Opinnäytetyö (YAMK)

Teknologiaosaamisen johtamisen koulutusohjelma

Tuotekehitys ja tuotteistaminen

2016

Sami Elmroos

DCISION 2.0

– Tietojärjestelmä kvantitatiivisen tutkimuksen ketterään toteutukseen

TURKU AMK
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (YAMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Teknologiaosaamisen johtamisen koulutusohjelma

2016 | 54

Ohjaajat: yliopettaja Timo Tolmunen, Toimitusjohtaja Jukka Pohjola

Sami Elmroos

DCISION 2.0 – TIETOJÄRJESTELMÄ KVANTITATIIVISEN TUTKIMUKSEN KETTERÄÄN TOTEUTUKSEEN

Henkilöstötutkimuksilla mitataan henkilöstön näkemyksiä ja kokemuksia esimerkiksi johtamisesta, työn mielekkyydestä tai tiedonkulusta. Sen avulla tuetaan organisaation ja sen henkilöstön kehittämistä. Työn toimeksiantaja toteuttaa asiakkailleen henkilöstötutkimuksia. Tutkimuksen toteutuksessa käytetään heidän itse kehittämiään tietojärjestelmiä. Toimeksiantajan toimiala on viime aikoina kehittynyt suuntaan, jossa asiakkaat tarvitsevat entistä nopeampia ja joustavampia henkilöstötutkimuksia.

Työn teoriaosuudessa perehdyttiin lähdekirjallisuuden pohjalta kvantitatiiviseen tutkimukseen, erilaisiin ohjelmistosuunnittelun teorioihin ja testausmenetelmiin. Opinnäytetyössä tutustuttiin toimeksiantajan nykyisiin ohjelmistokehitysprosesseihin sekä suunniteltiin niiden kehittämistä osana uuden ohjelmiston määrittelyä. Opinnäytetyössä luotiin toiminnallinen ja tekninen määrittely uudesta ohjelmistosta tutkimusprosessin läpivientiin. Osana määrityksiä kuvattiin, miten järjestelmää voidaan laajentaa ja jatkokehittää sisäisten sekä ulkoisten asiakastarpeiden pohjalta.

Lähtökohtana tässä työssä suunniteltavalle ohjelmistolle olivat toimeksiantajan nykyiset ohjelmistot sekä muuttunut liiketoimintakenttä uusineen asiakastarpeineen. Toimeksiantajan henkilökunnan näkemyksiä nykyisistä järjestelmistä ja niiden kehittämiskohteista kartoitettiin haastatteluiden avulla.

Työn tuloksena syntyi määrittely uudelle ohjelmistolle. Määrittelyssä on kuvattuna tarpeelliset toiminnallisuudet tutkimusprosessin eri vaiheisiin. Määrittelyn osana ovat myös kartoitettuna tarpeelliset tekniikat toiminnallisuuksien toteuttamiseen sekä testaamiseen.

Määritysten pohjalta kehittävä ohjelmisto antaa toimeksiantajalle kilpailuedun toimialallaan. Ohjelmisto on suunniteltu joustavaksi, jolloin toimeksiantajan on helppo jatkokehittää sitä tukemaan nopeasti muuttuvia asiakastarpeita. Määrityksissä on kehitetty tutkimusprosessin läpivientiä ohjelmiston näkökulmasta, joten ohjelmiston tuottaa toimeksiantajalle kilpailuedun lisäksi myös liiketoiminnallista lisäarvoa kustannustehokkaamman toiminnan muodossa.

Toimeksiantajan vaatimuksesta työn tuloksena syntynyt määritysdokumentti pidetään salattuna.

ASIASANAT:

ohjelmistosuunnittelu, kvantitatiivinen tutkimus, vaatimusmäärittelyt, tietojärjestelmät

MASTER´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Technological Competence Management

2016 | 54

Instructors: Principal Lecturer Timo Tolmunen, Managing Director Jukka Pohjola

Sami Elmroos

DCISION 2.0 – INFORMATION SYSTEM FOR AGILE IMPLEMENTATION OF QUANTATIVE RESEARCH

Personnels views and experiences for example from management, meaningfulness of the work or the flow of the information are measured with employee surveys. Survey supports development of the organization and personnel. Client of this thesis is conducting employee surveys to the customers. Implementing of the surveys are done by using their self-developed information systems. Client's industry has recently evolved in the direction, where the customers need faster and more flexible employee surveys.

In the theoretical part of the thesis, on the basis of the source literature, was to familiarize with quantitative research, various software engineering theories and testing methods. The thesis explored the client's existing software development processes, as well as developed them as part of the definition of the new software. The functional and technical specifications of the new software for the research process was created in this thesis. As part of the specifications are described how the system can be expanded and further developed based on the internal and external customer needs.

The starting point for software planned in this thesis were client's existing software and changed business field with new customer needs. The views of the client's staff about the existing systems and their development targets were mapped by interviews.

Result of the thesis was the definition of the new software. The definitions of the needed functionalities are described for the different stages of the research process. As a part of the definitions were also the necessary techniques for implementing the functionality and testing.

Software developed based on this thesis will give the client a competitive advantage in its sector. The software is designed to be flexible, so that the client is easy to further develop it to support the rapidly changing customer needs. The research process has been developed in the specifications by point of view of the software. For the client software will generate a competitive advantage, but also add value to a business in the form of cost-effective operations.

For the requirement of the client, definitions are concealed.

KEYWORDS:

Software design, quantitative research, requirement specifications, information systems

SISÄLTÖ

LIITTEET

KUVAT

TAULUKOT

SYMBOLI- JA LYHENNELUETTELO

f_i	havaintojen määrä moodiluokassa
N	havaintojen määrä
n	otoskoko
R	vaihteluväli
S	keskiarvon keskivirhe
s	keskihajonta
s^2	varianssi
V	variaatiokerroin
v	variaatiosuhde
x	muuttuja
$\bar{x}$	muuttujan keskiarvo
x_n	muuttujan havainnot
x_{max}	suurin havainto
x_{min}	pienin havainto
CSS3	Cascading stylesheets, CSS3 on verkkosivujen tyylin kuvauskieli. Sen kolmannen version ominaisuuksia standardoidaan tämän työn kirjoitus hetkellä.
HTML5	Hypertext Markup Language, HTML5 on standardoitu verkkosivujen kuvauskieli. Se on julkaistu vuonna 2014.
HTTP	HTTP-protokollan (Hyper Text Transfer Protocol) avulla määritetään liikenteen kulku asiakasohjelman ja palvelimen välillä. Protokolla määrittelee pyynnön ja vastauksen syntaksin.
HTTPS	HTTPS-protokolla (Hyper Text Transfer Protocol Secure) on suojattu versio http-protokollasta. Se tarkoittaa, että kaikki liikenne selaimen ja palvelimen välillä on salattua.
NLB	Network Load Balancing, on Microsoftin kehittämä tekniikka, jolla voidaan tasata palvelinten kuormaa. Asiakkaan ottaessa yhteyttä palvelimiin, NLB-palvelu valitsee palvelinklusterista vähiten kuormitetun palvelimen.
RPO	Recovery Point Objective, on suurin sallittu aika, jonka ajalta katastrofin sattuessa tiedot voidaan menettää.

RTO	Recovery Time Objective, on suurin sallittu aika, jonka katastrofin sattuessa palvelun käyttökatko saa kestää.
T-SQL	Transact-SQL, on Microsoftin kehittämä laajennus SQL-kieleen.
TLS	TLS-protokollalla (Transport Layer Security) salataan verkon yli liikkuva liikenne kahden pisteen välillä. Se on kooste useammasta korkeamman tason suojausprotokollassa, ja perustuu varmenteisiin joilla voidaan tunnistaa julkaiseva taho luotetuksi.
UML	Unified Modelling Language eli yhtenäistetty mallinnuskieli on vuonna 1997 julkaistu näkymän sisältöä kuvaava kieli, jossa yleisiä olio-ohjelmoinnin käsitteitä kuvataan mallinnuselementtien avulla. Näille on määritetty semantiikka eli muodollinen määritelmä.
XML	Extensible Markup Language, on standardoitu rakenteellinen kuvauskieli tiedon tallentamiseen.
XML-skeema	XML-skeema on standardoitu teknologia, jolla voidaan kuvata XML-dokumenttien rakenne.

1 JOHDANTO

Henkilöstötutkimuksella mitataan henkilöstön näkemyksiä ja kokemuksia esimerkiksi johtamisesta, työn mielekkyydestä tai tiedonkulusta. Sen avulla tuetaan organisaation ja sen henkilöstön kehittämistä. Tutkimuksen avulla voidaan tuottaa yrityksen johdolle tutkittua tietoa yrityksen vahvuuksista ja kehittämiskohteista. Henkilöstötutkimusten etuna on niiden toistettavuus ja vertailtavuus erilaisiin mittareihin. Tällaisia mittareita voivat olla esimerkiksi yrityksen aiemmat henkilöstötutkimus tulokset, yrityksen toimialan yleiset tulokset tai taloudelliset mittarit. (Corporate Spirit 2015)

Toimeksiantaja toteuttaa asiakkailleen määrällisiä henkilöstötutkimuksia. He ovat konseptoineet palvelut henkilöstön omistautuneisuuden mittaamiseen sekä 360-arviointien tekemiseen. Tutkimuksen toteutuksessa käytetään heidän itse kehittämiään ohjelmistoja. Toimeksiantajan nykyisissä ohjelmistoissa laaja normitietopankki on paras kilpailutekijä sekä asiakashyöty. Sen avulla asiakas voi verrata tuloksiaan esimerkiksi toimialansa yleiseen tulostasoon.

Toimeksiantajan palveluksessa on noin 30 henkeä. Heistä suurin osa työskentelee tutkimusten myynnin, toteutuksen sekä sen hyödyntämisen parissa.

1.1 Opinnäytetyön aihe ja tavoitteet

Toimeksiantajan toimiala on viime aikoina kehittynyt suuntaan, jossa asiakkaat tarvitsevat entistä nopeampia ja joustavampia henkilöstötutkimuksia. Heidän nykyiset ohjelmistonsa ovat suunniteltu tukemaan konseptoitujen tutkimuspalveluiden toteutusta. Siitä johtuen ne eivät ole kovin joustavia muunlaisten tutkimusten toteuttamiseen. Toimeksiantaja on kartoittanut kilpailijoiden ohjelmistojen ominaisuuksia. Ne on huomioitu toimeksiantajan nykyisissä ohjelmistoissa. Ostettavissa olevien ohjelmistojen kartoituksissa ei ole löytynyt sopivaa ohjelmistoratkaisua tai -määrityksiä asiakastarpeiden mukaisten tutkimusten toteuttamiseen.

Toimeksiantajan tutkimusten tekemistä ohjaavat vahvasti henkilötietolaki, toimeksiantajan laatujärjestelmä sekä markkina- ja mielipidetutkimusalan kansainvälisen kattojärjestön

ESOMARin ohjeistus tutkimusten tekemiseen. Henkilötietolaissa määritetään yksilön yksityisyyden ja yksityiselämän suoja (Tietosuojavaltuutetun toimisto 2013). ESOMAR on laatinut tutkimusten toteuttamiseen eettiset ohjeistukset (ESOMAR 2016). Toimeksiantajalla on käytössään laatujärjestelmä, joka on akkreditoitu eli sille on myönnetty tutkimusalan arvostettu kansainvälinen ISO 20252 -sertifikaatti vuodesta 2010 lähtien.

Lähtökohtana tässä työssä suunniteltavalle ohjelmistolle ovat toimeksiantajan nykyiset ohjelmistot sekä muuttunut liiketoimintakenttä uusineen asiakastarpeineen. Sidosryhmiä työn näkökulmasta ovat toimeksiantajan henkilöstö eli sisäiset asiakkaat, ulkoiset asiakkaat sekä toimeksiantajan omistajat, joille työssä määritettävän ohjelmiston tulisi tuottaa liiketoiminnallista arvoa.

Työn tarkoitus on tuottaa toimeksiantajalle selkeä kilpailuetu heidän toimialallaan. Työssä määriteltävän ohjelmiston tavoitteena on luoda määritykset tietojärjestelmälle, jonka avulla voidaan toteuttaa räätälöityjä määrällisiä tutkimuksia. Ohjelmiston valmistuttuaan tulisi kasvattaa toimeksiantajan kilpailukykyä joustavuudellaan ja ketteryydellään.

Ohjelmiston avulla voidaan suorittaa tutkimuksen suunnittelu, tiedonkeruu sekä perustason raportointi sekä sen tulee mahdollistaa räätälöityjen tutkimusten toteuttaminen asiakkaille tietoturvallisesti, kustannustehokkaasti ja nopeasti. Ohjelmiston tulee tukea hyvinkin erilaisten asiakastarpeiden toteuttamista ja olla helposti skaalattavissa ja jatkokehitettävissä tulevien asiakastarpeiden mukaan.

Työn lopputuotoksena tulisi syntyä kattava toiminnallinen ja tekninen määrittely uudesta ohjelmistosta. Määrityksissä tulisi olla kuvattuna ohjelmiston toiminta sekä tekninen määritys. Näiden lisäksi lopputuotoksessa tulisi olla kuvattuna miten ohjelmistoa voidaan laajentaa sekä jatkokehittää sisäisten ja ulkoisten asiakastarpeiden pohjalta.

1.2 Opinnäytetyössä käytetyt menetelmät

Työssä tutustutaan ensin määrällisen eli kvantitatiivisen tutkimuksen toteuttamiseen lähteiden kautta. Tämän avulla varmistetetaan käsitteiden ja tutkimusteorian riittävä tuntemus. Määrällisessä tutkimusmenetelmässä eli kvantitatiivisessa menetelmässä tietoa tarkastellaan numeerisesti. Se vastaa kysymyksiin kuinka moni ja kuinka usein. Kvantitatiivisessa tutkimuksessa aineisto kerätään numeerisessa muodossa. (Vilkka 2007, 14.)

Ohjelmiston määrittämistä varten tutustutaan vaatimusmäärittelyyn lähdemateriaalin avulla. Vaatimusmäärittely on ohjelmistokehityksen yksi perusvaatimuksista. Sitä on tehty yhtä kauan kuin on tehty ohjelmistojakin. Alkunsa sen voidaan ajatella saaneen 1993 pidetyssä ensimmäisessä vaatimusmäärittelykonferenssissa. (Paakki 2011, 2.)

Työ rajattiin vain ohjelmiston määrittämiseen, jättäen teknisen toteutuksen työn ulkopuolelle. Toimeksiantajan vaatimuksen mukaan uuden ohjelmiston määritykset pidetään salattuna. Tällä varmistetaan uuden ohjelmiston kehittämisestä saatava kilpailuetu. Työssä syntyneet määritykset jaoteltiin vaatimusmäärittelyn mukaisesti ennalta määrättyihin osioihin. Näitä ovat ohjelmiston yleiskuvaus, toiminnalliset vaatimukset-, ei-toiminnalliset vaatimukset-, käyttöliittymät-, tiedot ja tietokannat - sekä toteutus osiot. Ohjelmiston yleiskuvauksessa määritellään käyttötarkoitus, käyttäjät sekä liittymät muihin ohjelmistoihin.

Työssä toiminnalliset vaatimukset luokitellaan seuraavasti; käyttäjien kertomat ongelmat, yleiset rajoitukset, mahdolliset lisätoiminnot sekä käyttötapauskaaviot. Käyttäjien kokemuksia toimeksiantajan nykyisistä ohjelmistoista kartoitettiin haastattelemalla käyttäjiä sekä nykyisten ohjelmistojen kehittäjiä. Haastattelut tehtiin ajalla 8.2. – 19.2.2016. Haastateltavilta pyydettiin myös näkemyksiä siitä, miten he kokevat toimeksiantajan liiketoimintaympäristön kehittyvän lähivuosina. Haastattelun rakenne muodostettiin vaatimusmäärittelyn pohjalta siten, että kartoitettiin nykyisen ohjelmiston haasteet ja haastateltavan havainnot nykyisistä ja tulevista asiakastarpeista. Edelle mainittuja kohtia tarkennettiin valittujen tutkimusprojektin osioiden pohjalta.

Ohjelmistosta luodaan käyttötapauskaaviot haastatteluiden pohjalta. Niissä kuvataan tärkeimmät toiminnallisuudet käyttäjänäkökulmittain. Jokaisesta ohjelmiston osiosta tehdään käyttötapauskaavio. Käyttötapausmallissa määritellään toiminto ja etsitään toimijat sekä käyttötapaukset. Sen avulla kuvataan mitä ohjelmiston tulisi tehdä. Käyttötapauskaavio on Ivar Jacobsonin kehittämä tekniikka. Sen ei ole tarkoitus kuvata miten ohjelmisto toteuttaa kuvatut toiminnallisuudet. (Järvelä & Puusaari 2005, 9.)

