TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

HANS AHRENBERG

TIETOVARASTO- JA RAPORTOINTIJÄRJESTELMIEN TESTAUS

Diplomityö

Tarkastaja: professori Samuli Pekkola
Tarkastaja ja aihe hyväksytty tuotantotalouden ja rakentamisen tiedekuntaneuvoston kokouksessa 6. maaliskuuta 2013

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Tietojohtamisen koulutusohjelma

AHRENBERG, HANS: Tietovarasto- ja raportointijärjestelmien testaus
Diplomityö, 125 sivua, 5 liitesivua
Kesäkuu 2013
Pääaine: Tiedonhallinta
Tarkastaja: professori Samuli Pekkola
Avainsanat: Tietovarastointi, Business Intelligence, raportointi, ETL, testaus

Tutkimuksessa keskitytään tietovarasto- ja raportointijärjestelmien testaukseen. Tietovarastojärjestelmät mahdollistavat raportoinnin, jolla pyritään vastaamaan organisaatioiden kasvaviin informaatiotarpeisiin. Vaikka tietovarastoinnin ja raportoinnin käsitteet ovat jo vuosikymmeniä vanhoja, ja tietovarasto- ja raportointijärjestelmiä on hyödynnetty organisaatioissa pitkään, testaus on saanut suhteellisen vähän huomiota. Tietovarasto- ja raportointijärjestelmät tuottavat informaatiota, jonka avulla luodaan tietämystä ja tehdään sekä strategisia että operatiivisia päätöksiä. Oikeiden ja luotettavien päätösten tekeminen edellyttää oikeellista ja luotettavaa informaatiota, mikä edellyttää järjestelmän testausta ennen sen käyttöönottoa ja käyttöönoton jälkeen, sillä tietovarastoon ladataan jatkuvasti uutta tietoa, määrityksiä muutetaan ja uusia informaatiotarpeita herää ajan kuluessa. Tämän tutkimuksen tarkoituksena oli selvittää, miten tietovarasto- ja raportointijärjestelmiä tulisi testata ja mitä testauksessa tulisi ottaa huomioon. Tavoitteena oli muodostaa hyvien käytäntöjen mukaisia suosituksia eri teemoihin liittyen.

Tutkimuksen rakenne on kaksijakoinen. Teoreettinen osio perustuu käsiteanalyyttiseen kirjallisuustutkimukseen, jonka tarkoitus oli etsiä vastauksia tutkimuskysymyksiin. Tutkimuskysymykset muodostettiin ennalta valittujen teemojen perusteella. Tutkimuksessa käsiteltäviä teemoja ovat tietovarasto- ja raportointijärjestelmien erityispiirteet testauksen näkökulmasta, testauksen keskeiset haasteet, toistuvat ongelmat, testausprosessi, testausvaiheet sekä asiakkaan tukeminen. Tutkimuksen empiirinen osio toteutettiin näiden teemojen mukaisesti ja siinä noudatettiin toiminta-analyyttistä tutkimusotetta. Empiirinen lähdeaineisto kerättiin teemahaastatteluin.

Tutkimuksen tulokset muodostavat laaja-alaisen näkemyksen testauksen hyvistä käytännöistä ja tarjoavat karkean tason suosituksia organisaatioille, jotka hyödyntävät tai aikovat hyödyntää erilaisia tietovarastointisovelluksia liiketoiminnassaan. Nämä suositukset tarjoavat näkökulman, joka koostaa tietovarasto- ja raportointijärjestelmien testaukseen liittyvän perustietämyksen aiheesta kiinnostuneille ja luo samalla lähtökohdat mahdollisille jatkotutkimuksille. Tuloksista käy ilmi, että tietovarasto- ja raportointijärjestelmien testaus eroaa merkittävästi perinteisestä ohjelmistotestauksesta ja pitää sisällään yksityiskohtia, jotka kulminoituvat tietosisältöön ja sen käsittelyyn.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information and Knowledge Management

AHRENBERG, HANS: Testing of Data Warehouse and Business Intelligence Systems
Master of Science Thesis, 125 pages, 5 Appendix pages
June 2013
Major: Business Information Management
Examiner: Professor Samuli Pekkola
Keywords: Data Warehousing, Business Intelligence, reporting, ETL, testing

Data warehouse and business intelligence systems provide information that can be used in both strategic and operative decision making processes in organizations. Although data warehousing and business intelligence concepts have been used for decades, and data warehouse and business intelligence systems have been in use for a long time, testing as a part of the development lifecycle has received only a little attention. Making the right and reliable decisions requires also reliable and high-quality information, which, respectively, requires proper testing activities before and after the deployment of the system. Therefore, this research focuses on testing of data warehouse and business intelligence systems. The purpose of this research is to determine how data warehouse and business intelligence systems should be tested, and what kind of critical factors should be taken into account in the testing phase. The aim was to establish a list of recommendations that are based on good practices.

The structure of this research is two-folded. The theoretical part is based on conceptual literature review, which purpose was to seek answers to research questions formed on the basis of pre-selected themes. These themes consists of data warehouse and business intelligence systems' special characteristics from the testing point of view, the main challenges related to testing, recurrent problems, testing phases, and supporting customer in testing activities. The empirical part was carried out in accordance to these themes, and it follows the action-oriented research approach. Theme interviews were exploited to gather the empirical research material.

The results of this research provide a comprehensive perspective to good practices in testing of data warehouse and business intelligence systems in the form of list of high-level recommendations. These recommendations can be useful for organizations that utilize or are planning to exploit data warehouse applications in their business. The results show clearly that testing of data warehouse and business intelligence systems differs significantly from traditional software testing, and it culminates in data content related processes and activities.

ALKUSANAT

Tämän diplomityön tekeminen oli haasteellinen projekti. Työn aiheen kartoitus aloitettiin jo keväällä 2012, jolloin vaihtoehtoja oli useampia. Aiheen lopullinen valinta tehtiin toukokuussa 2012, jolloin kohdeyrityksen järjestämässä tietovarastointiin liittyvässä koulutuksessa heräsi kysymys siitä, mitä tietovarastojärjestelmien testaus oikeastaan pitää sisällään. Aktiivinen työskentely alkoi kesällä tutkimussuunnitelman tekemisellä, mutta pian syksyllä pidetyn ohjauspalaverin jälkeen työskentely jäi pitkälle tauolle muiden työkiireiden vuoksi. Työn tekeminen sai piristysruiskeen vuoden vaiheessa, jonka jälkeen allekirjoittaneen arki-illat ja viikonloput täyttyivät yllättävän nopeasti diplomityöhön liittyvillä asioilla. Lopulta haasteeksi muodostui aikataulu, sillä kevään mittaan aikaa oli käytettävissä melko vähän. Työn edistämiseksi otinkin kaksi viikon mittaista diplomityövapaata, jotka osoittautuivat jälkikäteen erittäin arvokkaiksi työn etenemisen kannalta.

Haluan kiittää tämän diplomityön ohjaajia, professori Samuli Pekkolaa, Jukka Lepolaa ja Petri Kiviahoa, arvokkaista, rakentavista neuvoista ja kommenteista, joiden pyytäminen ja saaminen oli vaivatonta koko diplomityöprosessin ajan. Ne laajensivat näkemyksiäni ja saivat pohtimaan asioita eri näkökulmista, jolloin toimivien ratkaisujen löytäminen oli helpompaa. Lisäksi haluan kiittää kaikkia, jotka olivat järjestämässä haastatteluita ja antoivat mahdollisuuden haastattelulle. Viimeiseksi haluan osoittaa erityiskiitokset isälleni, Harrylle, sekä Iidalle, jotka tukivat läpi diplomityöprosessin.

Helsingissä 9.6.2013

_____

Hans Ahrenberg

SISÄLLYS

TERMIT JA NIIDEN MÄÄRITELMÄT

Dimensiotaulu	Dimensiotaulut sisältävät dimensioita, jotka luokittelevat tapahtumia. Esimerkiksi asiakasdimensiota voidaan käyttää analysoimaan tilauksia sijainnin, ostajan ym. mukaan. (Khan 2012, s. 153.)
ETL-prosessi	ETL-prosessi koostuu kolmesta keskeisestä komponentista: poiminnasta, transformaatiosta ja latauksesta. ETL-prosessissa poimitaan tietoa lähdejärjestelmistä, muunnetaan sitä eri operaatioiden avulla ja ladataan fyysisesti kohdetietokantaan. (Khan 2012, s. 71.)
ETL-työ	ETL-prosessi voi muodostua yhdestä tai useammasta ETL-työstä. ETL-työ sisältää ETL-prosessille ominaiset kolme komponenttia.
Faktataulu	Faktataulut sisältävät faktatietoa eli liiketoiminnan suorituskykyä mittaavia arvoja, joita voidaan tarkastella dimensioittain. Fakta on mittari, joka mittaa tiettyä liiketoiminnan määrettä. (Khan 2012, s. 151.)
Federointi	Federoinnilla tarkoitetaan olemassa olevien järjestelmien ja ratkaisujen hyödyntämistä keskitetysti. (Khan 2012, s. 137.)
Luonnollinen avain	Luonnollinen avain yksilöi tietueen ja sisältää tyypillisesti jonkin merkityksen.
Materialisoitu näkymä	Materialisoitu näkymä on fyysinen tietokantataulu, johon on tallennettu ennalta jokin rajattu tulosjoukko yhden tai useamman tietokantataulun pohjalta.
Metatieto	Metatieto on tietoa tiedosta. Se on kriittinen osa tietovarastoarkkitehtuuria tarjoten tietoa muun muassa siitä, missä tiedot sijaisevat tietovarastossa ja millaista tieto on. Esimerkiksi tietovaraston tietokannan taulujen sarakkeiden tietotyypit ovat metatietoa. (Khan 2012, s. 48.)
Mustalaatikkotestaus	Testaustekniikka testitapausten muodostamiseksi ja valitsemiseksi hyödyntämällä järjestelmäkomponentin tai jär-

	jestelmän toiminnallisia tai ei-toiminnallisia määrityksiä. (ISTQB 2012.)
Näkymä	Näkymä pohjautuu yhteen tai useampaan tietokantatauluun ja esittää tietyin ehdoin rajatun tulosjoukon taulusta tai tauluista. Näkymä muodostetaan tietokantakyselyn avulla.
OLTP-järjestelmä	OLTP-järjestelmä on operatiivinen, transaktiopohjainen tietojärjestelmä, jota käytetään organisaatioiden päivittäisten operaatioiden suorittamiseksi. (Khan 2012, s. 20.)
OLAP	OLAP on BI-työkalu, joka mahdollistaa multidimensionaalisten analyysien suorittamisen perustuen analyyttiseen tekniikkaan, joka yhdistää datan pääsy sovelluksia analyyttiseen tietokantaan. Analyyttinen tietokanta käyttää dimensionaalisia näkymiä, joissa tarkastellaan esimerkiksi myyntiä tietyn tuotteen mukaan. (Khan 2012, s. 157.)
ODS	ODS on operatiivinen tietovarasto, joka sisältää strukturoitua dataa, joka on poimittu suoraan operatiivisista lähteistä. Sen tarkoitus on täyttää ad hoc kyselytarpeet ja päivittäiset taktiset informaatiotarpeet. ODS sisältää tyypillisesti lähes reaaliaikaista dataa. (Khan 2012, s. 20.)
Surrogaattiavain	Surrogaattiavain on keinotekoisesti muodostettu yhden tietueen yksilöivä avainsarake, mikä ei sisällä merkitystä.
Transformaatio	Transformaatio eli tiedon muunnosprosessi sisältää yhden tai useampia operaatioita, joiden avulla tietoa muunnetaan.
Valkolaatikkotestaus	Testaus, joka perustuu testattavan järjestelmäkomponentin tai järjestelmän rakenteeseen. (ISTQB 2012.)

