*. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. (2015b). *Kehittämistutkimuksen kirjoittamisen käytännönopas*. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. (2013). *Case-tutkimus opinnäytetyönä*. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. (2014). *Toimintatutkimus kehittämistutkimuksen muotona. Miten kirjoitan toimin-tatutkimuksen opinnäytetyönä*. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kiiskinen, S., Linkoaho, A., Santala, R. (2002). *Prosessien johtaminen ja ulkoistaminen*. Helsinki: WSOY.

Koistinen, H. (2002). *Tietojärjestelmien ylläpito*. Helsinki: Talentum.

Koponen, T. (2007). *Evaluation of Maintenance Processes in Open Source Software Projects Through Defect and Version Management Systems*. Väitöskirja. Kuopio: Kuopion Yliopisto. Luettu 20.10.2020. https://epublications.uef.fi/pub/urn_isbn_978-951-27-0107-0/urn_isbn_978-951-27-0107-0.pdf

Kvist, H-H., Arhomaa, S., Järvelin, K., Räikkönen, J. (1995). *Asiakasprosessit. Miten parannat tulosta prosesseja kehittämällä?* Helsinki: Sedecon Oy.

Laamanen, K. (2001). *Johda liiketoimintaa prosessien verkkona – ideasta käytäntöön*. Helsinki: Laatukeskus.

Lecklin, O. (2006). *Laatu yrityksen menestystekijänä*. Helsinki: Talentum.

Leopold, K., Kaltenecker, S. (2015). *Kanban change leadership: Creating a Culture of Continuous Improvement*. Hoboken: John Wiley & Sons, Inc. E-kirja. Luettu 21.2.2021. https://kamezproxy01.kamit.fi:2252/lib/kajaani-ebooks/reader.action?docID=1895926

Lin, F-R., Yang, M-C., Pai, Y-H. (2002). *A generic structure for business process modeling. Business Process Management Journal*, Vol. 8 No. 1, 2002, pp. 19-41. Luettu 29.10.2020. https://pdfs.se-man-
ticscholar.org/6341/9cfcc14a2b35ccbcac537edaeb314ed53c7e.pdf?_ga=2.207188051.1335538392.1603875157-312601251.1603875157

Martinsuo, M., Blomqvist, M. (2010). *Prosessien mallintaminen osana toiminnan kehittämistä*. Tampere: Tampereen teknillinen yliopisto. Luettu 6.3.2021. https://tutcris.tut.fi/portal/fi-les/2098668/prosessien_mallintaminen.pdf

Mäntyneva, M. (2000). *Asiakkuudenhallinta*. Helsinki: WSOY.

Schwaber, K., Sutherland, J. (2020). *Scrum-opas. Scrumin määritelmä ja pelisäännöt. Marraskuu 2020*. Luettu 21.2.2021. https://scrumwell.files.wordpress.com/2021/02/2020-scrum-guide-fin-nish.pdf

Toikko, T., Rantanen, T. (2009). *Tutkimuksellinen kehittämistoiminta*. Tampere: Tampereen Yli-opistopaino Oy.

Tuomi, J., Sarajärvi, A. (2013). *Laadullinen tutkimus ja sisällönanalyysi*. Helsinki: Kustannusosa-keyhtiö Tammi.

Ojasalo, K, Moilanen, T., Ritalahti, R. (2014). *Kehittämistyön menetelmät. Uudenlaista osaamista liiketoimintaan*. Helsinki: Sanoma Pro Oy.

Oksanen, T. (2010). *CRM ja muutoksen tuska. Asiakkuudet haltuun*. Helsinki: Talentum.

Peppers, D., Rogers, M. (2004). *Managing customer relationships*. Hoboken: Wiley.

Pigoski, T. (1997). *Practical Software Maintenance: Best Practices for Managing Your Software Investment*. New York: John Wiley & Sons, Inc.

Pohjonen, R. (2002). *Tietojärjestelmien kehittäminen*. Jyväskylä: Docendo

Vuorinen, T. (2013). *Strategiakirja: 20 työkalua*. Helsinki: Talentum. E-kirja. Luettu 21.2.2021. https://kamezproxy01.kamit.fi:2335/teos/CACBEXDTEB#/kohta:STRATEGIAKIRJA((20)-((20)20((20)TY((d6)KALUA((20)/piste:b1181

Wikipedia (2021a). *V-Model (software development).* Luettu 25.3.2021. https://en.wikipedia.org/wiki/V-Model_(software_development)

Wikipedia (2021b). *ITIL.* Luettu 25.3.2021. https://fi.wikipedia.org/wiki/ITIL

Teemahaastatteluissa esitetyt kysymykset:

I Haastattelukierros:

Aihe: Ohjelmiston ylläpito
1. Millaista ylläpitoa on (korjaavaa, täydentävää jne.) – haastateltavien perustiedot aiheesta?
2. Kuvailkaa näkemys ylläpidollisesta prosessista nykyisin ja toimiiko se kuvatulla tavalla?
3. Miten tieto ylläpidon tarpeesta tulee (millä tavalla puhelin, sähköposti vai joku muu)?
4. Miten ja miksi asiakkaat ottavat yhteyttä?
5. Ylläpidon määrittely, vaaditaanko asiakkailta lisäselvityksiä, ja jos vaaditaan, niin onko se tavallista vai osaavatko asiakkaat tuottaa kerralla tarpeellisen informaation?
6. Tehtävien määräytyminen työyhteisössä, kuka tekee ja mitä tekee?

Aihe: Asiakassuhteiden hallinta
1. Miten asiakkaisiin pidetään yhteyttä (sähköposti, puhelin vai joku muu)?
2. Asiakkaiden osallistuminen määrittelyihin, osallistuvatko?
3. Otetaanko asiakkaisiin muuten yhteyttä ja selvitetäänkö asiakkaiden muita tarpeita?
4. Jos asiakkailla on muita tarpeita, niin pyritäänkö ne täyttämään vaikkapa esim. ohjelmistoon lisättävillä lisäominaisuuksilla → jotta ne tuottaisivat asiakkaalle lisäarvoa?
5. Miten asiakkaille viestitään työnetenemisestä/ valmistumisesta?

II Haastattelukierros:

Aihe: Asiakassuhteiden hallinta
1. Yleistilanne (mistä asiakaskunta muodostuu)
2. Vuorovaikutuksen muodostuminen (ollaanko läsnä, vai tarjotaanko edellytyksiä asiakkaan omille prosesseille)
3. Asiakkaiden osallistuminen määrittelyihin
4. Asiakkaan odotusten seuraaminen – ennakointi muutoksiin – tarpeiden tunnistaminen
5. Asiakastyytyväisyyden seuraaminen – jälkimarkkinointi
6. Asiakaspalvelun mittaaminen

Aihe: Ohjelmiston ylläpito Datala Oy:ssä
1. Yleistilanne (mistä työt muodostuvat)
2. Ohjelmiston elinkaari – käyttöikä
3. Ohjelmiston ylläpito
    a. käsityksiä ylläpidosta ja ylläpidettävyydestä
    b. miten kuvaillaan ylläpitoa (korjaavaa, ennakoiva, muuttavaa jne.)
    c. mitenkä ylläpito tehtäviä tulee
    d. mistä osatekijöistä ylläpito muodostuu (prosessi)
    e. ylläpidon kehittyminen vuosien varrella -ohjelmiston ylläpidettävyys
4. Arvioita syistä, jotka johtavat ylläpito prosessiin
    a. ulkopuolisia tekijöitä
    b. Datala Oy:stä johtuvia

Esimerkki havaintopäiväkirjasta:
Havaintopäiväkirja – Opinnäytetyötä varten
Kirjataan ylläpitoon liittyvät havainnot. Asiakkaiden nimiä tai toimialaa ei mainita.

Päivämäärä	**Aika**	**Tapaus**	**Tapahtuma**
04.01.2021	10.30	1	Asiakas pyytää lisäselvitystä ohjelman antamasta ilmoituksesta. Ilmoituksessa mukana kuva ilmoituksesta. Sähköposti on tullut suoraan asian käsittelijälle.
04.01.2021	10.35	1	Ohjelman tutkiminen.
04.01.2021	10.45	1	Vastaaminen asiakkaalle ohjelman toiminnasta ko. tilanteessa.
04.01.2021	11.18	1	Asiakas kuittaa saamansa ohjeistuksen, ongelma ratkesi.
04.01.2021	14.44	2	Asiakkaan työntekijä ihmettelee puuttuvaa ohjelmaosiota. Ilmoitus tullut sähköpostilla yleiseen postiin.
04.01.2021	14.55	2	Esimies jakaa tehtävän.
04.01.2021	15.11	2	Selvitetty, että kyseessä on käyttöoikeuden puuttuminen ohjelmistoon.
04.01.2021	15.20	2	Käyttöoikeuden lisääminen ohjelmistoon ja kuittaus asiakkaalle.
04.01.2021	15.50	2	Asiakkaan työntekijä kuittaa – ongelma ratkesi.