Työssä ei-toiminnalliset vaatimukset luokitellaan luokkiin; käytettävyys, tietoturva, ylläpidettävyys ja huollettavuus, siirrettävyys, laajennettavuus ja uudelleenkäytettävyys, konfiguroitavuus sekä suorituskyky. Suorituskykytavoitteen asettamista varten tutustutaan erilaisiin tutkimuksiin käyttäjien kokemuksista suorituskyvyn riittävyyteen.

Ohjelmiston käyttöliittymät suunnitellaan tukemaan useita eri laitealustoja, pöytätietokoneista älypuhelimiin. Se huomioidaan käytettävyyttä suunnitellessa siten, että toiminnot sijoitellaan näkymiin selkeästi erilleen toisistaan, helpottamaan käyttöä mobiililaitteilla. Lisäksi mobiilitukea pyritään parantamaan huomioimalla suunnittelussa näkymien responsiivisuus. Responsiivisuudella tarkoitetaan tilannetta, jossa näkymän elementit sijoitellaan näytölle siten, että sama näkymä tukee kaiken kokoisia laitteita. Responsiivisuus voidaan toteuttaa käyttämällä HTML5:n sekä CSS3:n uusia tekniikoita (Leiniö 2012). Käyttöliittymien suunnitelmissa huomioidaan Nielsenin säännöt sekä Fittsin laki. Nielsenin säännöt on luonut yksi käytettävyyden edistäjistä tohtori Jacob Nielsen (Nielsen 1995a). Fittsin lain avulla pohditaan käyttöliittymien eri elementtien saavutettavuutta (Häkkinen 2014.).

Tietokannan suunnittelun pohjana käytetään työssä luotavia käyttötapauskaavioita sekä käyttöliittymän näkymien mallikuvia. Niiden avulla kartoitetaan mitä tietoa pitäisi tallentaa tietokantaan. Tietokannan tauluista luodaan relaatiomallit jotka normalisoidaan (Microsoft 2008).

Työssä toteutusvaiheen suunnitelmat jaotellaan seuraaviin luokkiin; vaihejakomalli, ohjelmistonarkkitehtuurikuvaus, ohjelmistokehys, lähdekoodin laatu, järjestelmäsuunnittelu, testaus, julkaisu sekä alustava aikataulu.

2 KVANTITATIIVINEN TUTKIMUS

Määrällisessä tutkimusmenetelmässä eli kvantitatiivisessa menetelmässä tietoa tarkastellaan numeerisesti. Se vastaa kysymyksiin kuinka moni ja kuinka usein. Kvantitatiivisessa tutkimuksessa aineisto kerätään numeerisessa muodossa tai laadullisen tutkimuksen eli kvalitatiivisen tutkimuksen aineisto muunnetaan numeeriseen muotoon. Tutkija esittää tulokset numeroina eli tunnuslukuina. Tuloksissa kuvataan miten eri asiat liittyvät toisiinsa. (Vilkka 2007, 14.)

2.1 Mittaaminen

Mittaaminen tarkoittaa erilaisten asioiden määrittämistä mitta-asteikoille. Määrällisessä tutkimuksessa mittaamista on kaikki, jossa määritellään asioille eroja mitta-asteikolla. Tutkimuksen tulee aina olla mahdollisimman objektiivinen ja puolueeton. Tuloksia tulkitessa ne voidaan asettaa moneen eri viitekehykseen. Määrällisessä tutkimuksessa on tyypillistä, että vastaajia on paljon. (Vilkka 2007, 16 – 17.)

Normaalisti tutkija asettaa tutkimusongelman eli hypoteesin. Tämän väitteen paikkaansa pitävyyttä pyritään selvittämään tutkimuksen avulla. Nollahypoteesi tarkoittaa tilannetta, jossa tutkimustulokset eivät vastaa asetettua hypoteesia. Tämä on osaltaan merkittävä tilanne, mikäli hypoteesi perustuu esimerkiksi teoriatiedon pohjalta tehtyyn väittämään. (Vilkka 2007, 24 – 25.)

Kysely on tiedonkeruun tapa, jossa jokaiselta vastaajalta kysytään samat asiat samassa järjestyksessä ja samalla tavalla. Kysely on hyvä aineiston keräämiseen kun vastaajia on paljon ja he ovat hajallaan. Kysely voidaan tehdä sekä paperilla että internetissä. Paperisen tiedonkeruun haasteina ovat vastausten palautumisen hitaus sekä niiden käsittely. Kyselyn ajoittaminen on osaltaan erittäin tärkeää, ettei sen vuoksi vastausprosentti jää alhaiseksi. Vastaamispyyntöä usein seuraakin muistutus vastaamisesta. (Vilkka 2007, 28.)

Mitattavan asia määrittämistä mitta-asteikolle kutsutaan operationalisoinniksi. Mitattavat teoreettiset asiat määritellään kysymyksiksi sekä vastausvaihtoehdoiksi. Toisin sanottuna jokaisen vastaajan samalla tavalla ymmärtämäksi arkikieleksi. Mikäli operationalisointia ei tehdä huolellisesti, se voi johtaa siihen, että kysymyslomake ei mittaa sitä mitä oli tarkoitus.

Hyvin tehdyn operationalisoinnin kautta saadut mittaustulokset ovat helposti muutettavissa takaisin teoreettiselle tasolle. (Vilkka 2007, 36 – 44.)

Erilaisia mitta-asteikoita asioiden mittaamiseen on useita. Mitta-asteikoilla erotellaan muuttujien ilmaisemia asioita. Muuttuja voivat olla esimerkiksi sukupuoli, ikä, ammattiasema, pituus tai paino. Mitta-asteikkoja ovat esimerkiksi laatuero-, järjestys-, välimatka- ja suhdelukuasteikko. (Vilkka 2007, 16, 45.)

Laatueroasteikolla eli nominaaliasteikolla mitataan asioita, jotka ovat jaettavissa ryhmiin tai luokkiin. Tällaisia mitattavia asioita ovat esimerkiksi sukupuoli tai ammatti. Paras keskiluku laatueroasteikolle on moodi. (Tilastokeskus 2002; Vilkka 2007, 48.)

Järjestysasteikoilla voidaan mitata luokkien järjestystä. Tällainen luokka on esimerkiksi ikäluokka. Paras keskiluku järjestysasteikolle on moodi tai mediaani. (Tilastokeskus 2002; Vilkka 2007, 49.)

Välimatka- eli intervalliasteikolla mitataan havaintojen etäisyyttä toisistaan. Siinä etäisyyden mittapisteiden välillä tulee olla täysin samanpituinen. Välimatka-asteikolla voidaan mitata esimerkiksi pituutta tai painoa. Paras keskiluku välimatka-asteikolle on aritmeettinen keskiarvo. (Tilastokeskus 2002; Vilkka 2007, 49 – 50.)

Suhdelukuasteikko eroaa välimatka-asteikosta siten, että sen nollapiste on todellinen. Sillä voidaan mitata esimerkiksi tuloeroja. (Tilastokeskus 2002; Vilkka 2007, 50.)

Asenneasteikoilla mitataan vastaajan kokemukseen perustuvaa mielipidettä mitattavasta asiasta tai asennetta sitä kohtaan. Useimmiten käytetyistä asenneasteikoista tunnetuimmat ovat Likertin asteikko ja Osgoodin asteikko. (Vilkka 2007, 45.)

Likertin asteikolla keskikohdasta lähtien toisella puolella ovat saman mieliset vastausvaihtoehdot ja toisella puolella eri mieliset vastausvaihtoehdot (Taulukko 1). (Vilkka 2007, 46.)

Taulukko 1. Esimerkki Likertin asteikosta neliportaisena.

1	Täysin eri mieltä
2	Jokseenkin eri mieltä
3	Jokseenkin samaa mieltä
4	Täysin samaa mieltä

Osgoodin asteikolla ääripäihin sijoittuvat vastakkaiset asteikot (Taulukko 2.). (Vilkka 2007, 47.)

Taulukko 2. Esimerkki Osgoodin asteikosta viisiportaisena.

Kyselyn täyttäminen oli helppoa	1	2	3	4	5	**Kyselyn täyttäminen oli vaikeaa**

2.2 Otanta

Otannalla tarkoitetaan perusjoukon osaa, jolla saadaan kokonaiskuva koko tutkimuksen kohderyhmästä. Perusjoukko kattaa koko kohdejoukon, josta halutaan saada tietoa. Perusjoukosta valitaan tutkimuksen otanta käyttämällä otantamenetelmiä. Näitä ovat esimerkiksi yksinkertainen satunnaisotanta, systemaattinen otanta, ositettu otanta tai ryväsotanta. (Vilkka 2007, 52.)

Yksinkertaisessa satunnaisotannassa havaintoyksiköt eli tutkimuskohteet valitaan satunnaisesti perusjoukosta. Systemaattisessa otannassa havaintoyksiköt valitaan esimerkiksi aakkosjärjestyksessä siten, että havaintoyksiköitä valitaan 5 yksikön välein tasaisesti koko perusjoukosta. Ositetussa otannassa perusjoukko ryhmitellään ja jokaisesta ryhmästä valitaan saman verran havaintoyksiköitä. Ryväsotannassa perusjoukko koostuu yleensä jostain luonnollisista ryhmistä, esimerkiksi organisaatioista. Tutkimuskohteena voi tällöin olla tällainen ryhmä. (Vilkka 2007, 53 – 55)

Tutkimus voidaan tehdä myös kokonaisotantana, jolloin koko perusjoukko on mukana. Tätä suositellaan, mikäli valittu otantamenetelmä sisältäisi yli puolet perusjoukosta. (Vilkka 2007, 52.)

Otoskoon tulisi olla niin suuri, että sen voidaan sanoa edustavan kokonaisvaltaisesti perusjoukkoa. Otoskoon riittävyyttä voidaan myös arvioida laskemalla keskiarvolle keskivirhe. Keskivirhe voidaan laskea kaavalla

$$S = \frac{s}{\sqrt{n}} \tag{1}$$

jossa S on keskiarvon keskivirhe, s on otoksesta laskettu muuttujan keskihajonta ja n on otoskoko. (KvantiMOTV 2004; Vilkka 2007, 56 – 57.)

Otoskoon arvioinnissa tulisi huomioida myös mahdollinen kato. Kadosta puhutaan kun havaintoja puuttuu esimerkiksi vastaamatta jättämisen vuoksi. (Vilkka 2007, 60.)

2.3 Kysymyslomake

Kysymyslomake tulee suunnitella huolella, sillä sen pitää mitata oikeita asioita. Lomakkeen suunnittelijan tulee varmistaa, että kyselyyn vastaavalla henkilöllä on mahdollisuus vastaamiseen sekä siihen tarvittavat tiedot. Esimerkiksi sähköisessä tiedonkeruussa on varmistettava, onko jokaisella vastaajalla mahdollisuus vastata internetissä tapahtuvaan tiedonkeruuseen. Normaalisti kysymyslomake koostuu saatesivusta ja kysymyksistä. (Vilkka 2007, 63 – 64.)

Saatesivun on tarkoitus kertoa vastaajalle mitä tutkimuksella mitataan sekä miten aineistoa käsitellään. Lisäksi siinä tulisi olla ohjeet miten lomakkeeseen vastataan sekä tieto vastaamisen luottamuksellisuudesta. Tutkimusalan sertifikaatti ISO 20252 määrittelee erittäin tarkkaan saatesivun sisällön. (Vilkka 2007, 81.)

Lomakkeen kysymysten tulisi olla muotoiltu ja järjestetty siten, ettei virhetulkintojen vaaraa ole. Lomakkeen eteneminen pitäisi suunnitella siten, että lomakkeen eteneminen on järkevää ja loogista. (Vilkka 2007, 71.)

Lomakkeen testaaminen on erittäin tärkeää. Virheitä ei pysty korjaamaan enää tiedonkeruun päättymisen jälkeen. Tästä syystä huonosti testattu lomake voi vaarantaa koko tutkimuksen luotettavuuden. Esimerkiksi tilanteessa, jossa lomakkeella on ollut mahdollista ymmärtää kysymykset useammalla eri tavalla. (Vilkka 2007, 78.)

2.4 Aineiston tarkastus

Vastausten palautuessa tai vasta tiedonkeruun päättyessä tutkija tarkistaa aineiston. Tarkistaessa arvioidaan palautuneiden vastausten laatu ja poistetaan asiattomasti täytetyt lomakkeet. Tärkein asia aineiston tarkistuksessa on arvioida tutkimuksen kato käyttäen katoanalyysia. Katoanalyysi voi osoittaa, että aineisto on systemaattisesti vinoutunut. Tässä tilanteessa tutkija voi painottaa havaintoja vastaamaan paremmin tutkimuksen perusjoukkoa. (Vilkka 2007, 106 – 107.)

Mikäli vastaajat ovat syystä tai toisesta jättäneet vastaamatta kaikkiin kysymyksiin tai ovat valinneet ”en tunne asiaa” -vaihtoehdon, tutkija voi jättää nämä vastauslomakkeet kokonaan pois tai käsitellä ne omana luokkanaan. Jos aineisto on pieni eikä tutkija halua pienentää aineistoa, voidaan puuttuvien havaintojen arvot korvata aineiston keskiarvolla. ”En tunne asiaa” -vaihtoehdot voidaan myös käsitellä omana luokkanaan. (Vilkka 2007, 108 – 109.)

Aineistosta muodostetaan havaintomatriisi eli datamatriisi, jossa tiedot jaotellaan vastaajittain taulukkoon (Taulukko 3.). Vaakarivit edustavat vastaajia ja pystysarakkeet vastauksia kysymyksittäin. (Vilkka 2007, 111.)

Taulukko 3. Esimerkki datamatriisista.

Vastaaja	Kysymys 1	Kysymys 2	Kysymys 3	…	Kysymys N
Muuttujan arvot	(1,2,3,4)	(1,2,3,4)	(1,2,3,4)	…	(1,2,3,4)
Vastaaja 1	2	4	1	…	1
Vastaaja 2	3	4	2	…	3
…	…	…	…	…	…
Vastaaja N	4	1	3	…	4

2.5 Aineiston analyysi

Analyysit valitaan siten, että analyysimenetelmä antaa tietoa siitä, mitä ollaan tutkimassa. Analyysin valintaan vaikuttaa myös ollaanko tutkimassa yhtä muuttujaa vai kahden muuttujan välistä suhdetta. (Vilkka 2007, 119.)

2.5.1 Keskiluvut

Moodilla voidaan tutkia mikä on aineiston tyyppiluku eli mikä muuttuja esiintyy aineistossa useimmin. (Vilkka 2007, 121.)

Mediaani kuvaa aineiston keskikohtaa eli kohtaa, jossa havaintoja on mediaanin molemmin puolin lukumääräisesti yhtä paljon. Mikäli havaintoja on parillinen määrä, on mediaani kahden keskimmäisen havainnon aritmeettinen keskiarvo. (Vilkka 2007, 122.)

Aritmeettinen keskiarvo kuvaa havaintojen keskimääräistä suuruutta. Se lasketaan ottamalla havaintojen summa ja jakamalla se havaintojen lukumäärällä.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{2}$$

jossa tilastomuuttujan x keskiarvoa merkitään $\bar{x}$ ja havainnot ovat $x_1, x_2, \ldots, x_n$. (Vilkka 2007, 122 – 123.)

2.5.2 Variaatiosuhde ja vaihteluväli

Variaatiosuhteella kuvataan sitä kuinka suuri osuus havainnoista on muuttujan moodiluokassa. Se voidaan laskea kaavalla

$$v = 1 - \frac{f_i}{N} \tag{3}$$

jossa v on variaatiosuhde, f_i havaintojen määrä moodiluokassa ja N on havaintojen määrä. Mikäli kaikki havainnot ovat lähellä toisiaan, on variaatiosuhde lähellä nollaa. Jos taas variaatiosuhde lähenee yhtä, on aineisto hajaantunut. (Vilkka 2007, 123.)

Vaihteluväli kertoo pienimmän ja suurimman havainnon välin. Vaihteluväli lasketaan asettamalla havainnot suuruusjärjestykseen ja vähentämällä suurimmasta havainnon arvosta pienin havainnon arvo. Se voidaan laskea kaavalla

$$R = x_{max} - x_{min} \tag{4}$$

jossa R on vaihteluväli, x_{max} suurimman havainnoin arvo sekä x_{min} pienimmän havainnon arvo. (Vilkka 2007, 124.)