1 JOHDANTO

1.1 Tutkimuksen tausta

Teknologian nopea kehittyminen ja muutokset yritysten toimintaympäristössä synnyttävät uusia mahdollisuuksia ja ennalta-arvaamattomia uhkia. Näin ollen yritykset eivät kykene säilyttämään kilpailukykyään tukeutumalla vanhoihin toimintatapoihin ja perinteisiin johtamismalleihin sillä muutos on jatkuvaa. Toisaalta eräs merkittävä muutos on yritysten sisäisten sekä ulkoisten tietovarantojen räjähdysmäinen kasvu. Tiedolla ja tiedon hallinnalla on nykyään merkittävä rooli yritysten liiketoiminnassa erityisesti kilpailukyvyn kannalta. Hovi (2001, s. 10) on todennut, että yritys, joka kykenee hyödyntämään tietoa tehokkaasti, on kilpailijoihinsa nähden edullisemmassa asemassa. Tieto eri muodoissaan (data, informaatio, tietämys) on siis kriittinen menestystekijä, jos sitä vain hallitaan tehokkaasti. Yhä useammin yrityksen operatiivinen liiketoiminta ja toisaalta strateginen päätöksenteko perustuvat käytettävissä olevaan tietoon tai tarkemmin informaatioon. Yrityksen menestyksekäs johtaminen edellyttää nopeita ja oikeita päätöksiä liiketoiminnan eri tasoilla. Menestys onkin seurausta oikeaan aikaan tehdyistä oikeista päätöksistä. Muuttuva liiketoimintaympäristö ja tulevaisuuden huono ennustettavuus pakottavat päätöksentekijät tarkastelemaan asioita aiempaa laajemmin ja pohtimaan myös eri tekijöiden syy-seuraussuhteita, mikä edellyttää asianmukaisia teknologioita ja toimintatapoja. Tietovarastointi (engl. Data Warehousing) on eräs tiedonhallinnan keskeisin menetelmä tai järjestelmä strategisen informaation tuottamiseksi (Hovi 2001, s. 18), ja se tarjoaa mahdollisuuden yritykselle parantaa strategista päätöksentekoa, markkinasignaaleihin reagointia, luotettavaa muutosten ennakointia, trendien analysointia ja toisaalta liiketoiminnan päivittäistä raportointia ja ohjaamista (Watson et al. 2002). Tietovarastointi on olennainen osa liiketoiminnan kehittämistä (Hovi 2001, s. 18) ja sen avulla yritykset voivat vastata tietolähteiden ja -määrien eksponentiaaliseen kasvuun yrittäessään tehdä oikeita päätöksiä käytettävissä olevien tietovarantojen pohjalta (Watson et al. 2002). Koska tiedolla on olennainen merkitys päätöksenteossa, tietovarastojen on tarjottava oikeaa, laadukasta informaatiota oikeaan aikaan, oikeassa paikassa käyttäjilleen. Tämä johtaa siihen, että tietovarasto- ja raportointijärjestelmiä tulisi testata osana kehityselinkaarta ja vielä tuotannossakin. Golfarelli & Rizzi (2011) ovat sitä mieltä, että testaus on olennainen osa mitä tahansa ohjelmistoprojektia, mutta korostavat sen merkitystä erityisesti tietovarasto- ja raportointijärjestelmien tapauksessa, sillä käyttäjien on pystyttävä luottamaan saatavilla olevaan informaatioon. Riittävä ja asianmukainen testaus ei välttämättä takaa menestystä mutta mahdollistaa sen, että tietovarastoa pystytään käyttämään tehokkaasti ja että siitä saadaan paras mahdollinen hyöty suhteessa sen aiheuttamiin kustannuksiin. Testauksella voidaan ottaa myös kantaa siihen, kuinka hy-

vin tietovarasto vastaa yrityksen tarpeita. Tässä tutkimuksessa perehdytään erityisesti tietovarasto- ja raportointijärjestelmien testaukseen osana kehityselinkaarta.

1.2 Tutkimusongelma ja tutkimuksen tavoitteet

Tämän tutkimuksen kohdeyritys on Rongo Oy (jäljempänä Rongo), joka tuottaa informaation hallinnan konsultointipalveluita. Rongolle tietovarasto- ja raportointijärjestelmäprojektien onnistuminen ja korkea laatu ovat hyvin tärkeitä tekijöitä strategisten tavoitteiden saavuttamiseksi ja toisaalta kriittisiä tekijöitä myös asiakkaan liiketoiminnan kannalta. Rongo haluaa toimittaa laadukkaita palveluita, jotka tuottavat aidosti hyötyä asiakkaalle, sekä kehittää brändiarvoaan ja tunnettuuttaan erottautuakseen kilpailijoistaan. Näin ollen laadunvarmistaminen ja oikeassa suhteessa tapahtuva testaaminen on keskeinen tekijä Rongon liiketoiminnan jatkuvuuden näkökulmasta. Toimialan luonteelle on ominaista, että asiakas haluaa ennen kaikkea laadukkaita toimituksia, eikä mielellään maksa esimerkiksi puutteellisesta testauksesta aiheutuvia virhekustannuksia, mikä vaikuttaa suoraan asiakastyytyväisyyteen. Rongon kaltaisessa yrityksessä järjestelmätoimitusten testauksen merkitystä tulisikin korostaa. Varsinaisen toimeksiantajan eli kohdeyrityksen esittelyyn palataan myöhemmin luvussa 5.

Tutkimuksen tavoitteena on selvittää, miten organisaatioiden tiedonhallintaa tukevat tietovarasto- ja raportointijärjestelmät tulisi testata eli millaisia hyviä käytäntöjä tai menettelytapoja testauksessa tulisi noudattaa ja mitä huomion arvioisia ominaispiirteitä testauksessa tulisi ottaa huomioon. Tutkimuksen tulosten tarkoitus on havainnollistaa tietovarasto- ja raportointijärjestelmien testauksen pääperiaatteet ja testaukseen liittyvät erityispiirteet tietovarastoinnin ja raportoinnin kontekstissa sekä jakaa tietämystä tietovarasto- ja raportointijärjestelmien testaukseen liittyvästä problematiikasta.

Tutkimuksessa nojaudutaan sekä teoreettiseen että empiiriseen tutkimusaineistoon. Teoreettinen tutkimusaineisto kootaan aiemmin julkaistuista tutkimuksista, tutkimusartikkeleista, kirjallisuudesta, konferenssijulkaisuista sekä muusta vastaavasta virallisesta dokumentaatiosta. Empiirinen tutkimusaineisto kerätään haastattelututkimuksen menetelmin ja sitä analysoidaan aineiston sisällönanalyysin menetelmin. Haastatteluissa hyödynnetään Rongon omien asiantuntijoiden sekä Rongon ulkopuolisten asiantuntijoiden tietämystä aiheesta.

Tutkimusongelma määritellään seuraavasti:

- **Miten tietovarasto- ja raportointijärjestelmiä tulisi testata ja mitä testauksessa tulisi huomioida?**

Tutkimusongelman jäsentämiseksi ja ratkaisun löytämiseksi on laadittu tarkentavia tutkimuskysymyksiä, joihin tässä tutkimuksessa pyritään vastaamaan:

- Mitä tietovarasto- ja raportointijärjestelmällä tarkoitetaan ja mitä erityispiirteitä siihen liittyy testauksen näkökulmasta?

- Miten testauksen tulisi asettua osaksi tietovarasto- ja raportointijärjestelmän kehityselinkaarta?

- Mitä keskeisiä haasteita tietovarasto- ja raportointijärjestelmien testaukseen liittyy?

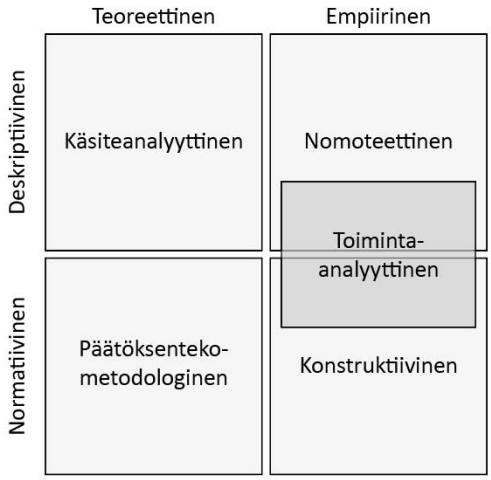
- Mitkä ovat tietovarasto- ja raportointijärjestelmien testauksen menestystekijät?

- Millainen on tietovarasto- ja raportointijärjestelmien testausprosessi ja mitä hyviä käytäntöjä siihen liittyy?

- Miten eri järjestelmäkerroksia tulisi testata ja mitä eri testausvaiheissa tulisi ottaa huomioon?

1.3 Tutkimusmenetelmät

Positivismi ja hermeneutiikka ovat kaksi vallalla olevaa tieteenkäsitystä. Positivismi perustuu vain ja ainoastaan tosiasioihin välttämällä epävarmuutta ja hylkäämällä havaitsemattomissa olevat asiat. Vastaavasti hermeneutiikassa pyritään keskittymään asioiden selitettävyyteen ja käsitteiden tulkintaan. Tämä tutkimus perustuu ensisijaisesti positivismiin, sillä se suosii helposti strukturoitavia tutkimuskohteita ja ongelmia. (Olkkonen 1994, ss. 26-27) Tietovarasto- ja raportointijärjestelmien testaamiseksi on olemassa strukturoitavissa olevia säännönmukaisuuksia, mutta varsinainen informaatio on hajautunut osaksi organisaation rakenteita ja henkilöstön hiljaista tietämystä. Lisäksi ratkaisumallit, projektit ja projektiorganisaatioiden muodostama tietämys voivat vaihdella, minkä johdosta kaikkia säännönmukaisuuksia ei voida standardisoida tai yleistää kovin tarkasti. Epävarmuuden vuoksi tutkimuksen voidaan katsoa noudattavan osin myös hermeneutiikan ominaispiirteitä.



Kuva 1.1. Tutkimusotteiden luokittelu (mukailtu lähteestä Kasanen et al. 1991, s. 302)

Liiketaloudellisten tutkimusten taustalla on käsiteanalyyttinen, nomoteettinen, päätöksentekometodologinen tai toiminta-analyyttinen tutkimusote, mitkä yhdessä muodostavat tutkimusotteiden nelijaon (Olkkonen 1994, s. 61; Neilimo & Näsi 1980). Käsiteanalyyttisissä ja päätöksentekometodologisissa tutkimusotteissa tietoa hankitaan teorialähtöisesti, kun vastaavasti nomoteettiset ja toiminta-analyyttiset tutkimusotteet perustuvat pääasiallisesti tutkimalla havainnollistettuun tietoon eli empiriaan (Olkkonen 1994, s. 78; Kasanen et al. 1991, s. 302). Konstruktiivinen tutkimusote on normatiivinen ja siinä keskitytään lähtökohtaisesti kehittämään ongelmanratkaisumenetelmiä (Olkkonen 1994, s. 76). Tutkimusotteet voidaan jakaa myös deskriptiivisiin ja normatiivisiin. Deskriptiivinen tutkimus pyrkii kuvailemaan ilmiöitä luokittelemalla ja mallintamalla tosielämän objekteja, kuvailemalla käsitteitä ja prosesseja sekä selittämällä syy-seuraussuhteita. Deskriptiivisen tarkastelun systemaattisuus luo puitteet asian syvälliselle ymmärtämiselle, jonka avulla löydetään uusia ratkaisuja. Normatiivisessa tutkimuksessa sen sijaan tuotetaan normeja, ohjeita, ratkaisumenetelmiä, joilla kehitetään toimintaa. Tällöin olennaista on pystyä osoittamaan tutkimuksen hyödyt. (Olkkonen 1994, ss. 44-45) Tutkimusotteiden jaottelu on esitetty kuvassa 1.1.

Tutkimuksen teoriaosuus pyrkii kuvailemaan tosielämän ilmiöitä, joten se noudattaa käsiteanalyyttistä tutkimusotetta. Tutkimuksen empiriaosuudessa vastaavasti tehdään havaintoja tosielämästä ja pyritään analyysin avulla esittämään kelvollisia toimenpidesuosituksia tai ratkaisuvaihtoehtoja korostamalla tutkimuskohteen hyötyjä. Tutkimuksen empiriaosuutta ei voida suoranaisesti luokitella nomoteettiseksi, sillä se merkitsisi positivismin vahvaa läsnäoloa ja toisaalta empirian näkökulmasta suurta havaintomäärää tilastollisten yleistysten mahdollistamiseksi (Salmi & Järvenpää 2000).

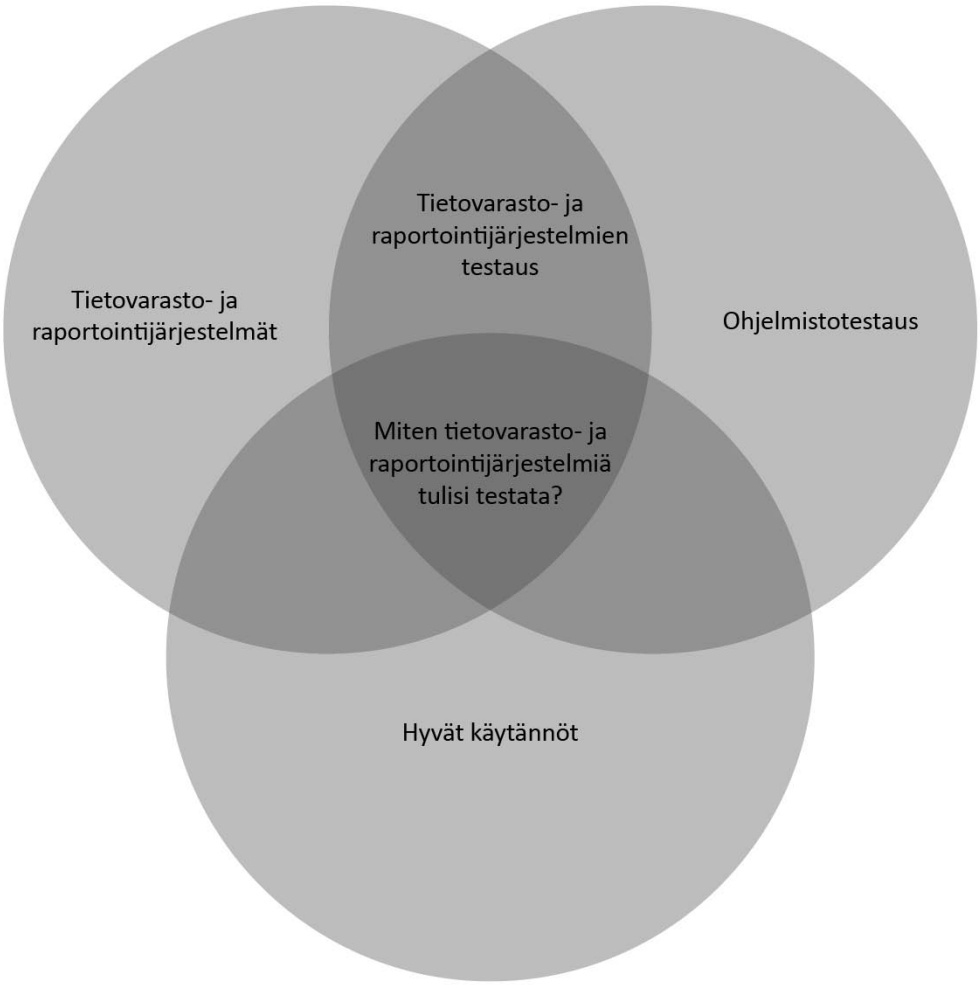
Tämän tutkimuksen ensisijaisena tavoitteena on tutkimusongelmaan liittyvien ilmiöiden ymmärtäminen ja helposti ymmärrettävän ohjesäännöstön kehittäminen. Lisäksi tälle tutkimukselle on ominaista tutkimustulosten tulkitseminen sekä empiiristen kyselyiden

tai haastatteluiden pohjalta että teorian näkökulmasta. Näin ollen tutkimukseen sisältyy eri tasoista tulkintaa ja sen voidaan katsoa noudattavan toiminta-analyyttistä tutkimusotetta. Toiminta-analyyttisen tutkimusotteen tarkoituksena ei ole yleistettävissä olevien lainalaisuuksien etsiminen ja löytäminen, vaan käsitteiden ja niiden tulkinnan kehittäminen, jonka avulla voidaan ymmärtää tutkittavan ilmiön toimintaa. Ilmiötä tutkittaessa on otettava huomioon, että toiminta-analyyttisella tutkimusotteella on oma sisältörakenteensa, jolloin empirian pohjalla esiintyy vaihtelua käsityksissä ja tulkinnoissa. (Neilimo & Näsi 1980, s. 35) Tutkimusmenetelmää valittaessa kiinnitetään huomiota siihen, kuinka selkeä tutkittava alue on, ovatko vastaukset odotusarvoisesti yksikäsitteisiä vai monitahoisia, tarvitaanko vastauksiin selvennystä ja halutaanko syvällisiä vastauksia. (Hirsjärvi & Hurme 2008, s. 35) Tietovarasto- ja raportointijärjestelmien testaukseen ja sen suunnitteluun ei ole olemassa yksikäsitteisiä vastauksia, vaan ratkaisuvaihtoehdot riippuvat monista eri tekijöistä. Koska empirian tiedonlähteenä käytetään pientä joukkoa asiantuntijoita, olisi pelkkien kyselyiden teettäminen riskialtis vaihtoehto. Lisäksi kyselyt edellyttävät yksiselitteisyyttä. Näin ollen haastattelu on tutkimusmenetelmänä parempi vaihtoehto, sillä sen avulla on mahdollista pureutua syvällisesti moniselitteisten ongelmien pohdintaan.

Haastattelu voi olla kysymysten jäsentelyn mukaan joko strukturoitu, puolistrukturoitu tai strukturoimaton eli avoin haastattelu. Strukturoidussa haastattelussa täysin jäsennellyt kysymykset esitetään määrätyssä järjestyksessä ja vastausvaihtoehdot ovat selkeästi esitetty. (Hirsjärvi & Hurme 2008, s. 44-45) Ennalta määrätyt vastausvaihtoehdot tekevät haastattelusta vertailukelpoisen ja helposti analysoitavan mutta jättävät vastaukset melko pinnalliselle tasolle. Näin ollen vastausten löytäminen voi olla haasteellista. Tällöin haastattelijan oma näkökulma saattaa vaikuttaa vastausten tuloksiin, kun haastateltava ei osaa antaa riittävän yksiselitteistä vastausta. Puolistrukturoidussa haastattelussa eli teemahaastattelussa edetään ennalta määrättyjen teemojen mukaan, jolloin kysymykset voidaan esittää joko strukturoidussa tai strukturoimattomassa muodossa järjestyksestä välittämättä. Tämä antaa haastateltavalle enemmän tilaa jäsennellä vastausta ja saada äänensä kuuluviin. Puolistrukturoitu haastattelu antaa haastattelijalle mahdollisuuden ohjata haastattelun kulkua paremmin kuin avoin haastattelu. Tämän lisäksi puolistrukturoidussa haastattelussa voidaan pureutua syvällisemmin tutkimuskohteen kannalta oleellisiin teemoihin, ja samalla se antaa mahdollisuuden tarkastella varsinaista tutkimusongelmaa laaja-alaisemmin. (Hirsjärvi & Hurme 2008, s. 44-47.) Strukturoimattomassa haastattelussa haastateltava vastaa avoimiin kysymyksiin haluamallaan tavalla. Avointen kysymysten avulla pyritään luomaan haastateltavan kokemuksia kertaamalla yhtenäinen, kokemusperäinen näkemys tutkimuskohteesta. Tällöin haastattelu noudattaa perinteisen keskustelun kaavaa, jossa edetään vähitellen kysymyksestä toiseen. (Hirsjärvi & Hurme 2008, s. 44-45.) Strukturoimattomalla haastattelulla saadaan selvitettyä moniulotteisia ja –selitteisiä ongelmia ja voidaan löytää sellaisia näkökulmia tutkimuskohteeseen liittyen, joita haastattelija ei ole välttämättä ajatellut lainkaan. Vastausten analysointi on kuitenkin hankalaa, koska vastaus voi olla tulkittavissa eri tavoin. Lisäksi

vastauksia voi olla vaikea jäsentää, eivätkä ne ole välttämättä ole vertailukelpoisia. Tutkimuksen kannalta oleellinen asia saattaa jopa puuttua, jos kysymykset on tulkittu jo alunperin väärin. Tämän tutkimuksen empiriaosuudessa hyödynnetään teemahaastatteluja, joiden luonteeseen palataan myöhemmin luvussa 5.

1.4 Tutkimuksen teemat ja rakenne

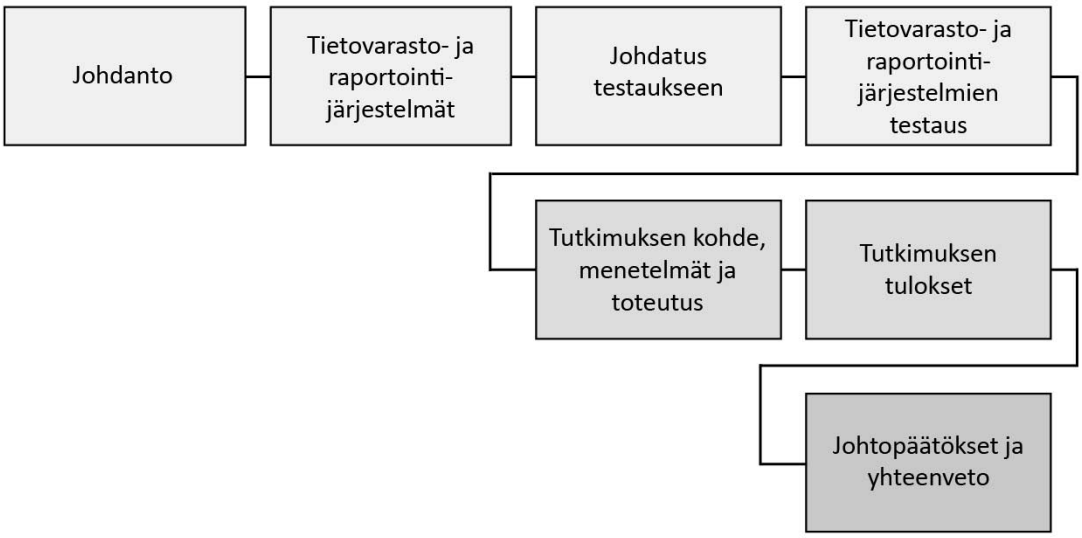


Kuva 1.2. Tutkimuksen teemat

Tutkimuksen olemusta voidaan havainnollistaa kuvan 1.2 mukaisesti jakamalla se kolmeen eri teemaan, joita ovat tietovarasto- ja raportointijärjestelmät, ohjelmistotestaus sekä hyvät käytännöt. Näiden kolmen teeman pohjalta muodostetaan tutkimuksen neljäs ja olennaisin teema eli tietovarasto- ja raportointijärjestelmien testauksen hyvät käytännöt.

Tutkimuksen rakenne (kuva 1.3) muodostuu kolmesta osiosta: teoreettisesta ja empiirisestä tutkimusosiosta sekä johtopäätöksistä. Teoriaosiossa esitetään tieteellisestä kirjallisuudesta, artikkeleista ja tutkimuksista poimitut tulokset, empiriaosiossa teemahaastat-

teluiden avulla aikaansaadut empiiriset tulokset ja johtopäätöksissä koostetut, analyyttiset tulokset. Nämä osiot sisältävät eri vaiheita ja kussakin vaiheessa käsitellään tiettyä tutkimuksen kannalta olennaista teemaa. Kuvassa 1.3 havainnollistetaan tutkimuksen eri osien ja vaiheiden jako sekä tutkimuksen etenemisjärjestys.



Kuva 1.3. Tutkimuksen rakenne

Ensimmäisessä luvussa eli johdannossa esitellään lyhyesti tutkimuksen taustalla oleva ilmiö, määritetään tutkimusongelma ja tutkimuksen tavoitteet, tehdään tarvittavat rajaukset ja otetaan kantaa tutkimusmetodologiaan sekä teoreettisen että empiirisen tutkimusosion osalta. Teoreettinen tutkimusosio sisältää yhteensä kolme lukua:

- tietovarasto- ja raportointijärjestelmät,
- johdatus testaukseen,
- tietovarasto- ja raportointijärjestelmien testaus.

Tutkimuksen toisessa luvussa käsitellään tietovarasto- ja raportointijärjestelmien perusteita ja määritetään niihin liittyvät käsitteet. Vastaavasti tutkimuksen kolmannessa luvussa tarkastellaan ohjelmistotestauksen periaatteita ja määritetään tämän tutkimuksen kannalta olennaiset käsitteet. Näiden teorialukujen tarkoitus on antaa riittävät lähtökohdat tutkimuksen varsinaisen tutkimusongelman käsittelyyn sekä teoreettisesta että empiirisesta näkökulmasta. Neljännessä luvussa esitetään teorioiden ja tieteellisten tutkimustulosten pohjalta yhtenäinen, kokonaisvaltainen viitekehys tietovarasto- ja raportointijärjestelmien testauksen hyvistä käytännöistä. Viitekehys on kokoelma selkeästi jäsenneltyjä, suositeltavia käytäntöjä, jotka liitetään tietovarasto- ja raportointijärjestelmien arkkitehtuuriin ja kehityselinkaareen. Viides luku on empiirisen tutkimusosion ensimmäinen luku, jossa tehdään lyhyt katsaus tutkimuksen toimeksiantajaan eli kohdeyritykseen, käytettyyn haastattelumenetelmään sekä aineiston analyysimenetelmään.

Kuudennessa luvussa esitetään jäsennelty ja analysoitu haastatteluaineisto ja tutkimuksen empiiriset tulokset. Seitsemännessä luvussa tehdään tutkimuksen johtopäätökset ja reflektoidaan teoreettista tulosaineistoa empiirisiin tuloksiin. Tässä luvussa pyritään korostamaan teorian ja empirian keskinäistä vuoropuhelua soveltuvilta osin, ja muodostetaan vertailun avulla lyhyt yhteenveto vastaukseksi tutkimuskysymyksiin ja lopulta tutkimusongelmaan. Lopuksi arvioidaan tutkimusta ja analysoidaan jatkotutkimusmahdollisuuksia.