2.5.3 Varianssi, keskihajonta sekä variaatiokerroin

Varianssi kuvaa havaintojen jakautumista aritmeettisen keskiarvon ympäristöön. Se voidaan laskea kaavalla

$$s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 \quad (5)$$

jossa s^2 on varianssi. (Vilkka 2007, 124.)

Keskihajonta kuvaa, kuinka kaukana yksittäiset havainnot ovat keskimäärin aritmeettisesta keskiarvosta. Keskihajonta saadaan laskemalla varianssin neliöjuuri

$$s = \sqrt{s^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \quad (6)$$

jossa s on keskihajonta. (Vilkka 2007, 124.)

Variaatiokerroin on keskihajonnan ja keskiarvon välinen suhde. Sillä voidaan ilmaista kahden eri asteikolla olevan muuttujan arvojen suhteellista hajontaa. Se voidaan laskea kaavalla

$$V = \frac{s}{\bar{x}} \quad (7)$$

jossa V on variaatiokerroin, s on keskihajonta sekä $\bar{x}$ on havaintojen keskiarvo. (Vilkka 2007, 125.)

2.5.4 Ristiintaulukointi ja korrelaatiokerroin

Ristiintaulukoinnilla voidaan tarkastella kahden eri muuttujan välisiä riippuvuuksia. Ristiintaulukoinnissa vastaukset jaotellaan yksittäisen muuttujan arvojen mukaisiin luokkiin, esimerkiksi miehet ja naiset. (Vilkka 2007, 129.)

Korrelaatiokertoimen avulla voidaan laskea kahden muuttujan välistä suhdetta. Korrelaatio kuvaa riippuvuuden suuntaa, voimakkuutta sekä yhteisvaihtelun olemassaoloa. Se voidaan laskea esimerkiksi Pearsonin tulomomenttikorrelaatiokertoimen avulla

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{ns_x s_y} \quad (8)$$

jossa n on lukuparien x_i ja y_i lukumäärä, s_x sekä s_y ovat muuttujien x ja y keskihajonnat ja $\bar{x}$ sekä $\bar{y}$ ovat muuttujien x ja y keskiarvot. (Vilkka 2007, 130.)

2.6 Tulosten esittäminen

Määrällisen tutkimuksen tuloksia voidaan esittää taulukkomuodossa, kuvioin, tunnuslukuina sekä tekstinä. Tulosten esittämisen tulee olla objektiivista eli tuloksia ei pidä painottaa siten, että tulosten lukija voi saada niistä vääränlaisen kuvan. (Vilkka 2007, 135.)

Taulukkomuoto sopii esitystavaksi silloin, kun esitettävää tietoa on paljon ja se halutaan esittää yksityiskohtaisesti. Kuviot helpottavat tulosten nopeaa esittämistä ja ne antavat helposti yleiskuvan tuloksista. Tunnusluvut esittävät havainnollisesti yksittäisen muuttujan tulokset. Tekstimuodossa tutkija voi selittää tuloksia. (Vilkka 2007, 135.)

Erilaisia kuvioita voivat olla esimerkiksi pylväskuvio, piirakkakuvio, viivakuvio tai aluekuvio. Näillä voidaan esittää tuloksia graafisesti. (Vilkka 2007, 139 – 147.)

2.7 Tutkimuksen reliabiliteetti

Tutkimuksen reliabiliteetti eli luotettavuus on hyvä silloin, kun otanta edustaa perusjoukkoa kattavasti. Lisäksi mittaamisessa tulisi olla mahdollisimman vähän satunnaisvirheitä. Määrällinen tutkimus tulee aina tehdä tieteelliselle tutkimukselle asetettujen vaatimusten mukaan. (Vilkka 2007, 152 – 154.)

3 OHJELMISTON MÄÄRITYKSET

Ohjelmiston määrittämistä varten tutustuttiin vaatimusmäärittelyyn. Vaatimusmäärittelystä ja sen eri osioista etsittiin riittävästi lähdemateriaalia määritysten tekemisen tueksi.

Vaatimusmäärittelyssä ohjelmistolle asetetaan toiminnallisia sekä ei-toiminnallisia vaatimuksia. Ne kuvataan sopivalla tavalla ohjelmiston ominaisuuksiksi. Niissä kuvataan mitä ohjelmistolta vaaditaan sekä miten jatkokehitys toteutetaan. Nykyisin puuttuvat ja virheelliset vaatimusmäärittelyt ovat yksi suurimmista syistä ohjelmistoprojektien epäonnistumisiin. (Paakki 2011, 3 – 7.)

Vaatimusmäärittelyssä on tarkoitus vastata kolmeen kysymykseen. Mitä tehdään, miksi tehdään ja kuka tekee. Usein lähtökohtana vaatimusmäärittelylle on nykyinen ohjelmisto tai prosessi. Sen ei välttämättä tarvitse olla olemassa oleva ohjelmisto, vaan se voi olla myös prosessi tai toimintatapa. Joissakin tilanteissa ohjelmiston lähtökohtana voi siis olla nykyinen tekeminen. Tässä tapauksessa ohjelmisto suunnitellaan korvaamaan aiempi usein vanhanaikainen prosessi. Nykyistä mallia tutkimalla voidaan määrittää tarpeet ja tavoitteet uudelle ohjelmistolle. (Paakki 2011, 7 – 14.)

Toiminnalliset vaatimukset vastaavat siihen, miten tuleva ohjelmisto vaikuttaa ympäristöönsä, eli mitä se tekee. Ei-toiminnalliset vaatimukset taas määrittävät miten ohjelmisto täyttää toiminnalliset vaatimuksensa. (Paakki 2011, 27 – 28.)

Ei-toiminnallisia vaatimuksia voivat olla esimerkiksi ohjelmiston laatu-, mukautuvuus-, arkkitehtuuri- sekä kehitystyön vaatimukset. (Paakki 2011, 30.)

Ohjelmiston toimintaympäristö voidaan jakaa osiin järjestelmä- ja ohjelmistovaatimukset. Järjestelmävaatimukset kertovat miten tuleva ohjelmisto suhteutuu ympärillä oleviin järjestelmiin tai syötteisiin. Järjestelmävaatimuksia voivat esimerkiksi olla ohjelmiston rajapinnat tai ohjelmistoa suorittava laitteisto. Ohjelmistovaatimuksilla määritetään ohjelman sisällä tapahtuvat asiat. Ne ovatkin lähtökohtaisesti vain ohjelmistokehittäjille tarkoitettuja määrityksiä. (Paakki 2011, 19 – 23.)

Vaatimusmäärittelyn vaiheet voidaan jakaa karkeasti luokkiin ongelmakentän ymmärtäminen, vaatimusten kartoitus sekä vaatimusten arviointi. (Paakki 2011, 36 – 38.)

Vaatimusmäärittelyn oikeellisuuden varmistamiseksi sille tehdään lopuksi muodollinen tarkastus. Tarkastajina tulee ehdottomasti olla ohjelmiston tulevia käyttäjiä. (Joensuun yliopisto 2007.)

Opinnäytetyössä ohjelmiston määritykset jaoteltiin osioihin yleiskuvaus, toiminnalliset vaatimukset, ei-toiminnalliset vaatimukset, käyttöliittymät tiedot ja tietokannat, toteutus.

3.1 Ohjelmiston yleiskuvaus

Yleiskuvauksessa määriteltiin ohjelmiston käyttötarkoitus, käyttäjät sekä liittymät muihin ohjelmistoihin. Ohjelmiston tulee olla soveltuva toteuttamaan räätälöityjä määrällisiä tutkimuksia toimeksiantajan asiakkaille. Sen tulee kyetä toteuttamaan tutkimusprojektin suunnittelu-, tiedonkeruu- sekä raportointivaiheet joustavasti ja ketterästi. Ohjelmiston suunnittelun osioiksi valittiin tutkimuksen mallintaminen, tutkimuksen suunnittelu, tutkimuksen tiedonkeruu sekä tutkimuksen raportointi.

Mallintamisella kuvataan tutkimuksissa yleisesti käytettäviä asioita, jotta niitä olisi joustavampi liittää tutkimuksiin projekteissa. Sen avulla voidaan mallintaa esimerkiksi usein käytetty lomakepohja, jolloin sitä ei tarvitse luoda alusta alkaen jokaiseen tutkimusprojektiin.

Suunnitteluosio jakautuu tutkimusprosessin mukaisiin vaiheisiin. Nämä vaiheet ovat projektin aikataulu, tutkimuksen sisältö, otannan hallinta, käyttäjien hallinta. Tiedonkeruuosiossa määritellään tutkimuksen tiedonkeruu eli kyselylomake, vastaajat sekä aineiston käsittely ja datan validointi. Tutkimuksen raportointi -osiossa määritellään miten tutkimustulokset raportoidaan tai esitetään.

3.1.1 Käyttötarkoitus

Ohjelmiston käyttötarkoituksena on tutkimusprosessin läpivienti. Sen tulee olla helposti jatkokehitettävä, jotta sen avulla voidaan reagoida nopeasti muuttuviin asiakastarpeisiin.

3.1.2 Käyttäjät

Työssä käyttäjiksi määriteltiin toimeksiantajan henkilökunta ja asiakkaat. Nämä jaoteltiin kahteen käyttäjänäkökulmaan siten, että toimeksiantajan henkilökunta edustaa sisäisiä käyttäjiä sekä asiakaskäyttäjät ulkoisia käyttäjiä.

3.1.3 Liittymät muihin ohjelmistoihin

Ohjelmiston tulee liittyä toimeksiantajan olemassa olevaan tutkimusohjelmistoon. Toimeksiantajan nykyisessä ohjelmistossa on erittäin kehittyneet tutkimuksen raportointitoiminnot. Ne on suunniteltu toimeksiantajan konseptoimille tutkimustuotteille. Näiden tuotteiden osalta työssä suunniteltava ohjelmisto toteuttaa siis vain tutkimuksen suunnittelu- sekä tiedonkeruuvaiheet.

3.2 Toiminnalliset vaatimukset

Työssä toiminnalliset vaatimukset luokiteltiin seuraaviin luokkiin; käyttäjien kertomat ongelmat, yleiset rajoitukset, mahdolliset lisätoiminnot sekä käyttötapauskaaviot.

3.2.1 Käyttäjien kertomat ongelmat

Käyttäjien kokemia ongelmia sekä haasteita toimeksiantajan nykyisessä ohjelmistossa kartoitettiin haastattelemalla käyttäjiä sekä nykyisten ohjelmistojen kehittäjiä. Haastateltavat valikoitiin toiminnoittain siten, että kustakin tutkimusprojektin osiosta valittiin edustaja haastateltavaksi. Haastattelut tehtiin ajalla 8.2.2016 – 19.2.2016.

Haastattelun rakenne muodostettiin vaatimusmäärittelyn pohjalta siten, että kartoitettiin nykyisen ohjelmiston haasteet ja haastateltavan havainnot nykyisistä ja tulevista asiakastarpeista. Edelle mainittuja kohtia tarkennettiin valittujen tutkimusprojektin osioiden pohjalta (Liite 1).

Kaikki haastateltavat painottivat käyttöliittymän selkeyttä, asiakaslähtöistä terminologiaa sekä ohjelmistossa navigoinnin helppoutta.

Tutkimuksen tekniseen toteutukseen liittyen haastateltiin projektiasiantuntijaa. Tämän lisäksi kysyttiin yleisesti projektiasiantuntijaryhmältä kommentteja nykyisten työkalujen toimivuudesta.

Projektiasiantuntijat kertoivat nykyisen mallin erilaisten lomakkeiden luomiseksi olevan raskas käyttää. He toivoivat enemmän keskitettyjä työkaluja erilaisten lomakkeiden hallintaan. Esimerkiksi tutkimuksissa joissa käytössä on sekä internetlomake että paperilomake, tulisi työkalujen tukea näiden toteuttamista yhden rakenteen pohjalta.

Projektiasiantuntijoille on nykyään raskasta hallita lomakkeella identtisiä tekstisisältöjä elementeittäin. Heidän näkemyksensä mukaan saman sisältöisten tekstien hallinta tulisi olla keskitettyä. Lomaketyökalun käyttöliittymien osalta projektiasiantuntijat toivoivat selkeää käyttöliittymää, jossa tarvittavat työkalut ovat aina saatavilla. Lomaketta luodessa on tarpeellista nähdä luotavasta lomakkeesta esikatselunäkymä samalla kun lomakkeen sisältöä määritetään.

Nykyisessä organisaationhallintatyökalussa he kokivat haasteelliseksi raahaustoiminnon, jonka avulla yksiköitä järjestellään. Puunäkymässä yksiköiden raahaaminen on työlästä ja virhealtista. Projektiasiantuntijat toivoivat työkaluihin yksiköiden järjestelyä nipuittain. Myöskään nykyinen muutoshistoria ei toimi toivotulla tavalla. Heidän mielestään muutoshistorian pitäisi olla kattavammin tallennettu. Työkalussa pitäisi kyetä kumoamaan helpommin tehtyjä toimenpiteitä.

Vastaajien hallinnan he kokivat hieman rajoitetuksi ja toivoivat siihen työkaluja vastaajien vapaampaan muokkaamiseen. Vastaajien hallinnan käyttöliittymän osalta olisi tarpeellista tehdä valintoja esimääritettyjen arvojen pohjalta. Nykyinen käsin syötettävien arvojen asettaminen on heidän mukaansa hyvin virhealtista.

Raportoinnin ja asiakastarpeiden kartoitusta varten haastateltiin tutkijoita. Heidän mielestään käyttöliittymässä projektilähtöiset näkymät palvelevat asiakasta paremmin kuin nykyisen ohjelmiston asiakkaan palvelut -näkökulma.

Tutkijat kertoivat tutkimusten lähtevän nykyisin enemmän ihmisten kuin organisaation näkökulmasta. Asiakkaat miettivät ensin ihmiset, ketkä osallistuvat tutkimukseen. Tämän jälkeen he jaottelevat ne rakenteeseen tulosten raportoinnin näkökulmasta.

Nykyisin asiakkailla on tarpeita yhdistää erilaisten mittausten tuloksia. Asiakkaat toivovat mahdollisuutta linkittää henkilöstötutkimuksen tuloksia esimerkiksi 360-arviointien tuloksiin.

Nykyisten teknisten haasteiden kartoitusta varten haastateltiin sovelluskehittäjiä. Sovelluskehittäjät kertoivat jatkokehittämisen olevan haasteellista nykyisen järjestelmän rakenteen vuoksi. Pientenkin muutosten tekeminen aiheuttaa usein virheitä joustamattoman rakenteen vuoksi. Muutoksia tehtäessä ei ole täysin selvää kuinka moneen paikkaan muutos voi vaikuttaa.

Nykyinen vaihejakomalli, jossa ominaisuus kehitetään kerralla valmiiksi annettujen määritysten pohjalta, ei ole toimiva. Ominaisuuden valmistuttua se voi muuttua hyvinkin paljon. Tästä johtuen ohjelmakoodi ei välttämättä enää palvele parhaalla mahdollisella tavalla haluttua lopputulosta. Toiveena olikin joustavampi määrittely ominaisuuksille. Toimeksiantajalla asiakastyötä tekevät henkilöt eivät ole sovelluskehitystaustaisia henkilöitä, joten mahdollisuus luoda ensin vain käyttöliittymän näkymä ominaisuutta varten, mahdollistaisi sen tarkemman määrittelyn ennen ohjelmakoodin kirjoitusta.

Toimeksiantajan käytössä olevan laatujärjestelmän vaikutusten huomioinnin kartoittamista varten haastateltiin yrityksen laatupäällikköä. Laatupäällikön mukaan laatujärjestelmän vaatimien dokumenttien hallinta ohjelmistossa ei ole mielekästä. Paremmin toimiva malli olisi tallentaa dokumentit dokumentinhallintajärjestelmään siten, että tarvittavat tiedostot olisi saatavilla ohjelmistosta esitäytettynä. Nämä dokumentit voitaisiin tallentaa sen jälkeen dokumentinhallintajärjestelmään.

Kyselylomakkeen hyväksyntä asiakkaalla tulisi tehdä lopullista lomaketta vasten. Hyväksynnän jälkeen tehtävät muutokset saisivat olla lähinnä kosmeettisia sekä sisältöä rajaavia. Kerätyn datan muutoksista ja validoinnista pitää laatujärjestelmän mukaan jäädä erittäin tarkka dokumentointi. Tämä olisi järkevää tallentua dokumentinhallintajärjestelmään samalla tavalla kuin muutkin dokumentit.

Laatujärjestelmän vaatimien ohjetekstien ja ohjelmiston tuottaman grafiikan selitteet olisi hyvä saada keskitetysti ohjelmiston käyttöliittymään. Tällä vältettäisiin selittämästä analyysin tarkoitusta useassa eri paikassa ohjelmiston sisällä.

Toimeksiantajan visioista tarpeellisten raportointi- sekä kyselytekniikoiden osalta keskusteltiin yrityksen toimitusjohtajan kanssa. Hän toivoi mahdollisuutta luoda erilaisia raportointi- ja kyselytyökaluja. Ohjelmiston tulisi tukea uusien tekniikoiden joustavaa käyttöönottoa. Näiden lisäksi toimitusjohtaja painotti käyttöliittymän yhteneväisyyttä toimeksiantajan nykyisten ohjelmistojen kanssa.

3.2.2 Yleiset rajoitukset ja mahdolliset lisätoiminnot

Työssä rajattiin raportointivaihetta hyvin paljon. Kuitenkin pidettiin mahdollisena sitä, että raportointi on yksi eniten jatkokehitystä tarvitsevista vaiheista. Haastatteluissa todettiin, että tiedonkeruun räätälöinnin lisäksi asiakkailla on hyvinkin paljon tarpeita juuri raportoinnin räätälöimiseksi. Raportoinnissa asiakkaat haluavat enemmissä määrin saada heidän liiketoimintaansa tukevia analyysejä.

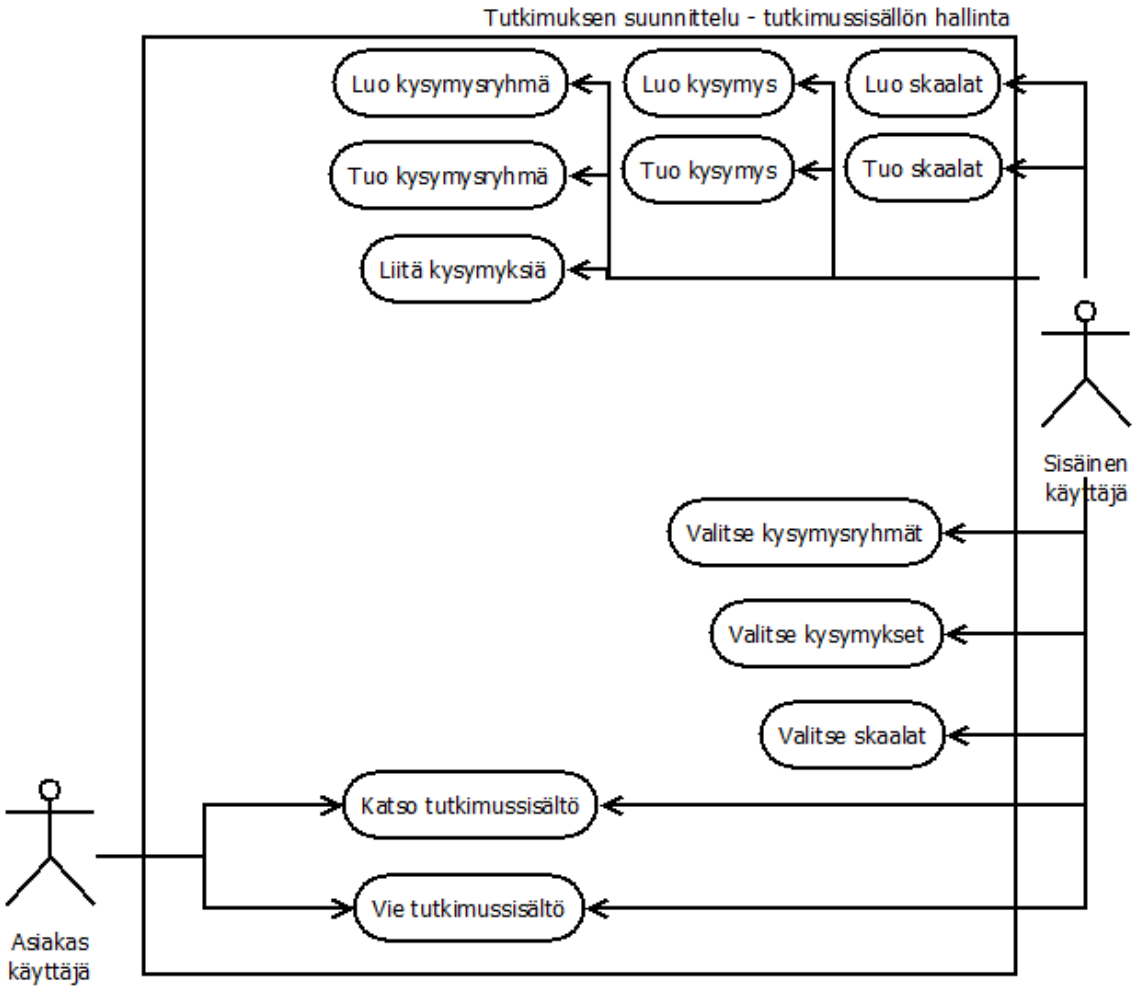
3.2.3 Käyttötapauskaaviot

Käyttötapausmallinnuksella kuvataan mitä ohjelmiston tulisi tehdä. Se on Ivar Jacobsonin kehittämä tekniikka. Siinä ei ole tarkoitus kuvata miten ohjelmisto toteuttaa kuvatut toiminnallisuudet. Käyttötapausmallinnuksen tärkeimmät osat ovat käyttötapaukset, toimijat sekä toiminto. Jokainen käyttötapaus kuvaa yhden täydellisen toiminnon. (Järvelä & Puusaari 2005, 9.)

Käyttötapausmallissa määritellään toiminto ja etsitään toimijat sekä käyttötapaukset. Käyttötapaukset kuvataan ja niiden väliset suhteet määritellään. Sillä kuvataan toiminnon ja toimijoiden välisiä suhteita. (Järvelä & Puusaari 2005, 9.)

Toimijaa kuvataan yleensä standardoidulla tikku-ukolla. Se voi olla esimerkiksi ohjelmiston käyttäjä, toinen järjestelmä tai jokin ulkoinen laite. Käyttötapaus kuvataan ellipsinä. Ne kuvaavat yhtä ohjelmiston tapahtumaa ja niiden yhteydessä tulee olla tapauksen nimi. (Järvelä & Puusaari 2005, 11 – 12.)

Käyttötapauskaaviot tehtiin UML-mallinnuskielellä. Niissä kuvattiin tärkeimmät toiminnallisuudet käyttäjänäkökulmittain. Jokaisesta ohjelmiston osiosta tehtiin käyttötapauskaavio (Kuva 1.).



Kuva 1. Esimerkki käyttötapauskaaviosta.

Käyttötapauskaavion yhteyteen lisätään usein sanallinen kuvaus käyttötapauksesta. Sanallisessa kuvaksessa määritetään ehdot, jolloin käyttötapaus voi toteutua. Tämän lisäksi siinä kuvataan käyttötapaus, sen poikkeukset sekä käyttötapauksen lopputulos. (Järvelä & Puusaari 2005, 15.)

Työssä käyttötapauskaavioiden yhteyteen tehtiin sanallinen kuvaus jokaisesta käyttötapauksesta (Taulukko 4.).

Taulukko 4. Esimerkki käyttötapauskaavion sanallisesta kuvauksesta.

KÄYTTÖTAPAUS:	Tutkimuksen suunnittelu – Tutkimussisällön hallinta
YHTEENVETO:	Valitaan / Luodaan tutkimukselle sisältö.
EHDOT:	Tutkimusprojektin tulee olla luotuna ohjelmistossa. Tutkimuksen kysymykset ovat sovittuna asiakkaan kanssa.
KUVAUS:	Sisäinen käyttäjä määrittelee tutkimuksessa käytettävät kysymykset ja niiden vastausvaihtoehdot eli skaalat. Kysymykset ryhmitellään kysymysryhmiin. Kysymykset, skaalat ja kysymysryhmät voidaan luoda uusina tai käyttää aiemmin luotuja kysymyksiä (ns. PoolQuestion). Pool-kysymykset, -skaalat sekä -kysymysryhmät voidaan myös tuoda nykyisestä ohjelmistosta. Asiakaskäyttäjällä on mahdollisuus tarkastella tutkimussisältöä koko tutkimusprojektin ajan.
POIKKEUKSET:	
LOPPUTULOS:	Tutkimusprojektin sisällöllinen rakenne, jota voidaan käyttää pohjana luotaessa lomaketta tai raportoinnille rakennetta.

3.3 Ei-toiminnalliset vaatimukset

Työssä ei-toiminnalliset vaatimukset luokiteltiin luokkiin; käytettävyys, tietoturva, ylläpidettävyys ja huollettavuus, siirrettävyys, laajennettavuus ja uudelleenkäytettävyys sekä konfiguroitavuus.

3.3.1 Käytettävyys

Työssä määritettiin, että käyttöliittymän tulee olla selkeä ja moderni. Tällä saavutetaan parempi käyttäjäkokemus ohjelmiston helppoudesta. Suunnittelussa tulee noudattaa samansuuntaisia linjoja toimeksiantajan nykyisen ohjelmiston kanssa, jotta aiempaan ohjelmistoon tottuneiden käyttäjien olisi helppo omaksua uuden ohjelmiston käyttö.

Käyttöliittymän rakenteen tulee tukea tutkimusprosessin läpivientiä. Käytännössä toiminnallisuudet sijoitellaan siten, että niiden eteneminen tukee normaalia prosessin kulkua.

Ohjelmistossa tulee olla sisäänrakennettuna näkymiä tukevia ohjetekstejä. Nämä voivat olla esimerkiksi upotettuna näkymään tai selkeästi avattavissa ohjelmistosta. Käyttäjien toimintaa tulee tukea tiedon syöttämisen eri vaiheissa. Virheellisistä syötteistä indikoidaan ohjaavilla virheilmoituksilla ja tarvittavat syötekentät kuvataan riittävällä tarkkuudella, jotta ohjelmistoa käyttävä käyttäjä voi helposti havaita mitä kyseessä olevaan kenttään odotetaan.

Ohjelmiston tulee olla kieliversioitavissa siten, että ennalta määritetyt kielet ovat valittavissa käyttöliittymäkieliksi tai kattavammin tiedonkeruu- sekä raportointikieliksi.

3.3.2 Tietoturva

Käyttäjien ja palvelimen välinen verkkoliikenne ohjelmistoon toteutetaan TLS-salauksella käyttäen HTTPS-protokollaa. Käyttäjän kirjautuessa ohjelmistoon, salasana kryptataan käyttäjän laitteella ennen sen lähettämistä verkon yli. Tietokannassa säilytettävät käyttäjien salasanat ovat tallennettuna vain salatussa muodossa.

Ohjelmiston eri osioiden välillä siirryttäessä tarkistetaan käyttäjän voimassa oleva kirjautuminen. Käyttäjän istuntokohtaiset tiedot säilytetään selaimen istunnossa. Näitä ovat esimerkiksi käyttäjätiedot, valinnat sekä välimuistiin tallennetut tiedot. Käyttäjäoikeuksia voidaan rajoittaa ohjelmistossa eri moduuleihin käyttäjäryhmien avulla.

3.3.3 Ylläpidettävyys ja huollettavuus

Ohjelmaa kehitetään jatkuvasti ja siitä julkaistaan ennalta määrätyin välein uusi versio. Tällä varmistetaan uusien asiakastarpeiden huomioiminen ohjelmistossa. Edelliset ohjelmaversiot varmuuskopioidaan aina uuden version julkaisun yhteydessä. Mikäli uudessa versiossa havaitaan kriittisiä puutteita, voidaan varmuuskopiosta palauttaa nopeasti edellinen toimiva ohjelmaversio.

Ohjelmakoodin muutoksia seurataan paketinhallintajärjestelmällä. Sen avulla voidaan seurata miten ohjelmakoodi on muuttunut. Tämän lisäksi voidaan seurata milloin muutokset on tehty ja kuka on tehnyt muutokset. Paketinhallintajärjestelmä on usein osa sovelluksen elinkaaren hallintaan tarkoitettua ohjelmistoa.

Tietokanta varmuuskopioidaan toimeksiantajan määrittämän palautumisaikatavoitteen (RTO) sekä palautumispistetavoitteen (RPO) mukaisesti.

3.3.4 Siirrettävyys, laajennettavuus ja uudelleenkäytettävyys

Ohjelmiston suunnittelussa huomiotiin sen laajennustarpeet siten, että ohjelmiston osia on helppo jatkokehittää. Ohjelmistosta suunniteltiin modulaarinen, jolloin eri moduulien lisääminen ja poistaminen on mahdollista.

Mikäli ohjelmiston kehityksessä tullaan käyttämään kolmannen osapuolen komponentteja, hyödynnetään niitä siten, että niiden vaihtaminen myöhemmin ohjelmiston elinkaaren aikana on mahdollista. Tällä varmistetaan ohjelmiston luotettavuus tilanteessa, jossa ohjelmistossa käytetyn komponentin tuki loppuu tai se havaitaan muusta syystä sopimattomaksi ohjelmistoon.

Ohjelmiston käyttöönoton jälkeen kehitystä tullaan suuntaamaan siten, että alkuun rinnalla käytettävä toimeksiantajan nykyinen ohjelmisto tulee todennäköisesti elinkaarensa päähän ennen uutta ohjelmistoa.

3.3.5 Konfiguroitavuus

Ohjelmiston elinkaaren aikana saattaa esiintyä asiakastarpeita, jotka lähtökohtaisesti ovat kertaluonteisia. Näille ei todennäköisesti ole järkevää tehdä projektitason määritystä, vaan ne voidaan toteuttaa objektitasoisina parametreina. Parametroinnilla voidaan ohjata tietyn toiminnon, esimerkiksi analyysilaskennan, toimintaa. Nämä helpottavat käyttöä tilanteessa, jossa poikkeavasta toimintatavasta seuraisi valinnan tekeminen jokaisessa tutkimusprojektissa. Esimerkiksi parametri, joka skaalaa indeksin eri skaala-asteikolle tietyn asiakkaan tutkimuksissa. Tällöin parametrilla ohjattaisiin analyysinlaskentaa vain jokaisessa kyseessä olevan asiakkaan tutkimuksessa.

Lähes jokaisessa projektissa tehtävät valinnat upotetaan näkymiin siten, että niiden hallinta on osa luonnollista projektin läpivientiä.

3.3.6 Suorituskyky

Tutkimuksissa on havaittu, että ohjelmiston latausajat vaikuttavat suuresti käyttäjäkokemukseen. 100 ms on aika, jossa aivot eivät ehdi rekisteröidä odotusaikaa. Googlen tutkimuksen mukaan 400 ms on raja-arvo, jonka jälkeen käyttäjät alkavat kokea ohjelmiston hitaaksi. Yli viiden sekunnin latausajat riittävät karkottamaan valtaosan käyttäjistä ja luovat tunteen ohjelmiston hitaudesta. (Gavalda 2015.)

Työssä määritetyn ohjelmiston suorituskyvyn tavoiterajaksi asetettiin yksi sekunti. Ohjelmoinnissa pyritään siihen, ettei yksittäisen komennon suorittamiseen menisi kauempaa aikaa.

Tästä suljetaan pois kuitenkin yksittäiset isommat kertaluonteiset toiminnallisuudet, kuten suurten raporttien generoimiset, laajat analyysit sekä muut vastaavat ajot.

3.4 Käyttöliittymät

Käyttöliittymäsuunnittelun pohjana käytettiin aiemmin luotuja käyttötapauskaavioita, joista voitiin havaita tarpeelliset toiminnot näkymittäin.

Käyttöliittymät suunniteltiin tukemaan useita eri laitealustoja pöytätietokoneista älypuhelimiin. Tämä huomioitiin käytettävyyttä suunnitellessa siten, että toiminnot sijoitellaan näkymiin selkeästi erilleen toisistaan helpottamaan käyttöä mobiililaitteilla. Lisäksi mobiilitukea pyrittiin parantamaan huomioimalla suunnittelussa näkymien responsiivisuus.

Responsiivisuudella tarkoitetaan tilannetta, jossa näkymän elementit sijoitellaan näytölle siten, että sama näkymä tukee kaiken kokoisia laitteita. Responsiivisuus voidaan toteuttaa käyttämällä HTML5:n sekä CSS3:n uusia tekniikoita. (Leiniö 2012.)