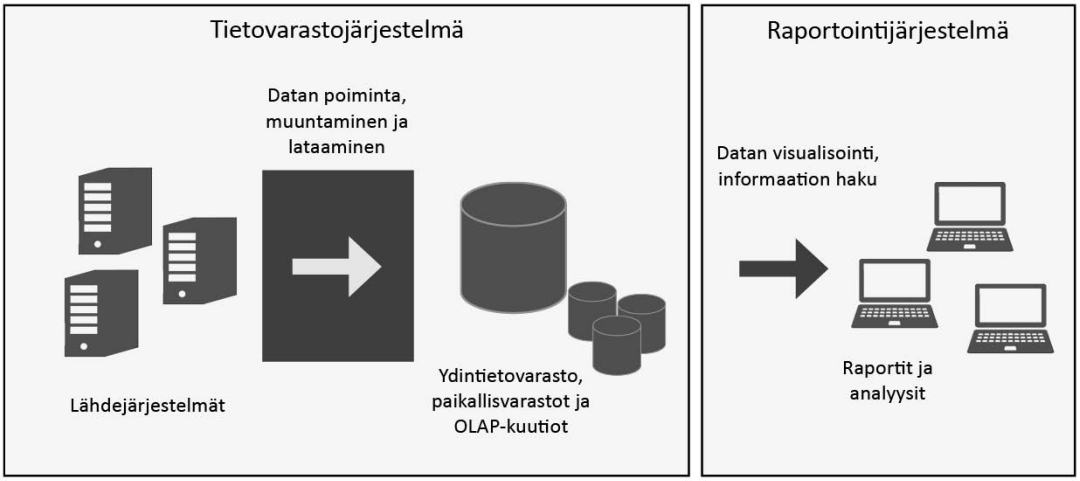
2 TIETOVARASTO- JA RAPORTOINTIJÄRJESTELMÄT

2.1 Johdatus tietovarastointiin ja raportointiin

Tiedon merkityksen kasvu ja teknologioiden kehitys on vaikuttanut olennaisesti liiketoiminnan tarvitseman tiedon luonteeseen ja määrään, mikä on johtanut toiminnan tietointensiivisyyteen. Tiedosta on muodostunut olennainen kilpailutekijä ja liiketoiminnan kehityksen ajuri. Liiketoiminta-analytiikan hyödyntäminen, liiketoiminnan johtaminen tiedolla ja tulevaisuuden ennakoiminen edellyttävät kuitenkin strategista informaatiota, jota on käytännössä mahdotonta tuottaa pelkästään operatiivisissa järjestelmissä (Ponniah 2010, s. 12-13). Jo pitkään yrityksiä vaivannut toistuva ongelma oikean, täsmällisen, ajantasaisen ja integroidun informaation puutteesta on ratkennut, sillä strategisesti tärkeää informaatiota voidaan tuottaa tehokkaasti hyödyntämällä tietovarasto- ja raportointijärjestelmää (Ariyachandra & Watson 2010). Sitä ei ole suunniteltu operatiivista toimintaa varten, vaan palvelemaan ensisijaisesti tietotyöläisten tarpeita ja valjastamaan olemassa olevan tiedon potentiaali liiketoiminnan päätöksenteossa, suunnittelussa ja ennakoinnissa (Ponniah 2010, s. 12-13; Velicanu & Matei 2007).

Tietovarasto- ja raportointijärjestelmän tavoitteet ja vaikutukset yritykselle on selvitettävä ennen kuin varsinaista kehitystyötä ryhdytään tekemään. On luonnollista, että tavoitteissa, vaikutuksissa ja tarpeissa esiintyy vaihtelua eri yritysten välillä, sillä kukin yritys toimii eri tavalla. Esimerkiksi missio, strategiset tavoitteet, toimialan ominaispiirteet, liiketoimintaprosessit, toiminnan luonne, tuotteet ja palvelut, asiakkaiden odotukset, kilpailutilanne ja muut vastaavat tekijät vaihtelevat yrityksittäin. Tietovarasto- ja raportointijärjestelmän perimmäisenä tarkoituksena on luoda yrityksessä päätöksiä tekeville henkilöille ja tietotyöläisille mahdollisuus tehdä liiketoiminnan kannalta oikeita, tarkoituksenmukaisia ja järkeviä päätöksiä tehokkaasti sekä tehdä luotettavia ennusteita tulevaisuuteen (Ponniah 2010). Toisaalta sen perimmäinen tarkoitus on tuottaa liiketoiminnan kannalta arvokasta informaatiota, mikä johtaa parempaan ymmärrykseen asiakkaiden tarpeista ja käyttäytymisestä, asiakkaille tarjotuista tuotteista ja palveluista sekä yrityksestä ja sen toiminnasta itsestään (Velicanu & Matei 2007). Tietovarasto- ja raportointijärjestelmä ei kuitenkaan ole ainoastaan tietojärjestelmä, vaan laajemmin informaatioympäristö, johon liittyy määrällisesti useita sovelluksia, teknologioita ja toimintatapoja. Tietovarastointia ja raportointia kehitetäänkin tyypillisesti yrityksen tiedon tai informaation hallintaan sekä yhteisen liiketoimintanäkymän tuottamiseen ja jakamiseen. Toisaalta tavoitteet tai tarve tietovarastoinnille ja raportoinnille voidaan yksinkertaistaa

seuraavasti: operatiivisen, liiketoiminnan kannalta relevantin raakatiedon eli datan poimiminen eri tietolähteistä ja jalostaminen analyyttiseksi informaatioksi liiketoiminnan hyödynnettävissä olevaan muotoon keskitetysti kaikkien sitä tarvitsevien saataville. Tämä on tietovarasto- ja raportointijärjestelmän keskeisin toimintaperiaate. Kuvassa 2.1 on havainnollistettu tämä toimintaperiaate yksinkertaisella tavalla ja siitä voidaan erottaa kaksi toisistaan eroavaa ympäristöä tiedon jalostusketjuineen aina datasta tietämykseksi. Nämä ympäristöt liittyvät kuitenkin toisiinsa erittäin läheisesti, eikä niitä käytännössä voida erottaa toisistaan. Raportointi viittaa tässä yhteydessä kaikkiin mahdollisiin sovelluksiin, joita tietovarastoinnin avulla voidaan toteuttaa.



Kuva 2.1. Tietovarastointi- ja raportointiympäristö (mukailtu lähteestä Ponniah 2010, s. 19; Tuomi 1999)

Tavoitteita voidaan lähestyä myös liiketoiminnallisten hyötyjen ja tarpeiden näkökulmasta. Tällöin lähtökohtana on liiketoiminnan tarpeet, joihin uudella tietojärjestelmällä pyritään vastaamaan. Tietojärjestelmältä edellytetään aineellisia ja aineettomia liiketoimintahyötyjä, joissa on huomioitava myös tietojärjestelmän aiheuttamat kustannukset sen elinkaaren aikana. Watson et al. (2002) ja DeLone & McLean (1992) mukaan tietovarasto- ja raportointijärjestelmällä tavoitellaan hyötyjä ainakin seuraavilla osa-alueilla:

- resurssien johtaminen ja optimointi,
- suorituskyvyn johtaminen tai liiketoiminnan suoritusarvojen mittaaminen,
- päätöksenteko,
- tehokas informaation hyödyntäminen.

Hyötyjen saavuttaminen edellyttää laadukasta dataa ja laadukkaita prosesseja datan jalostamiseksi informaatioksi kaikkien tarvitsijoiden saataville paikasta tai ajasta riippumatta. Kun tällaiset hyödyt saadaan liitettyä osaksi organisaation strategiaa ja tukemaan strategisia tavoitteita, ollaan tietovarastoinnin ja raportoinnin ytimessä (Watson 2008).

Tietovarasto- ja raportointijärjestelmän kehityksessä tulisikin kiinnittää huomiota juuri informaation laatuun ja erityisesti sen tuottamaan arvoon asiakkaalle, mikä on eräs testauksen motivaattoreista.

2.2 Tietovarasto- ja raportointijärjestelmän määritelmä

Kimball et al. (2008, s. 10) mukaan tietovarastointi on kokonaisvaltainen prosessi, jossa yritys kerää tietoa eri lähteistä muuttaen sen informaatioksi, jota voidaan hyödyntää liiketoiminnan päätöksenteossa. Gardner (1997) vertaa sitä vastaavasti prosessiin, jossa hallinnoidaan useista eri tietolähteistä kerättyjä tietoja yleisen näkymän saavuttamiseksi koko liiketoimintaan tai johonkin sen osa-alueisiin. Näin ollen tietovarastointi voidaan käsittää kokoelmana eri teknologioita, ajatusmalleja ja suunnittelumenetelmiä, joiden avulla tietotyöntekijät liiketoiminnan edustajina saavat mahdollisuuden tehdä aiempaa nopeampia ja parempia päätöksiä (Chaudhuri & Dayal 1997; Hovi et al. 2001). Tässä työssä sovelletaan lähinnä Kimball et al. (2008) ja Gardnerin (1997) prosessipohjaista lähestymistapaa tietovarastoinnille.

Usein kirjallisuudessa puhutaan sekä tietovarastoinnista että tietovarastoista. Ero näiden kahden käsitteen välillä on häilyvä jo senkin vuoksi, ettei kummallekaan ole olemassa vain yhtä ainoaa määritelmää. Yksinkertaistettuna tietovarastointi viittaa toimintaan, jossa hyödynnetään tietovarastoja ja niihin liittyvää infrastruktuuria. Tietovarasto on analyyttinen tietokanta, johon on siirretty jalostettua dataa eli informaatiota eri muodoissa eri tietolähteistä ja asetettu se sitä tarvitsevien saataville (Khan 2012, s. 17; Ariyachandra & Watson 2010). Lisäksi tietovarastoinnissa otetaan huomioon tiedon historiallinen näkökulma, mikä tarkoittaa historiatiedon varastoimista ja analysoimista. Tämä ajallinen näkökulma nousee usein esiin tietovarastoa määriteltäessä hieman teknisemmin. Esimerkiksi Inmon (2002, s. 389) määrittelee tietovaraston kokoelmaksi integroituja aiheorientoituneita päätöksenteon tueksi suunniteltuja tietokantoja, joissa aikaulottuvuus on tiedon keskeisimpiä ominaisuuksia.

Tietovaraston sisältämä tieto kerätään tyypillisesti yrityksen liiketoiminnalle kriittisistä operatiivisista järjestelmistä, muunnetaan ja integroidaan yrityksen liiketoiminnan tarpeisiin soveltuvaksi. Tämä tarkoittaa valmiiksi summattuja tietoja ja valmiiksi rakennettuja kyselyitä sekä tiedon organisointia analytiikan ja käytettävyyden tehokkuuden maksimoimiseksi. (Ponniah 2010, s. 23; Kimball & Ross 2002 s. 397-398.) Raportointi ei välttämättä liity tietovarastointiin, mutta tietovarastointi liittyy lähes poikkeuksetta raportointiin. Harding & Yu (1999) ovat esittäneet, että tietovarastointi tarjoaa käyttäjilleen keskitetyn informaatiolähteen, jota voidaan hyödyntää päätöksenteossa ja liiketoiminnan analytiikassa. Tietovarastoinnin eri komponenttien, kuten tietovaraston tai paikallisvarastojen, sisältämä data tai informaatio on merkittävä resurssi tai voimavara yritykselle ja sillä on olennainen rooli muun muassa päätöksenteossa (Velicanu & Matei 2007). Näin ollen raportointi on vahvasti läsnä tietovarastoinnissa ja toisaalta näitä kah-

ta käsitettä on hankala tarkastella toisistaan irrallisina. Ponniah (2010, s. 15) esittää tietovarasto- ja raportointijärjestelmän informaatioympäristönä, joka

- tuottaa yhden kokonaisvaltaisen ja integroidun näkymän yrityksen liiketoimintaan,
- tekee yrityksen nykyisistä ja historiallisista informaatiovaroista helposti saavutettavia strategista päätöksentekoa varten,
- parantaa päätöksentekoa ilman, että tämä aiheuttaa haittaa operatiivisille järjestelmille ja
- toimii joustavana ja interaktiivisena tietolähteenä yrityksen strategiselle informaatiolle.

Yhteenvetona voidaan kuitenkin todeta, että tietovarastointi perustuu ennen kaikkea tiedon keräämiseen ja jalostamiseen raportointia varten. Tietovarasto- ja raportointijärjestelmällä tarkoitetaan tässä työssä järjestelmää,

> *jolla kerätään ja integroidaan tietoa eri tietolähteistä, jalostetaan tiedosta informaatiota muodostamalla muun muassa summainformaatiota ja tarjotaan tämä informaatio erilaisten raportointivälineiden kautta kaikkien sitä tarvitsevien saataville ajasta ja paikasta riippumatta.*

Tämän prosessin aikana kaikki tieto tallennetaan tietovarastoon, jolloin sen hallitseminen on keskitettyä ja tehokasta. Näin päätöksentekoa ja analytiikkaa varten saadaan luotettavaa tietoa oikean aikaisesti.