Suunnitelmissa on huomioitu Nielsenin säännöt sekä Fittsin laki käyttöliittymien käytettävyyden suunnitteluun.

Nielsenin säännöt on luonut tohtori Jacob Nielsen. Hän on yksi käytettävyyden edistäjistä. Käyttöliittymän yksinkertaisuus on tärkeää, koska jokainen toiminto on uusi asia opeteltavaksi, uusi mahdollisuus väärin ymmärryksille tai yksi asia huomioitavaksi etsittäessä toimintoa näkymästä. (Nielsen 1995a.)

Käyttöliittymässä kielen tulisi olla käyttäjien äidinkieltä sekä ammattitermistöä. Metaforat, esimerkiksi roskakori poista-toiminnon yhteydessä helpottavat käyttäjää nopeasti tunnistamaan toimintoja. Käyttäjän ei pitäisi joutua muistamaan tarvittavia asioita vaan ne tulisi esittää järkevästi näkymissä. Muistamista voidaan vähentää esimerkiksi valintalistojen käytöllä tai tarvittavien syötteiden esimerkeillä. (Nielsen 1995a.)

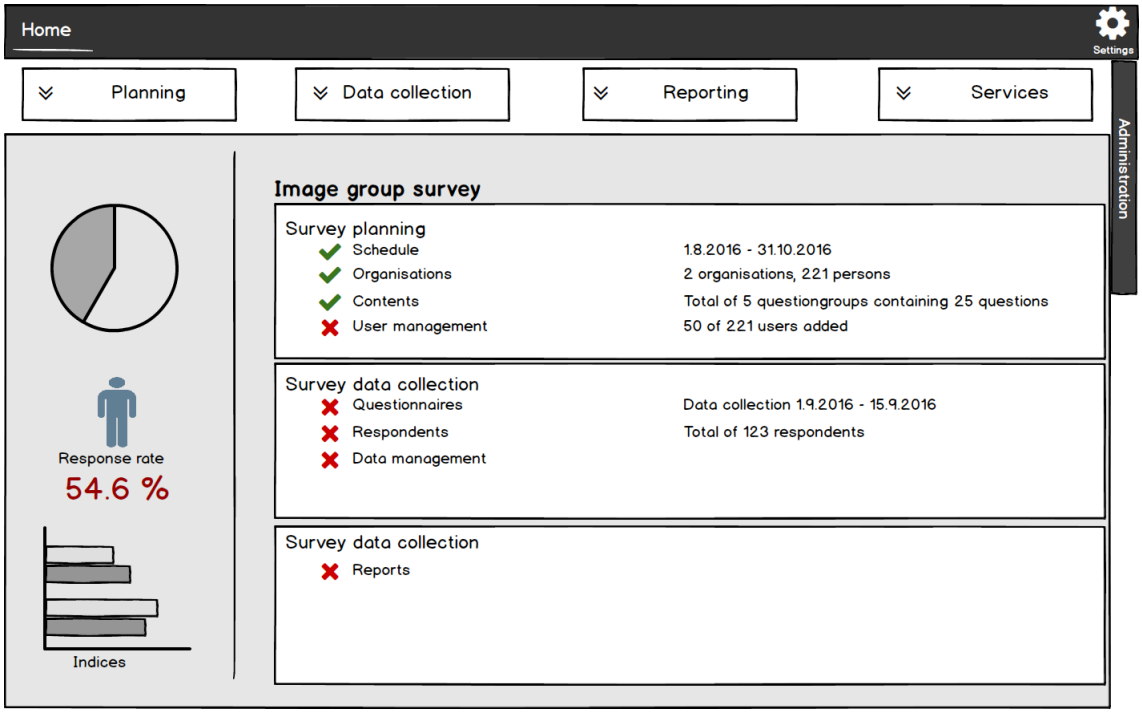
Jokaisen näkymän tulisi olla yhdenmukainen muiden näkymien kanssa. Vastaavat toiminnot tulisi aina löytyä samasta paikasta kaikissa näkymissä. Käyttäjälle tulee myös antaa riittävästi palautetta tehdyistä toimista. Esimerkiksi onnistuiko tallentaminen. (Auer 2009.)

Virhetilanteiden estämiseksi käyttäjälle voidaan esittää oletusarvoja sekä pyytää varmistusta kun tehdään peruuttamattomia toimintoja. Käyttöohjeet tulee rakentaa selkeästi ja sisältötietoisiksi. Ohjetta avatessa sen tulisi kohdistua suoraan oikeaa aihetta käsittelevään tietoon. Kuitenkaan pelkästään käyttöohjeisiin ei voi luottaa vaan käyttöliittymän tulisi olla riittävän selkeä siten, että ohjeisiin tulee nojata vain erikoisissa tapauksissa. (Auer 2009.)

Fittsin laki pohjautuu vuonna 1949 julkaistuun teoreemaan viestintäjärjestelmistä. Se on ihmisen psykomotorisen käyttäytymisen malli. Fittsin lailla lasketaan aika, joka vaaditaan kohteen saavuttamiseen nopealla tähdätyllä liikkeellä. (Häkkinen 2014.)

Lakia soveltamalla on huomattu, että näytöltä löytyy viisi pistettä, jotka ovat ylivoimaisesti parhaiten saavutettavissa kohdistimella. Nämä ovat näytön neljä kulmaa sekä kohdistimen sen hetkinen sijainti. (Pietilä 2006.)

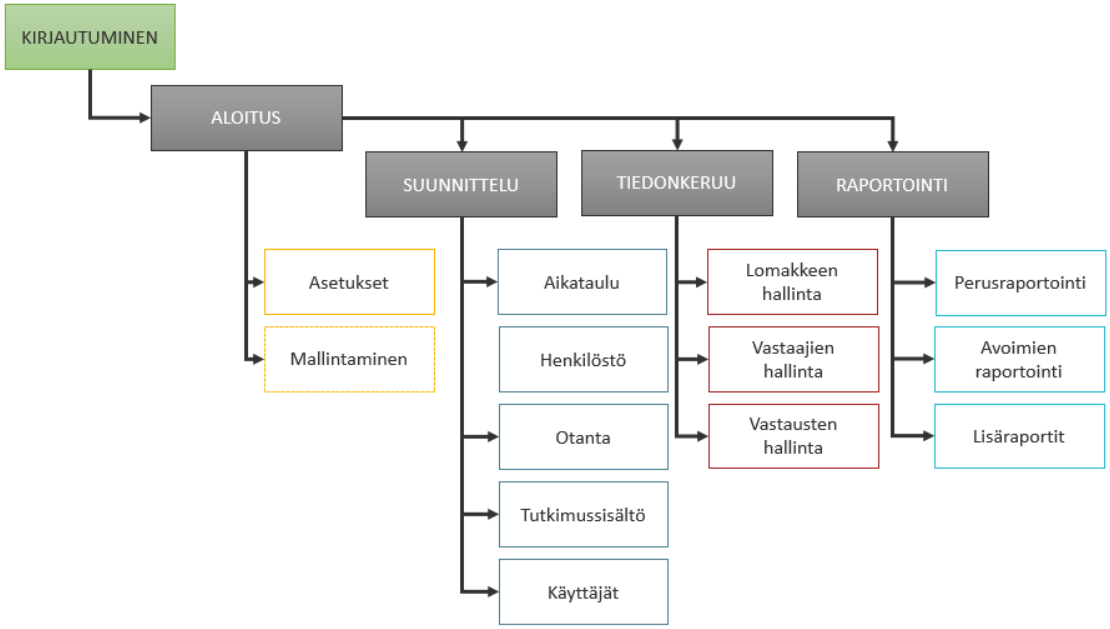
Käyttöliittymän näkymistä piirrettiin mallikuvat (Kuva 2.), joissa eri näkymien toiminnallisuudet on hahmoteltu paikoilleen.



Kuva 2. Esimerkki aloitussivun mallikuvasta.

Käyttöliittymän näkymien mallikuvia testattiin toimeksiantajan henkilökunnalla ja niihin tehtiin tarvittavia muutoksia käyttäjäpalautteen pohjalta.

Näkymien väliseen siirtymiseen suunniteltiin käyttöliittymäkartta (Kuva 3.), jonka avulla eri näkymien suhteet ja navigaatio voidaan hahmotella.



Kuva 3. Esimerkki käyttöliittymäkartasta.

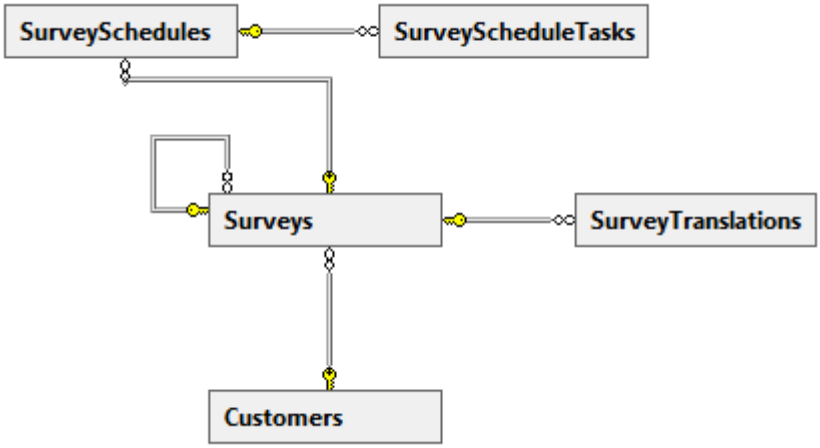
3.5 Tiedot ja tietokannat

Työssä tiedot ja tietokannat luokiteltiin seuraaviin luokkiin; tietokanta, tietokannan taulut, XML-mallit sekä analyysien rakenne.

3.5.1 Tietokanta

Tietokannan suunnittelun pohjana käytettiin aiemmin luotuja käyttötapauskaavioita sekä käyttöliittymän näkymien mallikuvia. Niiden avulla kartoitettiin mitä tietoa pitäisi tallentaa tietokantaan.

Relaatiomallissa tietokannan taulujen väliset relaatiot on kuvattu pääavain – viiteavain suhteina (Kuva 4.). Niillä kuvataan miten eri taulut ovat suhteessa keskenään. Suhteet voivat olla yhden suhde yhteen, yhden suhde moneen tai monen suhde moneen. (Laine H. 2005.)



Kuva 4. Esimerkki aikataulutoiminnon tietokantataulujen relaatiomallista.

Relaatiomallin avulla tietokanta normalisoitiin suunnitteluvaiheessa. Normalisoinnilla voidaan tietokannasta poistaa redundanssia sekä epäyhtenäisiä riippuvuussuhteita. Redundantit vaikeuttavat tietokannan ylläpitoa ja lisäävät levytilan käyttöä. Sama tieto sijaitsee useammassa paikassa, jolloin sen muuttumisen yhteydessä se pitää muuttaa moneen paikkaan. (Microsoft 2008.)

Normalisoinnilla on kolme tasoa. Ensimmäisellä tasolla tietokannasta poistetaan yksittäisissä tauluissa toistuvat ryhmät luomalla niille erillinen taulu sekä merkitsemällä ryhmät pääavaimella. Toisella tasolla tietokantaan luodaan taulut tiedoille jotka sijaitsisivat useassa taulussa ja viitataan niihin viiteavaimilla. Kolmannella tasolla poistetaan kentät, jotka eivät ole riippuvaisia avaimesta. Kolmannen tason toteuttaminen ei ole aina käytännöllistä tilanteissa, joissa on odotettavissa vain pientä redundanssia. Esimerkiksi Asiakas-taulun osoiterivi, joka voi sisältää saman osoitteen monella rivillä. Se on kuitenkin niin harvinaista, ettei tietokantaa kannata normalisoida tämän vuoksi. On olemassa myös neljäs- ja viidestaso, mutta niitä sovelletaan harvoin tietokannan suunnittelussa. (Microsoft 2008.)

3.5.2 Tietokannan taulut

Tietokannan taulut mallinnettiin samaan aikaan kun mallinnettiin relaatiot. Esitystapana käytettiin UML-kielen luokkakaaviota (Taulukko 5.). Tietokannan taulu esitettiin luokkana ja sen kentät attribuutteina. (Järvelä & Puusaari 2005, 18 – 19.)

Taulukko 5. Esimerkki tietokannan aikataulurivitaulusta.

SurveyScheduleTasks	
SurveyScheduleTaskID	bigint
SurveyScheduleID_FK	bigint
PoolScheduleTaskID_FK	bigint NULL
PoolScheduleTaskTypeID_FK	bigint
ListingOrder	bigint
Task	nvarchar(255)
StartDate	datetime NULL
EndDate	datetime
SupplierResponsible	bit
CustomerResponsible	bit
Deleted	bit

3.5.3 XML-mallit

Ohjelmiston suunnittelussa tavoitteena oli suunnitella helposti jatkokehitettävä ohjelmisto. Tähän tavoitteeseen pyrittiin käyttämällä tietokannassa XML-tietotyyppiä. Sen avulla on mahdollista tallentaa hyvinkin monimutkaisia rakenteita tietokantaan yhteen tauluun. Tämä on hyödyllistä silloin kuin on odotettavissa, että rakenteeseen tulee usein muutoksia.

Mikäli tietokanta suunniteltaisiin normaalisti relaatioiden kautta, vaikeuttaisivat muutokset tietokannan ylläpidettävyyttä sekä aiheuttaisivat turhaa levytilan käyttöä tietokannassa.

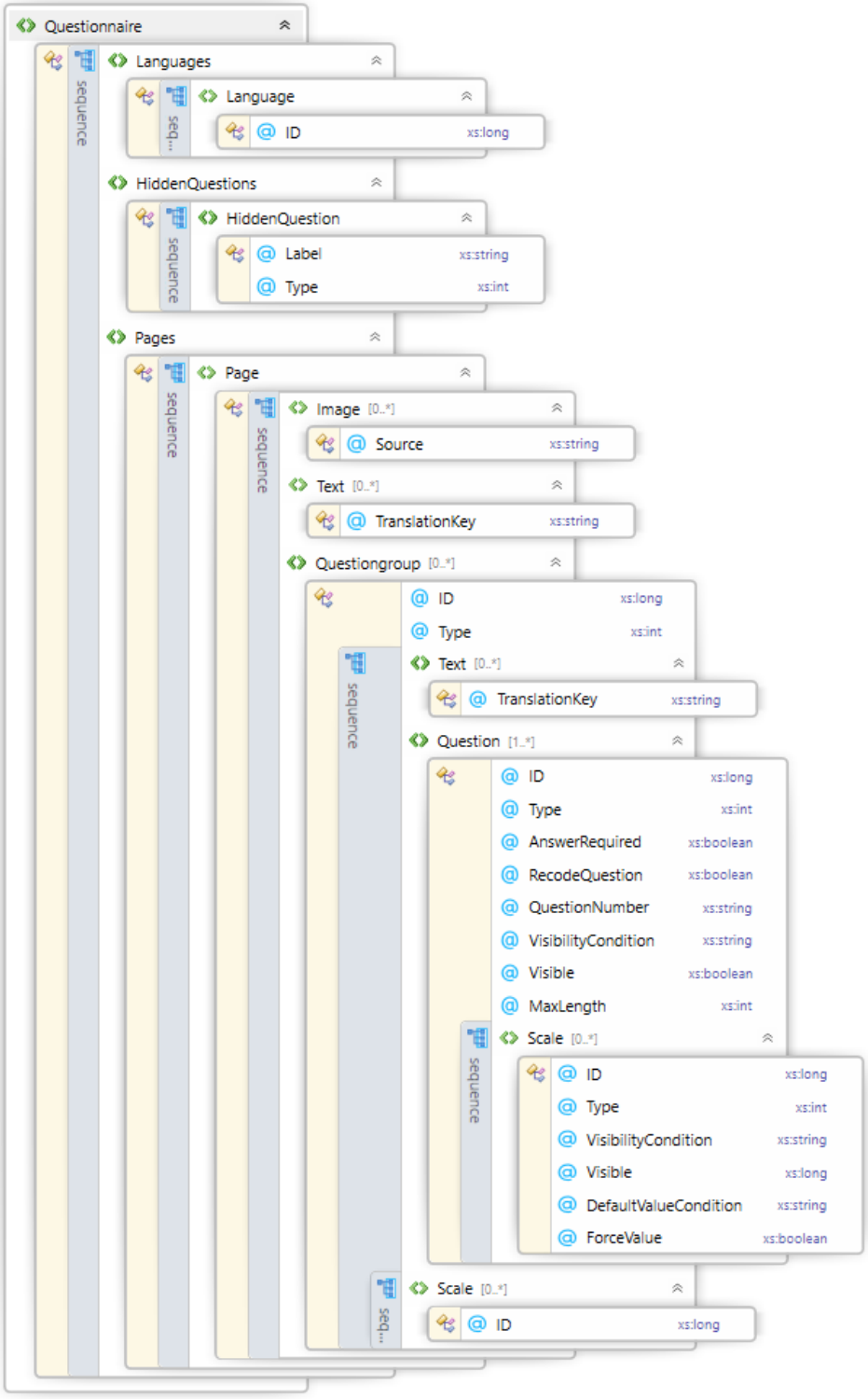
XML-tietotyypin kautta on mahdollista ennalta valittuja rakenteita muuttaa tai jatkokehittää ilman muutoksia tietokantaan.