2.3 Tietovarasto- ja raportointijärjestelmien ominaispiirteet

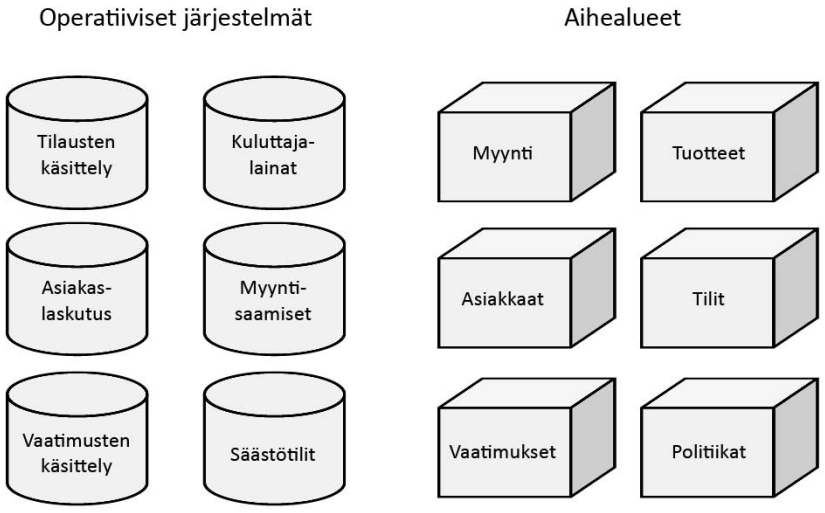
Organisaation tieto voidaan luokitella operatiiviseen ja informatiiviseen tietoon, mikä muodostaa merkittävimmän eron tavallisten tietojärjestelmien ja tietovarasto- ja raportointijärjestelmien välillä (Khan 2012; Hovi 2001; Ponniah 2010). Operatiivisissa järjestelmissä varastoidaan järjestelmäkohtaista, yksilöllistä tietoa, jota tuotetaan ja käytetään kunkin järjestelmän toimialueella. Raportoinnissa tai analytiikassa ei kuitenkaan voida hyödyntää operatiivista tietoa sen ominaisuuksien vuoksi. Operatiivisen ja informatiivisen tiedon keskeiset erot esitetään taulukossa 2.1.

Taulukko 2.1. Operatiivisen ja informatiivisen tiedon keskeiset ominaisuudet (mukailtu lähteistä Khan 2012; Hovi 2001; Ponniah 2010)

Operatiivinen tieto	Informatiivinen tieto
Rutiininomaisten prosessien ja tehtävien tukeminen	Strategisen päätöksentukeminen, toiminnan analysointi ja tulevaisuuden ennakoiminen
Ei aikasidonnaista	Aikasidonnaista
Yksityiskohtaista, usein vaikeaselkoista	Summattua tai koostettua, ymmärrettävää
Tietoa voidaa lisätä, poistaa ja lukea	Tietoa voidaan ainoastaan lukea, vanhoja tietueita ei päivitetä
Sidoksissa liiketoimintaprosesseihin, oltava nopeasti ja aina saatavilla	Saatavuus ei kriittistä liiketoiminnan kannalta
Tieto muuttuu jatkuvasti, reaaliaikaista tietoa	Päivitetään säännöllisesti, mutta olemassa olevaan tietoa ei muuteta
Paljon käyttäjiä	Vähän käyttäjiä

Ponniah (2010, s. 24) on määrittänyt Bill Inmonin (2002) lailla merkittävimmiksi erityispiirteiksi tiedon aihesidonnaisuuden, tiedon aikasidonnaisuuden, tiedon pysyvyyden, tiedon granulariteetin ja tiedon integroituneisuuden.

Tiedon aihesidonnaisuus



Kuva 2.2. Tiedon aihesidonnaisuus tietovarastoinnissa (Ponniah 2010, s. 25)

Tiedon aihesidonnaisuus on yksi tietovarastojen keskeisistä erityispiirteistä. Operatiivinen tieto rakentuu tyypillisesti tietyn toiminnallisuuden tai liiketoimintaprosessin ympärille, mikä voi esimerkiksi tilausjärjestelmän tapauksessa tarkoittaa yksinkertaisesti tila-

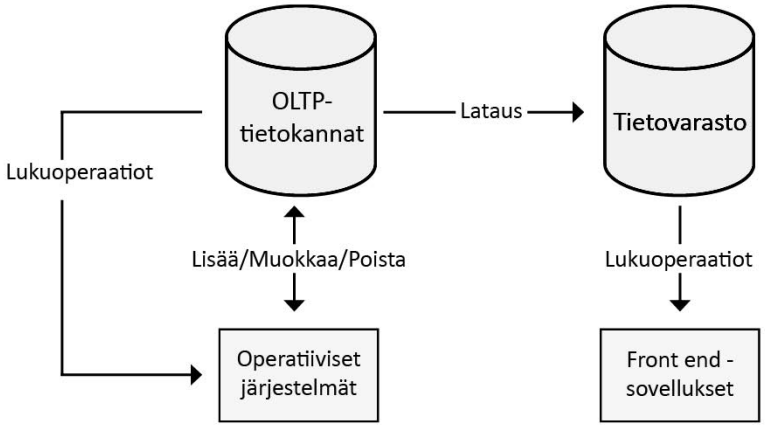
usten käsittelyä ja siihen liittyviä toimintoja. Tämä on tärkeää operatiivisen tiedon kannalta, sillä yksilöityjen tietojoukkojen on tarjottava järjestelmille mahdollisuus suoriutua määritetyistä tehtävistä ja toiminnoista tehokkaasti. (Ponniah, s. 23-25.)

Tietovaraston kohdalla tilanne on kuitenkin toinen, sillä sen lähtökohtana ei ole tehokkuus, vaan informaatio. Tietovarasto- ja raportointijärjestelmä käsittelee tietoa aihesidonnaisesti eli liiketoiminnalle kriittisten eri osa-alueiden mukaisesti, kuten kuvassa 2.2 havainnollistetaan (Ponniah 2010, s. 25). Esimerkiksi valmistavassa teollisuudessa, yritykselle kriittisiä liiketoiminnan osa-alueita ovat yleisesti myynti, tuotanto, toimitus ja varastointi. Tietovarasto- ja raportointijärjestelmä prosessoi ja kokoaa yhteen kuhunkin osa-alueeseen liittyvää tietoa liiketoiminnan hyödynnettäväksi.

Tiedon aikasidonnaisuus

Operatiiviset järjestelmät sisältävät nykyistä ajanhetkeä vastaavaa tietoa, koska niiden tarkoitus on mahdollistaa tai tukea yrityksen päivittäisiä toimintoja ja rutiineja. Tietovarasto- ja raportointijärjestelmän tarkoitusperät ovat kuitenkin erilaiset. Liiketoimintajohtoa kiinnostavien kysymysten kuten esimerkiksi tietyn asiakkaan ostokäyttäytymisen tutkiminen operatiivisten järjestelmien sisältämän nykyhetken tiedon avulla on käytännössä mahdotonta, sillä ostokäyttäytymisen analysointi ja tutkiminen edellyttää historiatietoa asiakkaan aiemmista ostotapahtumista. Tämä toteutetaan tietovarastoinnissa tallentamalla uusia tilannekuvia määrätyin väliajoin. Kaikki tieto sisältää aikaleiman, mikä vastaa jotain tiettyä tilannekuvaa. (Ponniah 2010, s. 26-27.) Tämä liittää menneisyydessä tapahtuneet asiat nykyhetken informaatioon ja mahdollistaa tulevaisuuden ennustamisen.

Tiedon pysyväisyys



Kuva 2.3. Tiedon pysyväisyys (Ponniah 2010, s. 27)

Tietovarastoa ei ole tarkoitettu liiketoiminnan päivittäisten toimintojen suorittamiseen, mikä tarkoittaa, ettei tietovarastoa päivitetä tai poisteta lähdejärjestelmissä tapahtuneiden muutosten mukaisesti. Itseasiassa, tietovarastosta poistetaan vain turhaa tietoa, jonka määritelmä vaihtelee organisaatioittain. Vastaavasti historiatietoa säilytetään määrätyltä aikaväliltä raportteja ja analyyseja varten, eikä sitä päivitetä enää ensimmäisen latauksen jälkeen. (Ponniah 2010, s. 27.) Jos tietoihin tehtäisiin muutoksia, koko tietovarastoinnin idea muodostuisi kyseenalaiseksi.

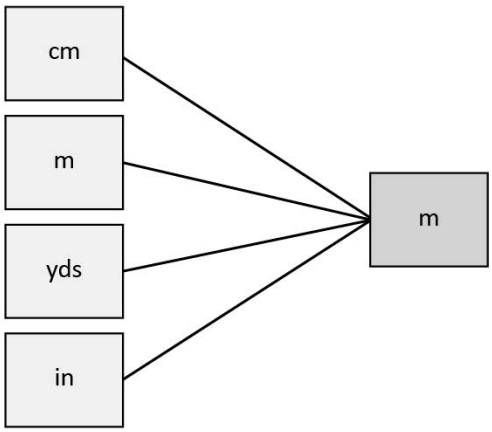
Liiketoiminnan vaatimuksista riippuen lataukset voidaan suorittaa esimerkiksi päivittäin, kerran viikossa tai tarpeen vaatiessa. Tyypillisesti korkeamman tason summatieto päivitetään harvemmin kuin hyvin atomisen tason tieto (Ponniah 2010, s. 27). Kuva 2.3 havainnollistaa eroa operatiivisen järjestelmän ja tietovarasto- ja raportointijärjestelmän välillä, ja sitä, millaisia toimintoja niihin kohdistetaan tiedon näkökulmasta. Käytännössä lisäykset tietovarastoon tehdään keskitetysti, eikä yksittäisiä muutoksia ladata lähdejärjestelmistä erikseen niiden tapahtumahetkellä. (Ponniah 2010, s. 28)

Tiedon granulariteetti

Tiedon granulariteetti ja sen vaihtelu on eräs keskeinen tietovarasto- ja raportointijärjestelmien erityispiirre. Granulariteetilla tarkoitetaan tiedon karkeusastetta eli sen yksityiskohtaisuuden tasoa. Atomisen tiedon granulariteetti on hyvin matala ja se kasvaa, kun tietoa summataan tai koostetaan. Atominen tieto on yksityiskohtaista tietoa, johon ei voida porautua, eikä sitä voida jakaa pienempiin osiin. Vastaavasti summaamisen ja koostamisen ideana on tuottaa informatiivisempaa tietoa (Geiger et al. 1997, s. 35). Tiedon granulariteetti on suunniteltava huolella, sillä liiallinen summaus voi piilottaa olennaisia asioita tai tuoda esiin vääriä asioita. Lisäksi granulariteetti vaikuttaa merkittävästi varastoitavaan tiedon määrään, mikä taas vaikuttaa muun muassa suorituskykyyn. (Ponniah 2010, s. 28.)

Tiedon integroituneisuus

Informatiivinen tieto tai strateginen informaatio integroidaan eri lähteistä, sillä lähteissä oleva data on vain harvoin käyttökelpoista sellaisenaan raportoinnin tai analytiikan tarpeisiin (Ponniah 2010, s. 25). Lisäksi data voi olla eri tyyppistä lähteestä riippuen. Esimerkiksi perusyksikön (engl. base unit of measure) integroiminen useammasta eri lähteestä voi edellyttää useampia erillisiä muunnoksia eli transformaatioita, kuten kuvassa 2.4 havainnollistetaan.

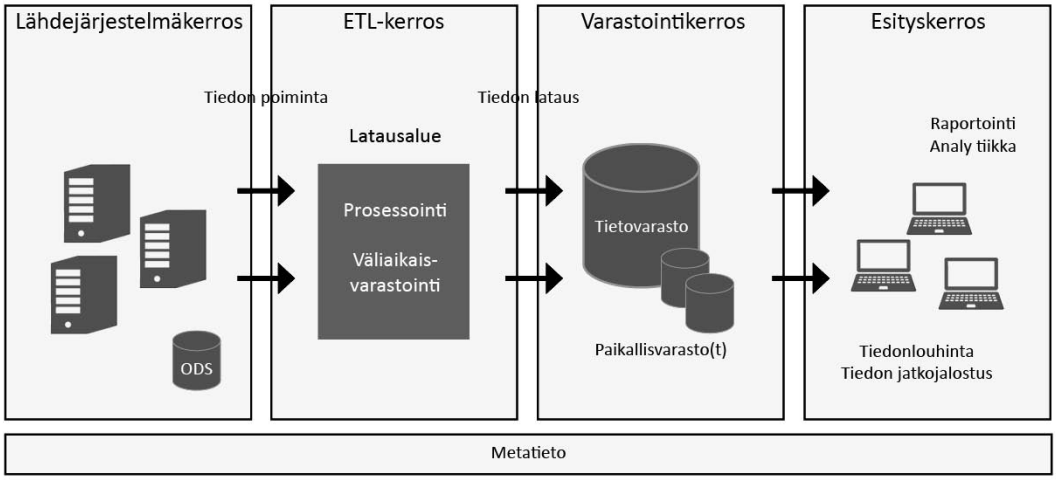


Kuva 2.4. Perusyksikön integroiminen useasta eri lähteestä (mukailtu lähteestä Inmon 2002, s. 84)

Integrointi on välttämätöntä, sillä lähdedata sijaitsee tyypillisesti eri paikoissa. Esimerkiksi asiakastiedot saattavat olla hajautuneena eri järjestelmiin siten, että tilaustiedot löytyvät tilausjärjestelmästä, yhteystiedot asiakkuudenhallintajärjestelmästä ja rahoitustiedot myyntijärjestelmästä. Eri järjestelmät ja sovellukset taas eroavat toisistaan ja käsittelevät tietoa eri tavalla. Esimerkiksi tiedostojen muoto, kirjainkoodien esitysmuoto ja kenttien nimeämiskäytännöt voivat erota merkittävästi toisistaan. Lisäksi joissain tapauksissa on hyödynnettävä organisaation ulkopuolisia tietolähteitä, jolloin epäjohdonmukaisuudet, päällekkäisyydet ja eroavaisuudet muun muassa semantiikassa täytyy eliminoida, mikä edellyttää standardisointityötä esimerkiksi nimeämiskäytäntöjen, koodien ja eri attribuuttien osalta. (Ponniah 2010, s. 25-26) Näin ollen, tietovaraston sisältämä tieto on lähes aina integroitua.

2.4 Järjestelmäkerroksista arkkitehtuurimalleihin

Chaudhuri & Dayal (1997), Ponniah (2010) sekä Kimball & Ross (2002) jakavat tietovarasto- ja raportointijärjestelmän neljään eri alueeseen tai kerrokseen, joita ovat lähdejärjestelmät, latausalue, tiedon varastointi- ja esitysalue sekä informaatiotyökalut. Tietovirta kulkee lähdejärjestelmistä, latausalueen kautta tiedon varastointi- ja esitysalueelle ja lopulta aina raporteille saakka. Lisäksi ETL-kerrokseen liittyvien tietovirtojen tapauksessa voi esiintyä tietovirtoja molempiin suuntiin, jos esimerkiksi operatiivisiin järjestelmiin tehdään muutoksia. Kuva 2.5 havainnollistaa eri järjestelmäkerrosten sijoittumista toistensa suhteen ja tietovirtoja näiden välillä.



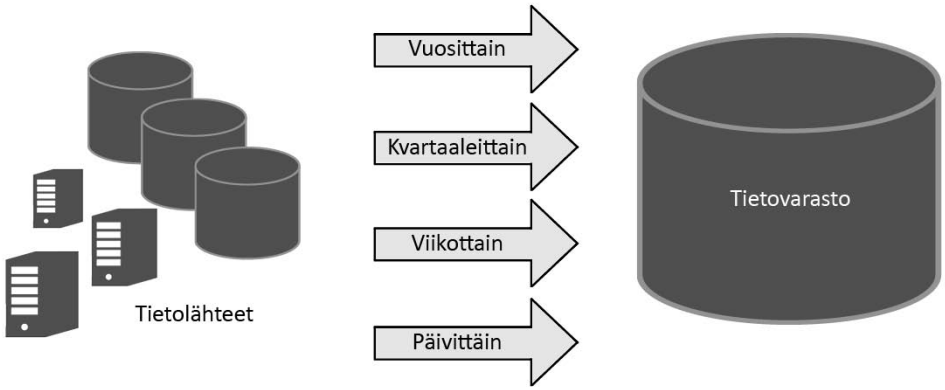
Kuva 2.5. Tietovarasto- ja raportointijärjestelmän eri kerrokset (mukailtu lähteistä Ponniah 2010, s. 34; Chaudhuri & Dayal 1997)

Lähdejärjestelmät, jotka tuottavat operatiivisten liiketoimintapahtumien seurauksena dataa myöhemmin tietovarastoinnin ja raportoinnin hyödynnettäviksi, muodostavat tietovarasto- ja raportointijärjestelmän näkökulmasta ulkoisen järjestelmäkerroksen (Kimball & Ross 2002; Ponniah 2010). Lähdejärjestelmä voi olla organisaation sisäinen tai ulkoinen tietolähde tai rekisteri. Ponniah (2010) luokitelee nämä tietolähteet neljään kategoriaan tiedon luonteen ja hankitatavan mukaan: tuotantotietoon, sisäiseen tietoon, ulkoiseen tietoon ja arkistoituun tietoon. Tuotantotietoa saadaan yrityksen operatiivisista järjestelmistä, kuten talous-, tuotanto- tai asiakasjärjestelmistä. Kaikkea tuotantotietoa ei ole järkevää, eikä hyödyllistä tuoda tietovarastoon, vaan valinta on suoritettava informaatiotarpeiden mukaisesti. Koska tuotantotieto on tyypillisesti lähtöisin useista eri tietojärjestelmistä tai –kannoista, sen ominaisuudet, luonne ja semantiikka voivat vaihdella huomattavissa määrin lähteestä riippuen. Esimerkiksi *asiakas* käsitteenä voi saada erilaisen merkityksen eri järjestelmissä, mikä on otettava huomioon kun kyseiseen käsitteeseen liittyvää tietoa viedään tietovarastoon. Vastaavasti, sisäisellä tiedolla voidaan tarkoittaa mitä tahansa organisaation tietojärjestelmiin tai -kantoihin varastoitua tietoa, jota tuotetaan organisaation sisäisten prosessien tai tietojärjestelmien avulla. Informaatiotarpeiden kasvaessa organisaatio saattaa tulla riippuvaiseksi myös ulkoisesta tiedosta, jota ei saada organisaation omista tietojärjestelmistä tai –kannoista. Toisaalta ulkoinen tieto harvoin vastaa suoraan organisaation standardeja, joten se on ensin muunnettava ja integroitava sopivaan muotoon. Arkistotieto on historiatietoa, jota on talletettu eri tavoin ja aikajäntein esimerkiksi operatiivisten järjestelmien varmistusnauhoille, tietokantoihin tai kiintolevyille. (Ponniah 2010, s. 34-37)

Tiedot lähdejärjestelmistä poimitaan ensimmäiseksi tietojen latausalueelle (engl. staging area). Poiminta on osa ETL-prosessia (engl. Extract, Transform, Load), joka on latausalueen keskeisin toiminto. ETL-prosessi koostuu joukosta ETL-proseduureja, joiden

päämääränä on poimia tarvittava tieto lähdejärjestelmistä, muuntaa se tietovaraston edellyttämään muotoon ja ladata edelleen tiedon varastointi- ja esitysalueelle esimerkiksi ydintietovarastoon. Tämä kokonaisprosessi sisältää erilaisia tehtäviä, kuten tiedon puhdistamista ja puuttuvien arvojen käsittelyä, tiedon yhdistelyä ja yhtenäistämistä, indeksointia, tietojen summausta ja koontia, duplikaattien eliminointia, keinotekoisten avaimien lisäämistä ja vakioarvojen asettamista (Ponniah 2010, s. 37-38). Varsinkin tietojen yhtenäistäminen eli standardoiminen on tärkeää, sillä tietotyypit, kenttien pituudet ja merkitykset vaihtelevat. Tässä kohdin tunnistetaan mahdollisia synonyymeja eli samaa merkitseviä objekteja, joiden tarkoitusperässä esiintyy samankaltaisuuksia. Toisaalta on varottava homonyymeja eli termejä ja käsitteitä, jotka muistuttavat toisiaan eri lähdejärjestelmissä, mutta joiden merkitys on olennaisesti erilainen. Keinotekoisten avainten luominen on myös välttämätöntä ETL-prosessin yhteydessä, sillä tyypillisesti lähdejärjestelmien luonnollisia avaimia ei voida hyödyntää tietovarastossa. (Ponniah 2010, s. 38.) Tämä johtuu siitä, että luonnollisilla avaimilla on usein jokin sisäänrakennettu merkitys, jota ei voida yhtenäistää kaikkien eri lähdejärjestelmien kesken.

Alkulatauksen lisäksi tietovarastoa päivitetään määrätyin väliajoin (kuva 2.6), mikä pitää sisällään tietojen lisäämisen sekä tarpeettomien tietojen poistamisen tietovarastosta (Chaudhuri & Dayal 1997; Ponniah 2010, s. 39). Tällaiset inkrementaalisesti suoritettavat, eräajopohjaiset lataus- tai päivitysoperaatiot ovat volyymiltaan pienempiä kuin massalataukset, joissa ladataan suuria määriä tietoa tietovarastoon. Massalatauksista käytetään myös nimitystä alkulataus. Lataus- tai päivitysoperaatioita tehdään informaatiotarpeiden muuttuessa tai kun tietovarastossa havaitaan jokin virhe tai puute. Jos informaatiotarpeet muuttuvat oleellisesti, voi myös lataus- tai päivitysoperaatioiden koko kasvaa huomattavan suureksi. ETL-prosessissa on otettava huomioon, että myös pienemmät lataukset vievät aikaa, jolloin lataus- ja päivitystiheys on suunniteltava huolellisesti siten, että sillä saadaan aikaan paras mahdollinen hyöty liiketoiminnan kannalta. (Ponniah 2010, s. 39.)



Kuva 2.6. Tietojen lataus ja päivittäminen (Ponniah 2010, s. 39)

Latausalue toimii ETL-prosessin alustana ja sisältää työkaluja sekä infrastruktuurin tiedon prosessoimiseen että väliaikaiseen varastoimiseen. Latausaluetta tarvitaan, koska dataa ei voida suoraan siirtää lähdejärjestelmistä tietovarastoon siinäkään tapauksessa, että tietolähteitä olisi vain yksi. (Ponniah 2010, s. 37-38.)

Tiedon säilytys- ja esitysalueella varastoidaan ja hallinnoidaan informaatiotarpeita vastaavaa tietoa. Tämä tapahtuu joko tietovarastossa (engl. data warehouse, DW), paikallisvarastoissa (engl. data mart, DM), operatiivisissa tietovarastoissa (engl. operational data store ODS) tai multidimensionaalisissa OLAP-kuutioissa. Paikallisvarastot ovat tietovarastosta johdettuja tiettyyn liiketoiminta-alueeseen tai toimintoon keskittyneitä loogisia tietovaraston alijoukkoja, jotka ovat usein suoraan loppukäyttäjien käytettävissä. Tietovaraston ja paikallisvarastojen keskeisimmät erot liittyvät niiden sisältämään tietoon, käyttötarkoitukseen ja –laajuuteen sekä tietomalliin. (Ponniah 2010, s. 39-40.)

Tietovarastossa data on melko pysyvää ja datan päivittäminen tehdään hallitusti ennalta määritetyin ehdoin. Vastaavasti operatiivissa järjestelmissä on tarpeen tehdä nopeita, yksityiskohtaisia hakuja ja toisaalta data voi muuttua odottamattomasti tilanteesta riippuen. Näin ollen tietovarasto ei sovellu operatiiviseen raportointiin, vaan sen fokus on historiatiedon analysoimisessa ja jalostamisessa tietämykseksi strategista päätöksentekoa varten. Toisaalta operatiiviseen raportointiin voidaan käyttää operatiivista tietovarastoa (ODS). Sen toimintaperiaate on samankaltainen tietovaraston kanssa, mutta sen fokus on taktisen päätöksenteon tukemisessa. Tietojen päivityssykli on nopeampi kuin tietovarastossa ja informaatiotarpeet erilaisia. Myös operatiivisten tietovarastojen yhteydessä tarvitaan ETL-prosessia poimimaan, muuntamaan ja lataamaan tiedot lähdejärjestelmistä kohdejärjestelmään. (Kimball & Ross 2002, s. 15.) Tietovarastoinnin ja raportoinnin merkityksen ja kokemusten kasvusta huolimatta on usein epäselvää, millaisen arkkitehtuurimallin pohjalta tietovarasto- ja raportointijärjestelmää ryhdyttäisiin kehittämään, ettei järjestelmän elinkaaren aikana törmättäisi ongelmiin laajennettavuudessa, suorituskyvyssä tai informaation laadussa (Ariyachandra & Watson 2010). Tietovarasto- ja raportointijärjestelmien arkkitehtuurimalleja eli tapoja toteuttaa tietovarastointia ja järjestää tiedon varastointi- ja esitysalueen sisältämiä komponentteja on lukuisia. Yksinkertaistamalla ja etsimällä yhtymäkohtia eri arkkitehtuurimalleista, Ponniah (2010, s. 32) sekä Ariyachandra & Watson (2010) esittävät viisi tyypillistä arkkitehtuurimallia:

- riippumattomat paikallisvarastot,
- paikallisvarastoväylä,
- hub-and-spoke (EDW),
- keskitetty tietovarasto,
- federoitu arkkitehtuuri.

Käytännössä yksinkertaisin tapa tietovarastoinnin ja raportoinnin organisointiin yrityksessä on käyttää toisistaan riippumattomia paikallisvarastoja. Tällaisessa arkkitehtuurimallissa rakennetaan ja kehitetään paikallisvarastoja palvelemaan tiettyjen yksiköiden, toimintojen tai prosessien informaatiotarpeita, minkä seurauksena informaatioympäristöstä muodostuu siiloutunut kokonaisuus, eivätkä tietotyöläiset pääse käsiksi yritystason liiketoiminta näkymään. Paikallisvarastojen riippumattomuus johtaa väistämättä datan ja informaation epäjohdonmukaisuuksiin ja eroihin semantiikassa sekä standardeissa. (Ariyachandra & Watson 2010; Ponniah 2010, s. 32.) Riippumattomia paikallisvarastoja parempi tapa organisoida yrityksen tietovarastointi ja raportointi on eräänlaisen paikallisvarastoväylän hyödyntäminen. Tämä on alunperin Ralph Kimballin esittämä lähestymistapa, jossa kehitystyö aloitetaan tunnistamalla liiketoimintaprosessien informaatiotarpeet ja ensimmäinen paikallisvarasto rakennetaan hyödyntäen liiketoiminnasta johdettuja dimensioita ja faktoja. Näitä dimensioita ja faktoja hyödynnetään myöhemmin rakennettavissa paikallisvarastoissa liiketoiminnan tarpeiden mukaisesti. Lopulta paikallisvarastoista muodostuu eräänlainen väylä, joka toimii loogisesti integroituna, yhtenä suurena paikallisvarastona tarjoten yritystason näkymän liiketoimintaan. (Ariyachandra & Watson 2010; Ponniah 2010, s. 33) Näin ollen tietovarastointia kehitetään ja infrastruktuuria laajennetaan tarpeiden mukaan kehitysprosessin lähtiessä liikkeelle alemmilta organisaatiotasoilta.