Työssä valittiin tällaiseksi rakenteeksi esimerkiksi lomakkeen rakenne. Sen voidaan odottaa muuttuvan projektista toiseen. XML-tietotyyppiä käytettäessä voidaan lomakkeen rakenne kuvata XML-kielen avulla sen sijaan, että sitä varten tehtäisiin relaatiotietokantaan tarpeelliset taulut tiedon säilytystä varten (Taulukko 6.).

Taulukko 6. Esimerkki tietokannan lomakelinkkitaulusta.

SurveyQuestionnaireLinks	
SurveyQuestionnaireLinkID	bigint
SurveyQuestionnaireID_FK	bigint
Description	nvarchar(255)
Content	xml
StartDate	datetime
EndDate	datetime
Default	bit
Deleted	bit

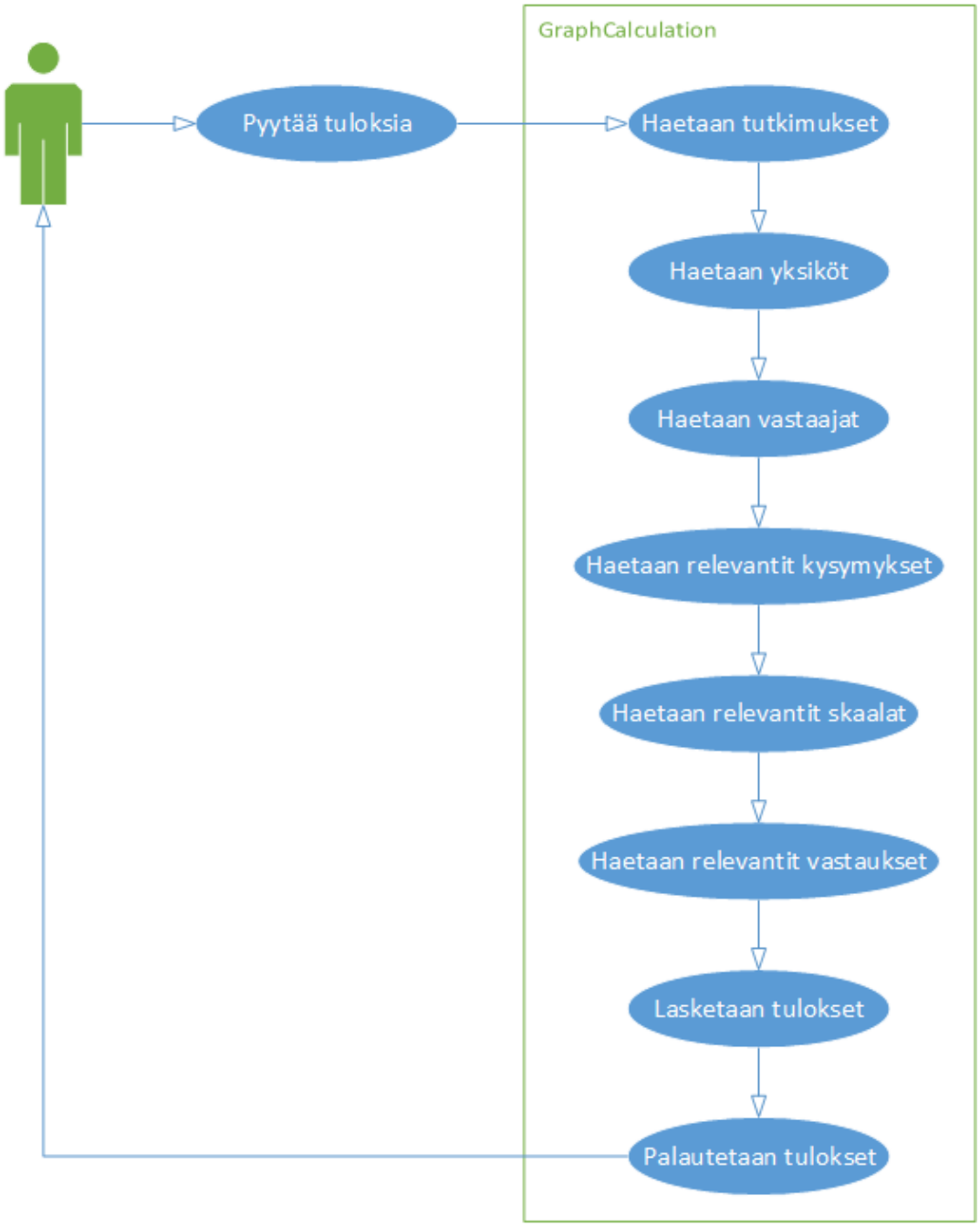
XML-dokumentin syntaksin varmentamista varten sen rakenne kuvattiin XML-skeeman avulla (Kuva 5.). Näitä luotuja XML-skeemoja voidaan käyttää tallennettavan tiedon validointiin ohjelmistokehitystyössä.



Kuva 5. Esimerkki lomakkeen XML-skeemasta.

3.5.4 Analyysien rakenne

Analyysien rakenne suunniteltiin modulaariseksi, jolloin sen eriosuuksien hallinta keskitetysti on helpompaa. Analyysin rakenne voidaan karkeasti jaotella seuraaviin ryhmiin; analyysin kohteen rajaus, datan poimiminen sekä lopuksi kerättyjen tietojen pohjalta analyysin laskenta (Kuva 6.).



Kuva 6. Analyysien rakenne.

Ohjelmalogiikasta kutsut eri analyyseihin toteutetaan samalaisella parametroinnilla, jotta kutsut voidaan lähettää tietokantaan keskitetysti.

Analyysin eri osuudet toteutetaan käyttämällä tietokannan proseduureja sekä funktioita. Niiden avulla tehdään keskitetyt työkalut esimerkiksi analyysiin liittyvien tutkimuskertojen rajaamiseen.

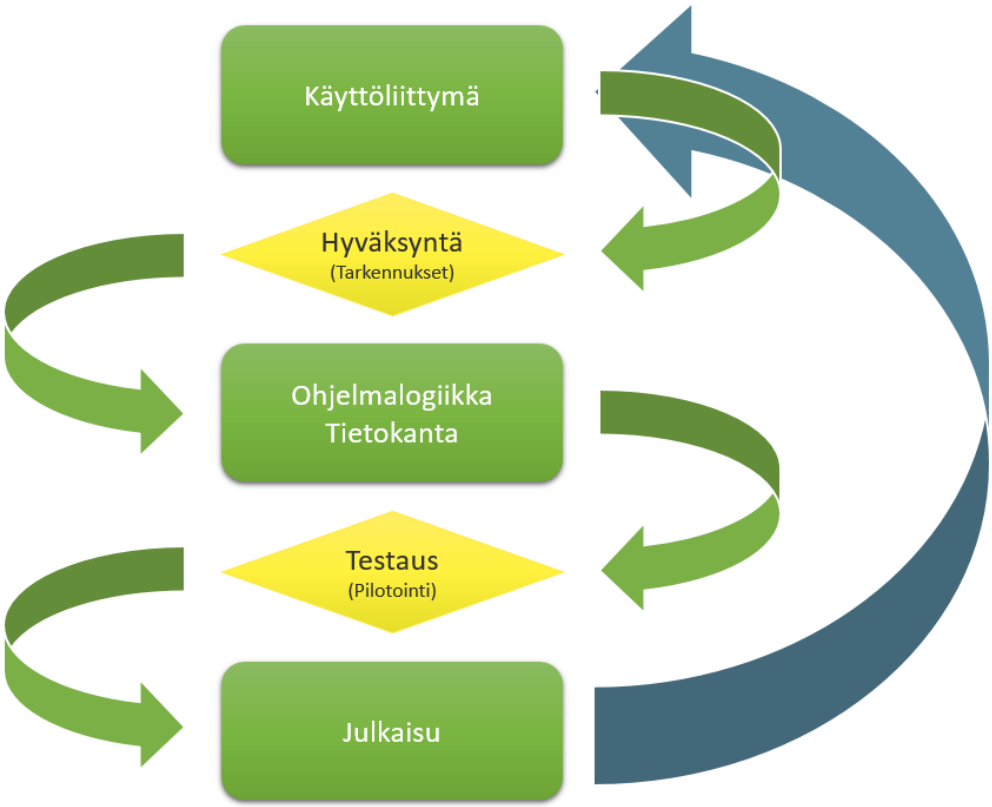
Kun analyysin kannalta tarpeellinen data on noudettu, lasketaan analyysikohtaisesti sen tulokset. Tulosjoukon tulee olla analyysista riippumatta samanlainen, jotta se voidaan ohjelmalogiikan puolella ottaa vastaan samanlaisen rakenteen avulla. Tällä saavutetaan helpommin ylläpidettävä esityslogiikka analyyseihin.

3.6 Toteutus

Työssä toteutusvaiheen suunnitelmat jaoteltiin seuraaviin luokkiin; vaihejakomalli, ohjelmistonarkkitehtuurikuvaus, ohjelmistokehys, lähdekoodin laatu, järjestelmäsuunnittelu, testaus, julkaisu sekä alustava aikataulu.

3.6.1 Vaihejakomalli

Ohjelmistokehityksen vaihejakomalliksi ajateltiin mallia, jossa kehitys etenee käyttöliittymäohjelmoinnin kautta toimintalogiikkaan (Kuva 7.). Tällä helpotetaan toimeksiantajan henkilöstön hahmottamista uusista toiminnoista. Usein pienet muutokset käyttöliittymässä voivat aiheuttaa suuria muutoksia toimintalogiikassa sekä tietokantakerroksessa.



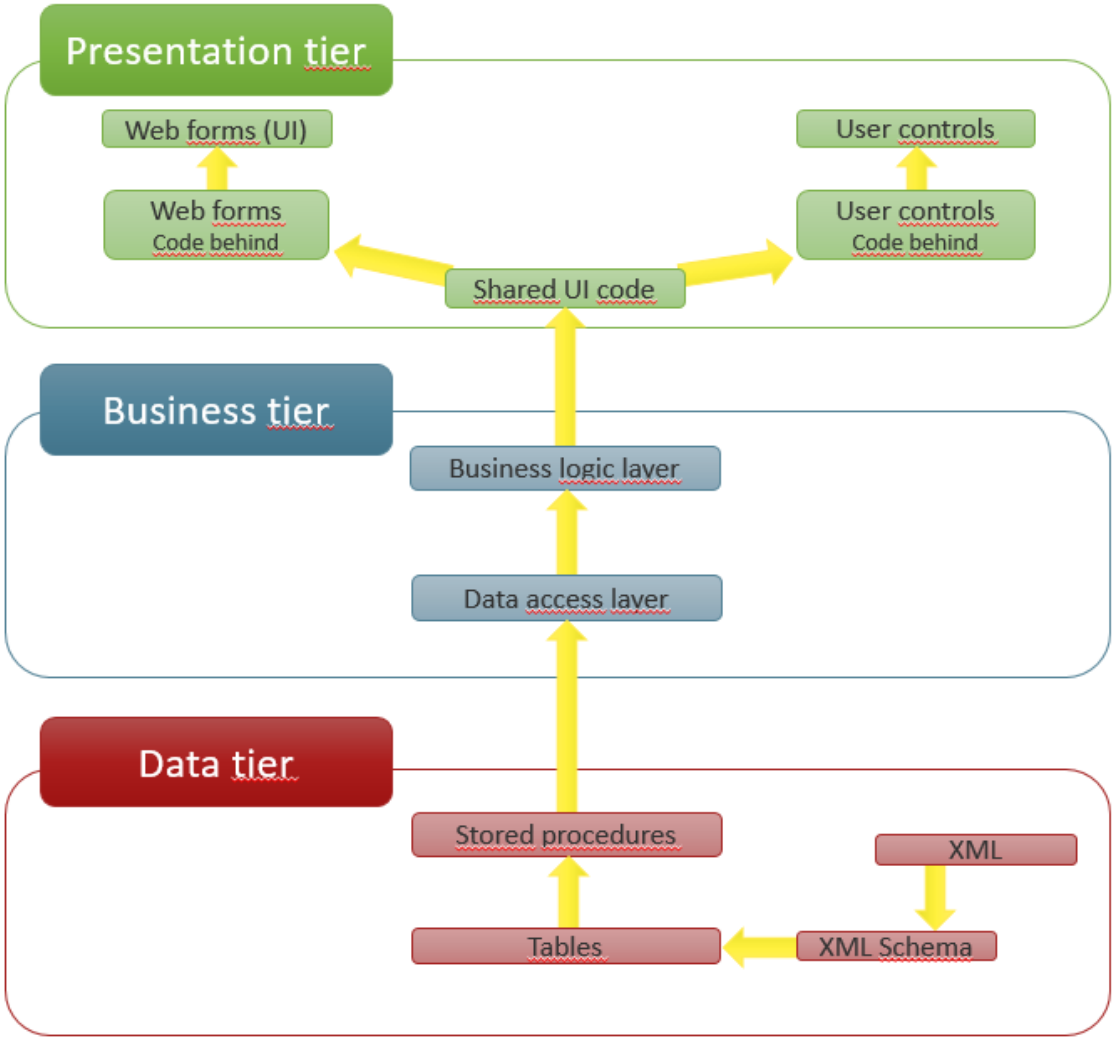
Kuva 7. Vaihejakomalli.

Ensin ohjelmoidaan käyttöliittymä joka hyväksytetään toimeksiantajan henkilöstön edustajalla, jolla on riittävä tieto uuden ominaisuuden tavoitteista. Hyväksynnän saamisen jälkeen voidaan aloittaa ohjelmalogiikan ohjelmoiminen.

Käyttöliittymän ja ohjelmalogiikan valmistuttua voidaan aloittaa ominaisuuden testaaminen. Kun testaaminen on saatu valmiiksi, voidaan ominaisuus julkaista osaksi ohjelmistoa asiakaskäyttäjien käytettäväksi.

3.6.2 Ohjelmiston arkkitehtuurikuvaus

Ohjelmiston arkkitehtuuriksi valittiin kolmikerrosarkkitehtuuri (Kuva 8.). Kolmikerrosarkkitehtuuri perustuu siihen, että kokonainen kerros voidaan vaihtaa vaikuttamatta muihin kerroksiin. Arkkitehtuuri tuo myös selkeyttä ohjelmalogiikkaan. (Mains, B. 2008.)



Kuva 8. Ohjelmiston arkkitehtuurikuvaus.

Esityskerroksessa luodaan käyttöliittymän näkymät ja niiden sisältö. Esityslogiikassa ei tyypillisesti ole lainkaan ohjelmakoodia, vaan kaikki käyttäjän antamat arvot ja tapahtumat välitetään taustalla olevalle taustakoodille. (Mains, B. 2008.)

Kaikki ohjelmalliset toiminnot kuten, arvojen laskemiset, tapahtumien käsittelyt sekä kutsut tietokantaan toteutetaan liiketoimintakerroksessa. Liiketoimintakerros koostuu usein

tietokannan tauluja vastaavista oliosta ja niiden metodeista. Tietokantakerroksessa suoritetaan kutsut tietokantaan ja palautetaan arvot liiketoimintakerrokselle. (Mains, B. 2008.)

3.6.3 Ohjelmistokehys

Työssä ohjelmistokehykseksi valittiin toimeksiantajan määrityksistä ja olemassa olevasta ympäristöstä johtuen Microsoftin .NET 4.6 -alusta. Tämä valinta määritti käytettäväksi käyttöliittymäkieleksi ASP.NET:n sekä ohjelmointikieleksi C# 6.0 -version.

Tietokantaliittymä ohjelmasta tietokantaan tullaan tekemään käyttäen Microsoft ADO.NET -kirjastoa. Se soveltuu erinomaisesti kolmikerrosarkkitehtuuriin ja tarjoaa kattavan kirjaston tietokantakutsujen luontiin. Tietokannassa kyselyt toteutetaan proseduureilla käyttäen T-SQL-kieltä.

Ohjelmointiympäristönä toimeksiantajalla on Microsoftin Visual Studio 2015. Paketinhallintajärjestelmänä toimeksiantajalla on käytössään Microsoftin Team Foundation Server. Molemmat sopivat tämän projektin työkaluiksi hyvin.

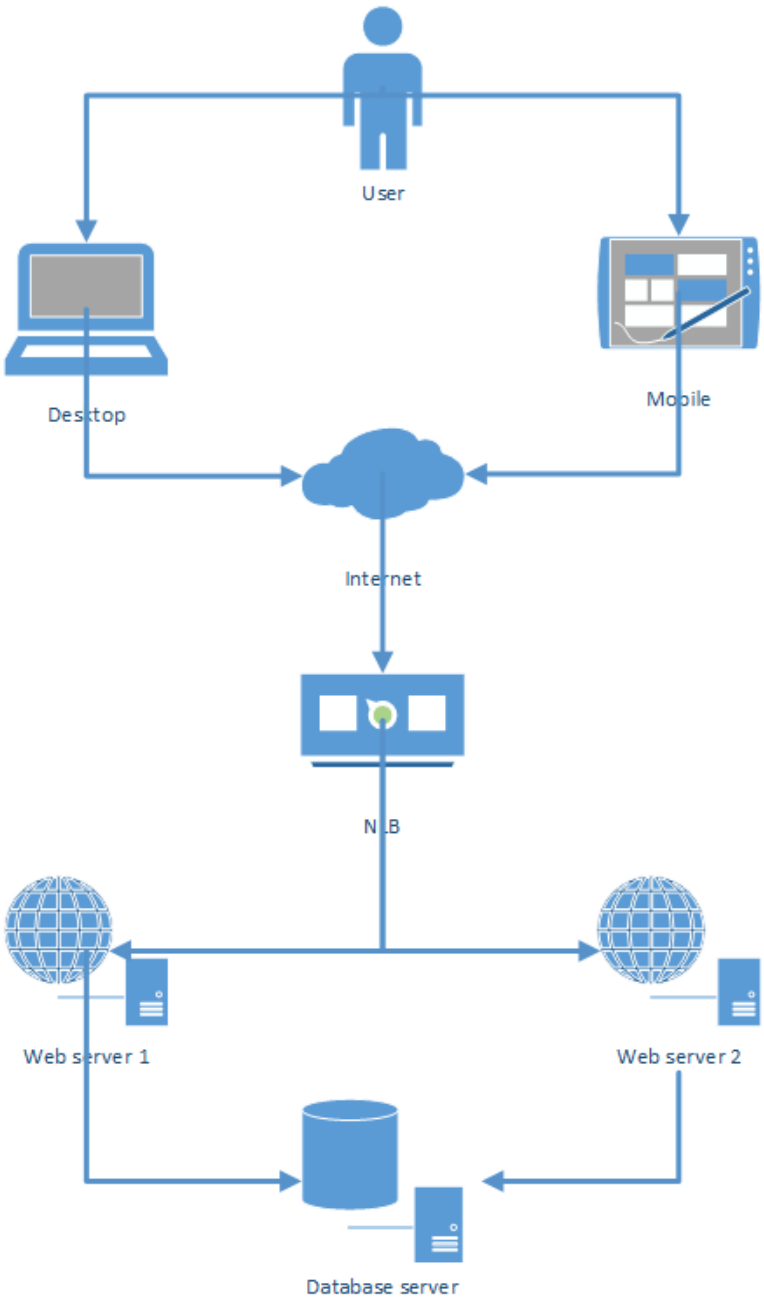
3.6.4 Lähdekoodin laatu

Lähdekoodin kirjoituksessa huomioidaan yleiset ohjelmointikielen kirjoitusohjeet ja dokumentoidaan Microsoftin .NET -alustan XML-dokumentointia hyödyntäen. Dokumentoinnilla saavutetaan parempi jatkokehitettävyys ja nopeampi perehdyttäminen ohjelmistoa ohjelmoivan henkilöstön vaihtuessa.

Lähdekoodi moduloidaan ja tiedot moduulien välillä kuljetetaan selaimen istunnossa. Modulointi helpottaa ohjelmiston osien itsenäistä jatkokehitystä ja edistää virheiden hallintaa ja testausta.

3.6.5 Järjestelmäsuunnittelu

Ohjelmiston tarjoavat järjestelmät toteutetaan siten, että loppukäyttäjälle tarjotaan työpöytäympäristön sekä mobiiliympäristön kanssa yhteensopivat käyttöliittymät (Kuva 9.).



Kuva 9. Järjestelmäsuunnittelu.

Palvelu tuotetaan kahdella verkkopalvelimelta, käyttäen Microsoftin NLB-palvelua kuorman tasaamiseen palvelinten välillä. NLB-palvelu mahdollistaa myös joustavammat ylläpito mahdollisuudet ohjaamalla käyttäjät tarvittaessa vain yhdelle palvelua tarjoavalle palvelimelle. Tiedot säilytetään tietokannassa, johon molemmilla verkkopalvelimilla on yhteys. Tietokantapalvelin ei ole yhteydessä julkiseen verkkoon.

3.6.6 Testaus

Käyttöliittymien käytettävyyden testaamisessa sovelletaan Nielsenin heuristiikkaa. Käyttöliittymien kieliversioinnissa käytetään pseudolokalisointia sekä kuormankestävyyttä testataan käyttämällä ulkoisia kuormankestävyyden testaamiseen tarkoitettuja testityökaluja.

Käyttöliittymä testauksessa pyydetään ennalta valittuja aloittelevia käyttäjiä käyttämään ohjelmistoa ja pohtimaan sen käyttöliittymää annetun muistilistan (Liite 2.) pohjalta. Saadut havainnot arvioidaan ja niille asetetaan vakavuusluokka muutaman henkilön toimesta. Arvioivien henkilöiden tulisi olla asiatuntijoita liittyen arvioitaviin käyttöliittymiin. Lopuksi annettujen vakavuusluokkien keskiarvot kertovat kuinka vakavasta ongelmasta on kysymys. (Nielsen 1995b.)

Pseudolokalisointi on tekniikka, jolla väärennetään ohjelmiston käännöstekstit. Sen avulla on mahdollista löytää heikkoudet ja virheet ohjelmiston lokalisoinnissa. Lokalisoinnissa yleisimmät haasteet ovat kovakoodatut tekstit tai tekstien yhdistely. Ohjelmoija on saattanut kirjoittaa käyttöliittymän tekstisisällöt ohjelmaan koodissa, jolloin ne eivät käänny valitun lokalisaation myötä tai tekstejä yhdistetään koodissa, jolloin ne eivät välttämättä ole kieliopillisesti oikein jokaisella tuetulla kielellä. Käyttöliittymässä ei välttämättä myöskään ole varattu riittävästi tilaa eripituisille käännöksille tai käyttöliittymä ei tue oikealta vasemmalle kirjoitettuja tekstejä. (Tamplin 2011.)

Pseudolokalisoinnissa käännökset korvataan generoidulla tekstillä, johon sisällytetään erikoismerkkejä sekä niitä venytetään alkuperäistä käännöstä pidemmiksi. Tällä havaitaan ongelmat tekstien pituudessa sekä erikoismerkkien tuessa. Oikealta vasemmalle kirjoitettuja tekstejä testatessa alkuperäinen käännös korvataan Unicode-merkistöllä, jolloin nähdään miten tekstit sijoittuvat näkymään. (Tamplin 2011.)

Ohjelmiston ohjelmalogiikkaa testataan erilaisilla syötteillä ja niiden antamaa ulostuloa arvioidaan annettujen vaatimusten valossa. Testaustekniikkana toimii Black box — White box -testaus.

Black box -mallissa testataan ohjelmiston ulostulon käyttäytymistä erilaisilla syötteillä. Siinä testaaja ei tunne eikä näe ohjelmiston rakennetta tai sisältöä. Saatuja ulostulon arvoja verrataan odotettuihin tuloksiin. White box -mallissa ohjelmiston testaajalla on pääsy ohjelmakoodiin ja tuntemus ohjelman rakenteesta. Toisin kuin black box -mallissa, white box -mallissa pyritään suorittamaan testit siten, että jokainen ohjelmakoodin käsky tulee vähintään kerran suoritetuksi, kaikki ohjelmakoodin haarat tulevat testatuksi sekä kaikki ehtolausekkeet saavat jokaisen mahdollisen arvon. (Kautto 1996.)

Toimeksiantajan henkilökunta suorittaa black box -mallin mukaisen testauksen ohjelmakehityksen yhteydessä. Heitä varten on luotu tarkastuslista (Liite 3.) avustamaan testaamisessa. Ohjelmistokehittäjät suorittavat white box -mallin mukaisen testauksen ohjelmaa kehittäessään.

3.6.7 Julkaisu

Ensimmäisen tuotantoversion julkaisu voidaan tehdä kun ohjelmisto saavuttaa kaikki alussa määritetyt tavoitteensa, ohjelmisto on kattavasti testattu sekä tarvittavat pilotoinnit on tehty.

Ohjelmaa jatkokehitetään jatkuvasti ja siitä julkaistaan ennalta määrätyin välein uusi versio.

3.6.8 Alustava aikataulu

Alustavassa aikataulussa arvioitiin kunkin moduulin tekniseen toteuttamiseen kuluvaa aikaa työpäivinä (Taulukko 7.).

Taulukko 7. Alustava aikataulu.

Osatoiminto	Käyttöliittymä	Ohjelmakoodi	Testaus
Tausta-arkkitehtuuri		10	2
Tietokantakerros		15	3
Liiketoimintakerros		15	3
Tutkimuksen mallintaminen	5	15	4
Tutkimuksen suunnittelu			
Aikataulu	5	5	2
Käyttäjien hallinta	5	10	3
Otannan hallinta	5	10	3
Tutkimussisällön hallinta	5	10	3
Raportoinnin hallinta	5	10	3
Tutkimuksen tiedonkeruu			
Lomakkeen hallinta	10	15	5
Vastaajien hallinta	5	10	3
Aineiston hallinta	5	5	2
Tutkimuksen raportointi	15	10	7
Yhteensä	*65*	140	43
Koko projekti	*248*		

4 POHDINTA

Työssä luotiin määritykset tietojärjestelmälle, jonka avulla voidaan toteuttaa räätälöityjä määrällisiä tutkimuksia. Ohjelmiston valmistuttuaan tulisi kasvattaa toimeksiantajan kilpailukykyä joustavuudellaan ja ketteryydellään. Tavoite saavutettiin ohjelmiston osalta. Määritetty ohjelmisto kykenee joustavaan ja ketterään tutkimuksen toteuttamiseen.

Nykyiset asiakastarpeet on pääosin huomioitu määrityksissä. Eri toimintojen osalta tarvitaan kuitenkin vielä lisää tarkennuksia käytännön tekemisen kautta asiakkaiden kanssa yhteistyössä. Tämän avulla voidaan yleiset näkemykset ja tekniikat tarkentaa tukemaan käytännön työtä. Asiakkailla on usein käytössään erilaisia henkilöstöhallinnon ohjelmistoja, joista voi olla mahdollista tuoda erilaisia henkilöstötutkimukseen tarvittavia tietoja. Näitä voivat olla esimerkiksi vastaajatiedot tai tulosten tarkastelutasot. Tiedon tuontia henkilöstöhallinnon ohjelmistoista ei ole käsitelty tässä työssä vaan se jää kokonaan osaksi ohjelmiston kehitystä.

Ohjelmiston eri rakenteet ovat suunniteltu siten, että eri toiminnallisuuksien jatkokehittäminen on mahdollisimman helppoa. Kuitenkin esimerkiksi määrityksessä kuvatut toiminnallisuudet ja niiden käyttöliittymät on jätetty hieman yleiselle tasolle. Niitä pitää vielä määrittää tarkemmin ohjelman kehittämisen edetessä.

Varsinkin käyttöliittymien osalta ei responsiivisuutta ole suunniteltu vielä riittävällä tasolla. Jokainen luotava näkymä tulee kuvata eri alustoilla ohjelmistokehityksen yhteydessä. Näkymiin tulee suunnitella modulaarinen rakenne, joka tukee lisäominaisuuksien luontia ilman haastavaa käytön opettelua toiminnallisuuksien laajetessa.

Tietokannan rakenne on toteutettu tukemaan jatkokehitystä erinomaisesti. Normalisoinnin vuoksi suorituskykytavoite saattaa aiheuttaa haasteita. Tietokannassa säilytettävät tiedot on normalisoitu siten, ettei samaa tietoa säilytetä monessa paikassa. Ohjelman noutaessa tietokannasta tietoa, voidaan joutua se keräämään useammasta paikasta. Tämä saattaa aiheuttaa viivettä verrattuna normalisoimattomaan tiedon tallennukseen. Normalisointi on kuitenkin ohjelmiston käyttötarkoituksen vuoksi tärkeää. Tutkimuksen tietoja ja dataa tallennettaessa tietoa tulee odotetusti paljon, jolloin normalisoimaton tietokanta kasvaisi kooltaan nopeasti. Tämä aiheuttaisi toimeksiantajalle kustannuksia tarvittavan levytilan ja varmuuskopiointirutiinien muodossa. Ohjelmistokehityksen edetessä saatetaan joutua

tästä syystä muuttamaan normalisoinnin tasoa, jotta asetettu suorituskykytavoite saavutetaan.

Tutkimuksen raportointi määritettiin hyvin yleisellä tasolla. Tarvittavia analyysejä voidaan lisätä ohjelmistoon asiakastarpeiden pohjalta.

Työssä ei ole määritetty kovin tarkkaan ohjelmiston elinkaarta. Uuden ohjelmiston kehittäminen on toimeksiantajalle kallis ja pitkäkestoinen projekti, jonka vuoksi elinkaari tulee suunnitella mahdollisimman pitkäksi. Työssä määritetty modulaarinen rakenne tukee tätä ajatusta erinomaisesti. Mahdollisuus moduulien lisäämiseen ja poistamiseen pidentää ohjelmiston elinkaarta.

Työssä nousi esiin monia asioita, jotka ohjaavat myös toimeksiantajan nykyisten järjestelmien ja ohjelmistojen kehityksen suuntaa. Esimerkiksi testauksen pohjaksi luodut tarkastuslistat sekä -käytännöt käyvät sellaisenaan myös toimeksiantajan nykyisten ohjelmistojen kehityksen tueksi. Työssä esitelty vaihejakomalli voisi olla erinomainen myös toimeksiantajan nykyisten ohjelmistojen kehityksen pohjaksi. Sen avulla voidaan helposti osallistaa ohjelmiston kehittämiseen myös henkilöitä joilla ei ole riittävää teknistä osaamista. Mallissa luodaan ensin käyttöliittymä, jonka pohjalta kehitykseen osallistuvien henkilöiden on helpompi hahmottaa miten kehitettävä toiminto tulisi toimimaan ja miten sitä tulisi jatkokehittää. Pelkän käyttöliittymän luominen on useasti kevyempää kuin valmiin toiminnallisuuden kehittäminen kokonaisuudessaan. Vaihejakomallilla myös vältytään tekemästä vääriä teknisiä ratkaisuja ennen kuin yhteinen ymmärrys on saavutettu jokaisen ohjelmistoa määrittävän osapuolen kanssa.

4.1 Haasteet ja oppiminen

Työn ohjelmiston määritys -osuutta oli tarkoitus tehdä pääosin normaalin työajan puitteissa, mutta muut työtehtävät kiireineen siirsivät suurimman osan työstä tehtäväksi työajan ulkopuolelle. Kuitenkin aikataulun mukaisesti saatiin resursoitua tarpeellinen aika henkilöstön haastattelemiseen.

Lähdemateriaalia aiheeseen löytyi runsaasti ja tarvittavien lähteiden löytäminen oli helppoa. Työn edetessä työn rajaus hieman tarkentui tarpeellisten toimintojen osalta. Tästä joh-

tuen toiminnallisuuksien tarkempi määrittäminen päätettiin siirtää osaksi ohjelmistokehitystä. Tällä tavoin saavutamme paremman näkemyksen sekä henkilöstön mielipiteistä, että asiakastarpeista toimintotasolla.