Bill Inmon, Ralph Kimballin ohella toinen tietovarastoinnin pioneereista, on esittänyt arkkitehtuurimallin, joka perustuu keskitetysti toteutettuun tietovarastoon ja siitä johdettuihin paikallisvarastoihin. Tällaista arkkitehtuurimallia kutsutaan myös EDW-arkkitehtuuriksi (*engl. Enterprise Data Warehouse*). (Ariyachandra & Watson 2010.) Siinä johdetaan dataa yhdestä keskitetystä tietovarastosta pienempiin paikallisvarastoihin, jotka on kehitetty jotain tiettyä tarvetta, toimintoa, prosessia tai käyttäjäryhmää varten. Suurin osa kyselyistä kohdistuu näihin paikallisvarastoihin, vaikka todellisuudessa hyödynnettäisiinkin keskitettyä tietovarastoa. Toisin kuin Kimballin esittämässä arkkitehtuurissa, tässä lähdetään liikkeelle ylemmiltä organisaatiotasoilta. (Ponniah 2010, s. 33.) Keskitetty arkkitehtuuri on vastaavasti EDW-arkkitehtuurin kaltainen, mutta siinä ei lähtökohtaisesti hyödynnetä lainkaan paikallisvarastoja. Informaatiotarpeet otetaan kuitenkin huomioon organisaation laajuisesti ja eri yksiköt, toiminnot, prosessit tai käyttäjäryhmät hyödyntävät yhtä ja samaa tietovarastoa informaatiotarpeisiinsa (Ponniah 2010, s. 32).

Federaatioon perustuvaa tietovarastoinnin arkkitehtuurimallia hyödynnetään usein yrityksissä, joissa on jo olemassa jonkinlainen informaatioympäristön infrastruktuuri ja päätöksenteon tukirakenteet. Näitä ei useinkaan kannata jättää hyödyntämättä, vaan tietovarastointia ja raportointia voidaan kehittää federoidulla tavalla. Se ei pyri muodostamaan yhtä integroitua ratkaisua jo pelkästään poliittisista ja teknisistä syistä, vaan federoidussa ympäristössä data voi olla fyysisesti tai loogisesti integroitua esimerkiksi

jaettujen avaimien, metadatan tai hajautettujen kyselyiden avulla. (Ariyachandra & Watson 2010; Ponniah 2010, s. 33.)

Loppukäyttäjiä lähimpänä ovat informaatiotyökalut eli raportointisovellukset, jotka toimivat järjestelmän käyttöliittymänä ja välittävät käyttäjän tarvitsemaa informaatiota sopivassa muodossa tietovarastosta erilaisiksi raporteiksi tai analyyseiksi. Tietovaraston käyttäjäjoukko on laaja, eikä se ole sidottu tiettyihin organisaation yksiköihin tai liiketoiminta-alueisiin. Lisäksi käyttäjät voidaan luokitella eri ryhmiin informaatiotarpeiden ja osaamistason mukaan. Sellaiset käyttäjät, joille asia on uutta tai jotka tarvitsevat vain harvoin analyyttistä informaatiota, käyttävät usein ennalta määriteltyjä raportteja tai hakuja tietovarastosta. Normaalit käyttäjät tarvitsevat informaatiota silloin tällöin, mutta eivät välttämättä säännöllisesti. Tämän tyyppiset käyttäjät tarvitsevat myös ennalta suunniteltuja raportteja ja ennalta valmisteltuja hakuja. Toisaalta liiketoiminta-analyytikkojen ja tehokäyttäjien kohdalla informaatiotarpeet kasvavat ja monimutkaistuvat, mikä vaatii myös informaatiotyökaluilta kattavia ominaisuuksia. Informaatiotyökalujen ansiosta informaatio saadaan näiden kaikkien käyttäjäryhmien saataville ymmärrettävässä muodossa. (Ponniah 2010, s. 40-41.) Markkinoilla on tarjolla paljon informaatiotyökaluja ja organisaation kannattaa hyödyntää niitä mahdollisimman laajasti informaatiotarpeidensa täyttämiseksi (Ponniah 2010, s. 41).

Metatieto on erillinen tietovarasto- ja raportointijärjestelmän kerros, joka on liitoksissa muihin järjestelmäkerroksiin. Metatieto sisältää tietoa muun muassa tietovaraston tietorakenteista, tiedostoista, indekseistä ja muista olennaisista ominaisuuksista (Ponniah 2010, s. 41). Metatietoa tarvitaan myös latausalueella ja tietovirtojen hallitsemisessa. Vastaavanlaisena toiminnallisuutena voidaan pitää järjestelmän hallintaa, mikä koordinoi palveluiden ja toimintojen suoritusta. Se huolehtii tiedon muunnosprosessista ja tietovirroista sekä ohjaa informaation loppukäyttäjille toimien yhdessä tietokannan hallintajärjestelmien (engl. database management system, DBMS) kanssa. Näin ollen se varmistaa yhdessä metatiedon hallinnan kanssa, että tieto varastoidaan asianmukaisesti komponentista tai toiminnosta riippumatta. (Ponniah 2010, s. 41.)

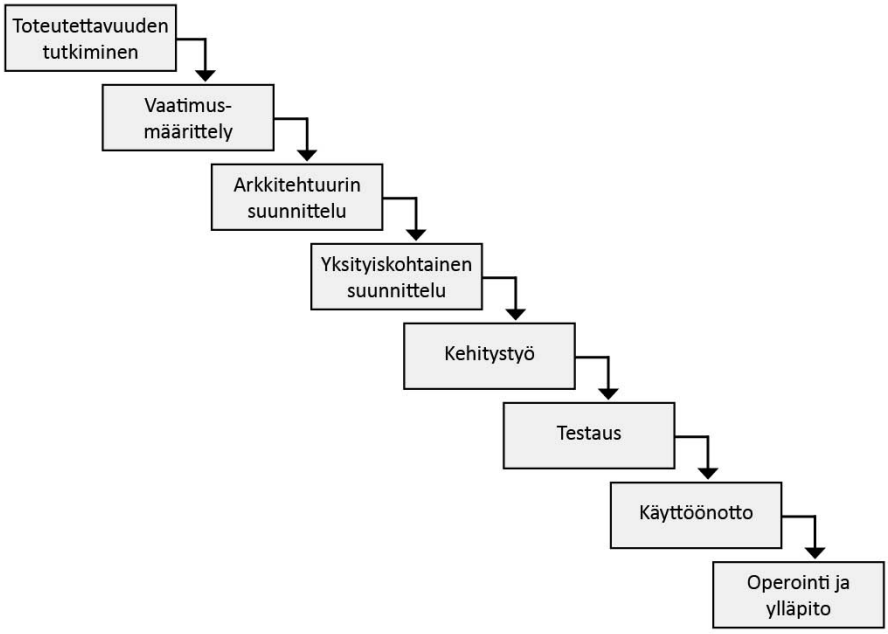
2.5 Tietovarasto- ja raportointijärjestelmien kehittäminen

Velicanun & Matein (2007) mukaan tietovarasto- ja raportointijärjestelmien kehitys noudattaa tyypillisesti joko *top down* tai *bottom up* -lähestymistapaa. Top down on yleinen tilanteissa, joissa liiketoiminnan ongelmat on helposti havaittavissa ja teknologiat tunnetaan hyvin. Tällöin tietovarasto- ja raportointijärjestelmästä pyritään rakentamaan jo alusta alkaen liiketoiminnan eri osia palveleva kokonaisuus. Vastaavasti bottom up on toimintamallina tätä joustavampi, sillä siinä kehitystyötä tehdään pienissä osissa. Kehityksen iteratiivinen luonne alentaa kustannuksia, vähentää riskejä ja järjestelmän hyödyt saadaan realisoitua nopeammin liiketoiminnan käyttöön. Toisaalta, järjestelmän osien integroiminen yhteen saattaa nostaa esiin odottamattomia ongelmia varsinkin ta-

pauksissa, jossa kehitystyötä tehdään paikallisesti pienen projektiorganisaation toimesta. Top down keskittyy tiedon integraatioon ja johdonmukaistamiseen koko organisaation laajuudella, kun taas bottom up lähtee liikkeelle liiketoiminnan vaatimuksista. Usein näitä kahta lähestymistapaa joudutaan yhdistelemään parhaan lopputuloksen saavuttamiseksi. (Velicanu & Matei 2007.)

2.5.1 Tyypillisen prosessimallin kuvaaminen

Rainardin (2008) mukaan tietovarasto- ja raportointijärjestelmien kehittämiseksi on tavallisesti olemassa kaksi toisistaan eroavaa prosessimallia: vesiputousmalli ja iteratiivinen kehitysmalli. Vesiputousmalli on tavanomaisin prosessimalli, joka muodostuu peräkkäisistä vaiheista. Vaiheiden määrä ja sisältö voi vaihdella projektista tai hankkeesta riippuen, mutta vaiheita suoritetaan ennalta määrätyssä järjestyksessä. (Rainardi 2008, s. 49.) Vaiheet esitetään kuvassa 2.6.

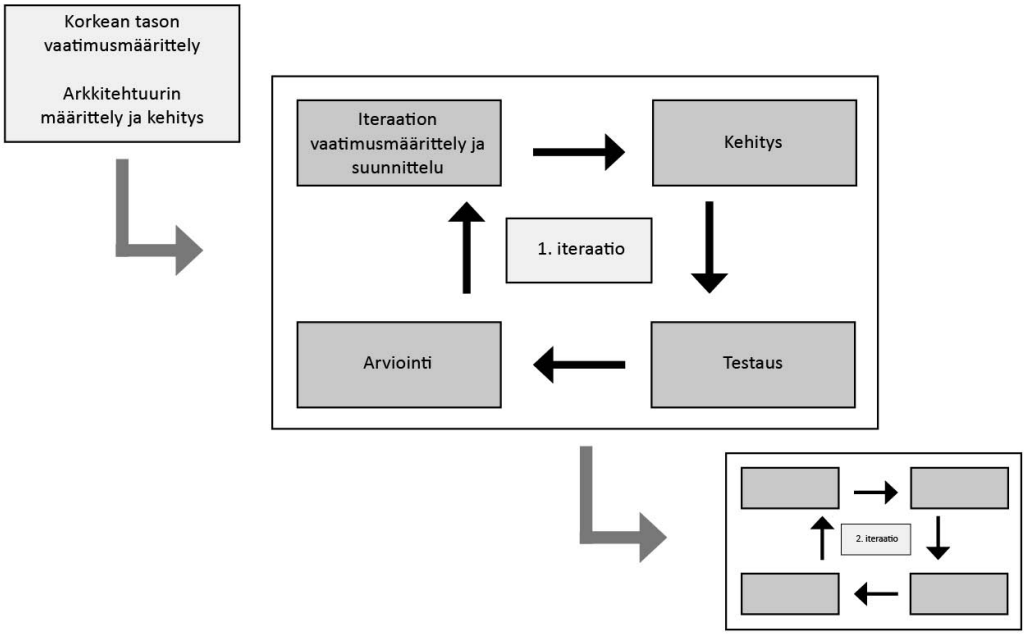


Kuva 2.7. Vesiputousmalli tietovarasto- ja raportointijärjestelmän tapauksessa (Rainardi 2008, s. 50)

Cobbin (2011, s. 170) mukaan vesiputousmallille (kuva 2.7) on ominaista, että valtaosa suunnittelu- ja määrittelytyöstä tehdään ennen varsinaisen kehitystyön aloitusta. Tällöin keskeisenä lähtökohtana on, ettei suunnitelmiin tai määrittelyihin tehtäisi muutoksia myöhemmissä vaiheissa prosessin aikana. Näin ollen, vesiputousmalli sopii ohjelmistokehitykseen, jossa päämäärät ja vaatimukset voidaan määritellä jo projektin alkuvaiheessa hyvin selkeästi ilman suurta muutosalttiutta (Cobb 2011, s. 168). Tämä on kuitenkin erittäin harvinaista tietovarasto- ja raportointijärjestelmien kohdalla. Vesiputousmalli on ongelmallinen myös testauksen näkökulmasta, sillä se ajoitetaan prosessin

loppupäähän, jolloin ongelmiin törmätään ensimmäistä kertaa liian myöhäisessä vaiheessa. Tällainen kerralla valmiiksi lähestymistapa ei yksinkertaisesti toimi tietovarasto- ja raportointijärjestelmiä kehitettäessä, sillä havaitut virheet saattavat vaikuttaa olennaisesti vaatimusmäärittelyyn, suunnitteluun tai esimerkiksi kehitystyöhön (Rainardi 2008, s. 53). Toisaalta, jos toimintaympäristössä vallitsee vain pieni epävarmuus, voi vesiputousmalli olla hyvin tehokas ja helposti hallittavissa oleva malli etenkin resurssien käytön näkökulmasta. Se ei myöskään edellytä muutoksia organisaatiokulttuurissa tai ajatusmalleissa. (Cobb 2011, s. 171.)