Työn aihepiiri oli erittäin mielenkiintoinen ja vaativa. Lähdemateriaaliin tutustuminen laajensi tietämystä ohjelmistomääritysten luomisesta, määrällisen tutkimuksen tekemisestä sekä yleisesti ohjelmistokehitysprojektin teoriasta. Ohjelmiston ylläpitovaiheeseen valmistui erinomaisia työkaluja lähdekoodin sekä käytettävyyden testaamiseen. Nämä työkalut soveltuvat erinomaisesti työssä määritetyn ohjelmiston sekä toimeksiantajalla jo olevien ohjelmistojen testausprosesseiksi sekä ylläpidon työkaluiksi.

5 YHTEENVETO

Työn tavoitteena oli luoda määritykset ohjelmistolle. Määritetyn ohjelmiston avulla voidaan toteuttaa räätälöityjä määrällisiä tutkimuksia ketterästi ja joustavasti toimeksiantajan muuttuneen toimialan vaatimusten mukaisesti. Työssä syntyi kattava määritys toimeksiantajalle ohjelmistokehityksen aloittamista varten. Määrityksissä on kuvattu tarpeelliset toiminnallisuudet ja suunniteltu käytettävät tekniikat ja ympäristö.

Määritysten pohjalta kehittävä ohjelmisto antaa toimeksiantajalle kilpailuedun toimialallaan. Ohjelmisto on suunniteltu joustavaksi, jolloin toimeksiantajan on helppo jatkokehittää sitä tukemaan nopeasti muuttuvia asiakastarpeita. Määrityksissä on kehitetty tutkimusprosessin läpivientiä ohjelmiston näkökulmasta, joten ohjelmisto tuottaa toimeksiantajalle kilpailuedun lisäksi myös liiketoiminnallista lisäarvoa kustannustehokkaamman toiminnan muodossa. Ohjelmistokehitysprojektille tyypilliseen tapaan määritykset tulevat luonnollisesti tarkentumaan ohjelmistonkehityksen edetessä.

Toimeksiantajan on tarkoitus työn tuloksena syntyneen määrityksen pohjalta aloittaa ohjelmistokehitys lähiaikoina. Toimeksiantajan henkilöstö on tarkoitus pitää mukana myös ohjelmistonkehityksen aikana. Heidän mielipiteitään sekä näkemyksiään tullaan kuuntelemaan eri osuuksien kehittämisen yhteydessä. Toimeksiantajalla on tarkoitus perustaa tekninen ohjausryhmä tukemaan ohjelmiston kehitystä sen edetessä. Ohjausryhmä koostuu asiantuntijoista, jotka ovat olleet mukana toimeksiantajan aiempien ohjelmistojen ja järjestelmien kehittämisessä.

LÄHTEET

Auer, L. 2009. Nielsenin säännöt. Viitattu 14.2.2016 http://www2.amk.fi/digma.fi/www.amk.fi/opintojaksot/030308/1146204519802/1146224777754/1146226177242/1146226608346.html

Comodo Instant SSL 2016. What is HTTPS. Viitattu 14.2.2016 https://www.instantssl.com/ssl-certificate-products/https.html

Corporate Spirit 2015. Henkilöstötutkimus. Viitattu 5.4.2016 http://www.corporatespirit.fi/Henkilostotutkimus

CSS3.info 2009. Everything you need to know about CSS3. Viitattu 14.2.2016. http://www.css3.info

ESOMAR 2016. Markkina- ja mielipidetutkimusalan kansainvälisen kattojärjestön ESOMARin kotisivut. Viitattu 12.3.2016 https://www.esomar.org

Gavalda, M. 2015. A Beginner's Guide to Website Speed Optimization. Viitattu 13.2.2016 https://kinsta.com/learn/page-speed

Häkkinen, N. 2014. Fittsin laki ja sen hyödyntäminen käyttöliittymäsuunnittelussa. Viitattu 15.2.2016 http://users.jyu.fi/~nipehakk/tekstit/hakkinen-fittsin-laki.pdf

Joensuun yliopisto, Tietojenkäsittelytieteen laitos 2007. Vaatimusmäärittely. Viitattu 13.2.2016 http://cs.joensuu.fi/tSoft/vaatimusmaarittely.

Järvelä, E. & Puusaari, E. 2005. UML-käsikirja. Kokkola: Jyväskylän yliopisto.

Kautto, T. 1996. Ohjelmistotestaus ja siinä käytettävät työkalut. Viitattu 16.2.2016 http://www.mit.jyu.fi/opiskelu/seminaarit/bak/testaus/#RTFToC20

Kuopion yliopisto, Tietojenkäsittelytieteen laitos 2002. Verkko-ohjelmointi, Luento 7. Viitattu 14.2.2016 http://www.cs.uku.fi/~mhassine/VOH/Luennot/lu7.html

KvantiMOTV 2004. Tilastollinen päättely. Viitattu 17.1.2016 http://www.fsd.uta.fi/menetelmaopetus/paattely/paattely.html

Laine, H. 2005. Tietokantojen perusteet, Tietosisällönkuvaus ER-mallilla. Viitattu 14.2.2016 http://www.cs.helsinki.fi/u/laine/tkp/tietomallit/ermalli.html

Leiniö, T. 2012. Mitä on resposiivinen desing?. Viitattu 14.2.2016 https://www.sofokus.com/blogi/mita-on-responsiivinen-design

Mains, B. 2008. Introduction to 3-Tier Architecture. Viitattu 14.2.2016 http://dotnetslackers.com/articles/net/IntroductionTo3TierArchitecture.aspx

Microsoft 2008. Tietokannan normalisoinnin perusteiden kuvaus. Viitattu 14.2.2016 https://support.microsoft.com/fi-fi/kb/283878

Microsoft Technet 2011. Network Load Balancing. Viitattu 15.2.2016 https://technet.microsoft.com/fi-fi/library/cc732855(v=ws.10).aspx

Network Working Group 2008. The Transport Layer Security (TLS) Protocol Version 1.2. Viitattu 14.2.2016 http://tools.ietf.org/html/rfc5246#page-4

Nielsen J. 1995a. 10 Usability Heuristics for User Interface Design. Viitattu 15.2.2016 https://www.nngroup.com/articles/ten-usability-heuristics

Nielsen, J. 1995b. How to Conduct a Heuristic Evaluation. Viitattu 16.2.2016 https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation

Paakki, J. 2011. Ohjelmistojen vaatimusmäärittely. Viitattu 13.2.2016 http://www.cs.helsinki.fi/u/paakki/Vaatimus-11-Luentokalvot-1.pdf

Pietilä, M. 2006. Fittsin laki. Viitattu 15.2.2016 https://koyttoliittyma.wordpress.com/2006/08/27/fittsin-laki

Recall 2015. Katastrofista palautuminen. Viitattu 14.2.2016 http://www.recall.fi/data-protection/disaster-recovery-services

Tamplin, J. 2011. Pseudolocalization to Catch i18n Errors Early. Viitattu 16.2.2016 http://google-opensource.blogspot.fi/2011/06/pseudolocalization-to-catch-i18n-errors.html

Techopedia 2016. Transact-SQL (T-SQL). Viitattu 21.2.2016 https://www.techopedia.com/definition/24476/transact-sql-t-sql

Tietosuojavaltuutetun toimisto 2013. Henkilötietolaki. Viitattu 12.3.2016 http://www.tietosuoja.fi/fi/index/lait/Henkilotietolaki.html

Tilastokeskus 2002. Asteikko on tilaston perusta. Viitattu 17.1.2016 http://www.stat.fi/tup/tietoaika/tilaajat/ta_07_02_melkas.html

Vilkka, H. 2007. Tutki ja mittaa. Määrällisen tutkimuksen perusteet. Helsinki: Tammi.

W3C 2014. HTML5. Viitattu 14.2.2016 https://www.w3.org/TR/html5

W3C 2015a. Extensible Markup Language (XML). Viitattu 14.2.2016 https://www.w3.org/XML

W3C 2015b. XML Schema. Viitattu 14.2.2016 https://www.w3.org/XML/Schema

Webhotellivertailu² 2015. Mikä on SSL-salaus tai TLS-salaus?. Viitattu 14.2.2016 http://www.webhotellivertailu2.fi/mika-on-ssl-salaus-tai-tls-salaus

Haastattelun pohja

1. Haastateltavan tausta
    a. Nimi:
    b. Työssäoloaika:
    c. Tehtävä:

2. Nykytila ja kehittämiskohteet
    a. Mallinnus
    b. Suunnittelu
    c. Tiedonkeruu
    d. Raportointi

3. Tulevaisuuden edellytykset ja kehittämiskohteet
    a. Mallinnus
    b. Suunnittelu
    c. Tiedonkeruu
    d. Raportointi

4. Muut huomiot ja kehittämiskohteet

Käyttöliittymätestauksen tarkastuslista

	Havaitut ongelmat Kirjoita tähän lyhyesti	**Vakavuus-luokka** 0 = Ei ongelmaa, 1 = Pieni ongelma - 4 = Katastrofaalinen ongelma
1. Taustatiedot		
a. Testauksen kohde		
b. Testaaja ja ajankohta		
c. Arvioija ja ajankohta		
2. Tuotteen tilan näkyvyys		
Käyttäjän pitäisi aina pystyä nopeasti huomaamaan mikä on tuotteen tila tai toiminto.		
a. Mitä toimintoja tuotteella voi tehdä (kehottaako se tiettyyn toimintoon)?		
b. Antaako tuote palautetta, kun käytän sitä oikein/väärin?		
c. Kun olen tehnyt tietyn työvaiheen, kertooko tuote, että nyt vaihe on valmis?		
3. Tuotteen ja tosielämän vastaavuus		
Tuotteen ja sen ohjeiston tulisi käyttää tavallisesta elämästä tuttuja termejä, sanontoja ja käsitteitä mieluummin kuin omaa erikoistermistöä.		
a. Toimivatko metaforat loogisesti?		
b. Onko tuotteen käyttö ristiriidassa muun maailman toimintaan?		
c. Onko käytetty kieli helppoa ymmärtää?		

4. Käyttäjän kontrolli ja vapaus		
Käytön pitäisi olla tuotteen käyttäjän määrättävissä, eikä päinvastoin. Peru ja Tee uudestaan toiminnot ovat suositeltavia. Kokeileva käyttö, josta ei aiheudu ongelmia, on suotavaa.		
a. Voiko tuotetta käyttää haluamassaan järjestyksessä vai määrääkö tuote vaiheiden logiikan?		
b. Voiko tuotteen ominaisuuksia kokeilla turvallisesti?		
5. Yhteneväisyys ja standardit		
Tuotteessa tulisi käyttää viestejä ja toimintoja yhteneväisesti tarkoittamaan aina samoja asioita (eikä vaihtaa merkityksiä lennossa). Hyvä tuote tukee opitun siirtämistä niin, että olemassa olevien käyttöstandardien avulla on helppo käyttää myös uutta tuotetta.		
a. Onko värejä, muotoja, tekstuureja, äänimerkkejä ja muita muotoiltuja ominaisuuksia käytetty yhteneväisesti tukemaan käytön ymmärtämistä?		
b. Onko tuotteen käyttö helposti päätteltävissä muiden (samankaltaisten) tuotteiden osaamisella?		
c. Toimiiko tuotteen käyttö loogisesti eri tilanteissa ja työvaiheissa?		
6. Virheiden estäminen		
Erinomaiset virheen tunnistukset ja ilmoitukset estävät virheiden syntymistä ja toistumista. Opastus tulisi olla aina helposti saatavilla ja ymmärrettävissä.		
a. Edellyttääkö onnistunut käyttö ohjeiden lukemista?		
b. Voiko tuotetta käyttää helposti virheellisesti, vai estääkö tuote virheellisen käytön?		

7. Tunnistaminen mielummin kuin muistaminen		
Tuotteen toiminnat ja vaihtoehdot tulisi olla näkyviä. Käyttöliittymän osat ja niiden kontrolloimat toiminnot olisi liitettävä toisiinsa loogisesti, niin että näiden yhteys on pääteltävissä tuotteesta. Käyttäjän ei tarvitsisi muistaa tuotteen käyttöä tehdessään tuotteella eri työvaiheita.		
a. Ovatko tuotteen aktiiviset elementit muotoiltu niin, että ne ymmärtää aktiivisiksi?		
b. Onko eri työvaiheissa tarvittavat ominaisuudet sijoitettu niin, että aina seuraavan vaiheen toimintoihin siirtyminen on luontevaa?		
c. Onko tuotetta helppo alkaa käyttämään opettelematta tai lukematta käyttöohjeita?		
d. Edellyttääkö käyttö tarkkaa keskittymistä ja muistamista – ja rikkooko työvaiheen keskeyttäminen onnistuneen käytön helposti?		
8. Käytön joustavuus ja tehokkuus		
Käytön tulisi olla joustavaa ja tehokasta sekä aloitteleville että edistyneille käyttäjille. Tue pikavalintoja ja henkilökohtaisia tapoja käyttää eri tavoin. Käytön tulisi olla myös joustavaa ja tehokasta käyttäjästä riippumatta (muista myös erityistarpeiset).		
a. Ovatko yleisimmät toiminnot helpoiten käytettävissä?		
b. Voiko tuotetta käyttää usealla eri tavalla onnistuneesti?		
c. Onnistuuko tuotteen käyttö näkö-, kuulo-, motoriikka-, tunto- tai muuten rajoittuneelta käyttäjältä?		

9. Esteettinen ja minimalistinen design		
Tuotteessa tulisi olla vain halutun tiedon, toiminnot, tunnelman ja tyylin ilmaisevat muodot, ei enempää. Esteettisen ilmaisun ei tulisi olla vaikeasti ymmärrettävää (ellei se ole tuotteen kantava idea).		
a. Onko tuotteessa käytetty hallitusti värisävyjä, valooriarvoja ja värikoodauksia?		
b. Onko muotoja käytetty miellyttävällä ja johdonmukaisella tavalla?		
c. Onko tyhjää tilaa hyödynnetty selkeyttämään tuotteen ominaisuuksien hahmottumista?		
d. Kiinnittyykö huomio tärkeimpiin elementteihin ensin?		
e. Hallitseeko yksi (tai useampi) elementti koko tuotetta muiden kustannuksella?		
f. Onko mahdollinen teksti sopivan mittaista, tyylistä ja kokoista, jotta lukeminen onnistuu.		

10. Virhetilanteiden tunnistaminen, ilmoittaminen ja korjaaminen		
Virheilmoitusten tulisi selvitä helposti: mitä tapahtui, miksi näin kävi, miten asia voidaan korjata ja kuinka se voidaan välttää ensi kerralla.		
a. Onko virheilmoitus ymmärrettävissä?		
b. Selviääkö virhesignaalista mitä tapahtui, miksi ja miten korjata/välttää tilanne?		
c. Ovatko virheilmoitukset kohteliaita ja välttävät syyttelyä?		
d. Ovatko korjaavat toimintaohjeet helposti seurattavissa?		
11. Opastus ja ohjeistus		
Vaikka pyrkimys olisikin selvitä ilman opastusta ja ohjeita, ovat ne usein välttämättömiä käyttäjille. Näiden tulisi olla helposti saatavilla, nopeasti etsittävissä, toimintaan opastavia, käyttäjän toimintaa tukevia ja riittävän lyhyitä.		
a. Annetaanko opastusta automaattisesti vaikeissa paikoissa?		
b. Ovatko ohjeet aina saatavilla?		
c. Ovatko ohjeet ja opastus käyttötilanne- ja toimintokohtaisia?		
d. Ovatko ohjeet helposti ymmärrettävissä ja vaiheet toteutettavissa?		
e. Ovatko ohjeet lyhyitä (lyhyisiin, mutta kokonaisiin osiin pilkottuja)?		

Testauksen tarkastuslista

	Havaitut ongelmat Kirjoita tähän lyhyesti
Taustatiedot	
f. Testauksen kohde	
g. Testaaja ja ajankohta	
Numeeriset arvot	
liian pieni arvo	
jokin arvo väliltä minimi – maksimi	
liian suuri arvo	
Päivämäärä arvot	
liian pieni arvo	
jokin arvo väliltä minimi – maksimi	
liian suuri arvo	
Tekstiarvot	
merkkijonon pituus: nolla (tyhjä merkkijono)	
merkkijonon pituus: ylittää maksimin	
normaalimittainen merkkijono	
numeroiden, kirjaimien ja erikoismerkkien yhdistelmät merkkijonossa	

Erikoistilanteet	
vuoden / vuosituhannen vaihde	
tyhjät kentät	
negatiiviset luvut	