Toinen yleisesti käytetty ohjelmistokehityksen prosessimalli on iteratiivinen malli (kuva 2.8), jossa ideana on jakaa kehitysvaiheet pienempiin osiin eli iteraatioihin (Cobb 2011, s. 173; Rainardi 2008, s. 54). Iteratiivisesta kehityksestä on olemassa erilaisia muunnelmia eri tarpeisiin ja se voi sisältää piirteitä vesiputousmallista ja toisaalta muistuttaa ketterää kehitystä. Iteratiivisuus mahdollistaa toiminnallisuuksien ja ominaisuuksien toimittamisen erissä, mikä mahdollistaa järjestelmän nopean kehityksen, vaikka vaatimuksia ei olisi lyöty lukkoon tai ne olisivat epäselviä (Velicanu & Matei 2007). Näin vältytään myös vesiputousmallille tyypillisiltä testausvaiheen ongelmilta (Rainardi 2008, s. 54). Esimerkiksi vaatimuksia voidaan muuttaa testausvaiheessa havaittujen virheiden mukaisesti. Yksityiskohtainen suunnittelu ja toteutus on jaettu helpommin hallittaviin kokonaisuuksiin, jolloin luotettavuuden arvioiminen tai testaus on helpompi sisällyttää prosessiin (Cobb 2011, s. 174; Rainardi 2008, s. 56-57). Toisaalta ensimmäisen julkaisuversion käyttöönotto rajoitetulle käyttäjäjoukolle antaa myös mahdollisuuden testata järjestelmää pidempään oikeassa ympäristössä (Rainardi 2008, s. 57).



Kuva 2.8. Iteratiivinen prosessimalli (mukailtu lähteestä Cobb 2011, s. 175)

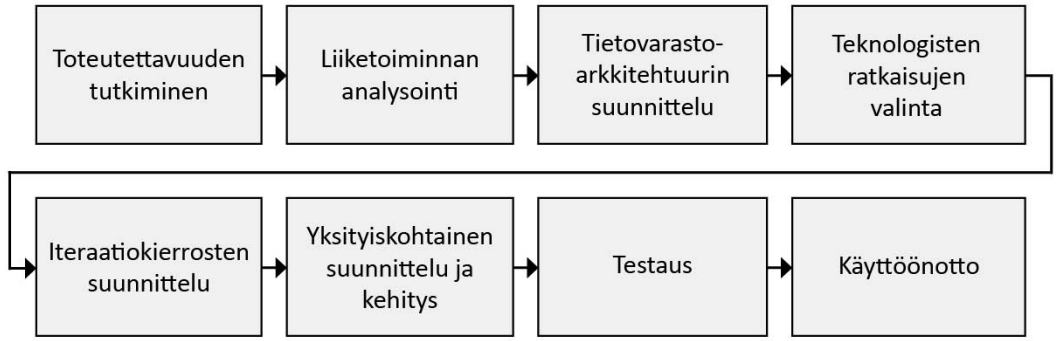
Tietovarasto- ja raportointijärjestelmän kehitykseen iteratiiviset prosessimallit antavat paremmat lähtökohdat jo siitäkin syystä, että tietovarasto- ja raportointijärjestelmien

vaatimukset ovat usein hankalasti määritettävissä projektin tai hankkeen alkuvaiheessa. Tietovarasto- ja raportointijärjestelmän kehitys yhtenä suurena projektinomaisena kokonaisuutena johtaa melko varmasti epäonnistumiseen (Velicanu & Matei 2007). Kun liiketoiminnan osa-alueita lähestytään yksi kerrallaan, fokus on pienissä kokonaisuuksissa, jolloin liiketoimintavaatimusten hallinta on tehokkaampaa ja riskit pienempiä. Todellisuudessa iteratiivisten prosessimallien avulla saavutetaan aineellisia ja aineettomia hyötyjä jo lyhyellä aikavälillä vesiputousmallia alhaisemmin kustannuksin. (Velicanu & Matei 2007.) Se, millaista muunnelmaa iteratiivisesta mallista käytetään, riippuu organisaatiosta ja järjestelmähankkeen tai -projektin lähtökohdista.

2.5.2 Iteratiivisen elinkaarimallin vaiheet

Tietovarasto- ja raportointijärjestelmien tapauksessa kehitys noudattaa useimmiten iteratiivista mallia sisältäen useita kyseisille järjestelmille ominaisia vaiheita tai tehtäväkokonaisuuksia (Kimball et al. 2008, s. 3-4; Velicanu & Matei 2007). Toisaalta, myös muita malleja tai mallien ominaisuuksia voidaan hyödyntää (Black 2009; Velicanu & Matei 2007). Iteratiivisessa mallissa elinkaaren vaiheet irrotetaan loogisista ohjelmistotuotannon aktiviteeteista, kuten suunnittelusta tai toteutuksesta (Leffingwell & Widrig 2000, s. 218-219). Toisin kuin esimerkiksi vesiputousmallissa, nämä vaiheet voidaan toistaa useita kertoja, mikä mahdollistaa esimerkiksi vaatimusmäärittelyn muutokset kesken kehitystyön. Projekti muodostuu iteraatioista, joilla tarkoitetaan aktiviteettien sarjaa ja joissa kussakin keskitytään tiettyjen toiminnallisuuksien aikaansaamiseen (Leffingwell & Widrig 2000, s. 219). Leffingwellin & Widrigin (2000, s. 219) mukaan iteratiivinen malli käsittää neljä keskeistä vaihetta: aloitusvaiheen, valmisteluvaiheen, kehitysvaiheen ja transitiovaiheen.

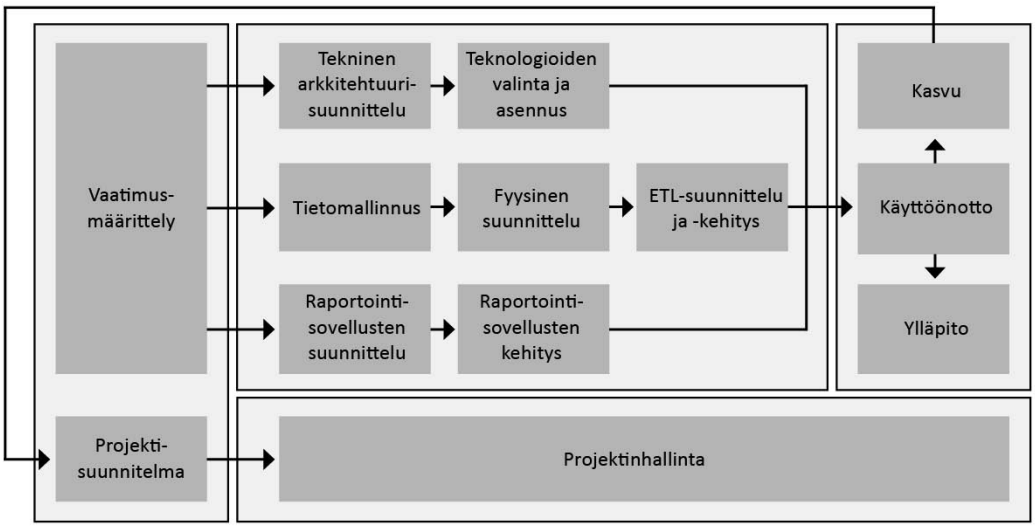
Aloitusvaiheessa keskitytään käytännössä liiketoiminnan vaatimusten, projektin laajuuden, aikataulun, budjetin, riskien ja toteutettavuuden arviointiin (Leffingwell & Widrig 2000, s. 219). Valmisteluvaiheessa liiketoimintavaatimuksia jalostetaan lisää, perustetaan arkkitehtuuri ja kehitetään prototyyppi, jota voidaan demonstroida liiketoiminnan edustajille. Kehitysvaiheessa suoritetaan valtaosa varsinaisesta kehitystyöstä sekä viimeistellään arkkitehtuurin design. Transitiovaihe kuvaa käyttöönottoa ja käyttäjien koulutusta uuden järjestelmän käyttöön. (Leffingwell & Widrig 2000, s. 219.) Näiden vaiheiden sisältö kuitenkin vaihtelee järjestelmästä ja projektista riippuen. Tietovarasto- ja raportointijärjestelmien kohdalla voidaan tunnistaa tiettyjä keskeisiä, projektista toiseen toistuvia aktiviteetteja (Kimball et al. 2008;Velicanu & Matei 2007). Nämä tietovarasto- ja raportointijärjestelmän elinkaaren eri vaiheiden tyypillisimmät aktiviteetit esitetään kuvassa 2.9.



Kuva 2.9. Tietovarasto- ja raportointijärjestelmän elinkaaren eri vaiheiden tyypillisimmät aktiviteetit (mukailtu lähteestä Velicanu & Matei 2007)

Myös Kimball et al. (2008) on esittänyt tietovarasto- ja raportointijärjestelmien elinkaarimallin keskeisimmät aktiviteetit ja kuvannut näiden aktiviteettien organisoitumisen toistensa suhteen (kuva 2.10). Velicanun & Matein (2007) ja Kimball et al. (2008) esittämät mallit ovat hyvin samankaltaisia ja näin ollen täydentävät toisiaan. Elinkaaren ensimmäisessä vaiheessa laaditaan projektisuunnitelma ja arvioidaan projektin toteutettavuutta (Kimball et al. 2008; Velicanu & Matei 2007).

Toteutettavuuden arvioiminen tai tutkimus on käytännössä strateginen analyysi yrityksen valmiuksista hyödyntää informaatiota osana liiketoimintaa ja sisältää siten eri liiketoiminnan osa-alueiden ja osien arviointia (Velicanu & Matei 2007). Puutteellinen valmius voi lisätä projektista aiheutuvia riskejä, jos esimerkiksi järjestelmä osoittautuu käyttöönoton jälkeen hankalasti hyödynnettäväksi liiketoimintaprosessien jäykkyydestä johtuen. Toteutettavuuden arvioinnissa tulisi ottaa huomioon strategisesti tärkeimmät liiketoiminnan osa-alueet, johdon sitoutuminen ja järjestelmän rooli liiketoiminnan suhteen. Tämän tulisi sisältää lähdejärjestelmäanalyyseja ja tiedon profilointia, sillä heikkolaatuinen data voi aiheuttaa merkittäviä ongelmia myöhemmin järjestelmää kehitettäessä (Kimball et al. 2008, s. 3-5). Toisaalta projektisuunnitelmaa tehdessä tulisi määrittää tehtävät, vastuut, roolit, kustannukset, hyödyt sekä kriittiset menestystekijät projektin onnistumisen kannalta. Näiden asioiden pohjalta voidaan laatia asianmukainen projektisuunnitelma ja parantaa kommunikointia sekä ymmärrystä kehitysprosessin tavoitteista kehitystiimin jäsenten kesken. (Kimball et al. 2008; Velicanu & Matei 2007.)



Kuva 2.10. Tietovarasto- ja raportointijärjestelmän elinkaaren tyypilliset vaiheet (mukailtu lähteestä Kimball et al. 2008, s. 3; Velicanu & Matei 2007)

Vaatimusmäärittely edellyttää ymmärrystä liiketoiminnasta ja kykyä tunnistaa liiketoimintaprosessien sekä loppukäyttäjien vaatimuksia (Velicanu & Matei 2007). Tämä on tyypillisesti erittäin haasteellinen osa tietovarasto- ja raportointijärjestelmien kehityksessä, sillä käyttäjäryhmä on usein hyvin heterogeeninen, eikä järjestelmän potentiaalia välttämättä ymmärretä ennen kuin se on virallisesti tuotantokäytössä.

Kimball et al. (2008, s. 63-64) mukaan liiketoimintavaatimuksia tulisi kerätä kahdella tasolla: makro- ja mikrotasolla. Makrotasolla pyritään ymmärtämään liiketoiminnan vaatimuksia ja prioriteetteja järjestelmäprojektin tasolla. Vastaavasti mikrotasolla keskitytään käyttäjien asettamiin vaatimuksiin ja tarkastelu rajataan tyypillisesti tiettyyn kehitysiteraatioon. Liiketoiminnan analyysin ja vaatimusmäärittelyn avulla tulisi siis aikaansaada organisaationlaajuinen näkymä käyttäjien ja liiketoiminnan vaatimuksiin sekä osoittaa olennainen lähdedata näiden vaatimusten täyttämiseksi. (Kimball et al. 2008, s. 63-64; Velicanu & Matei 2007.) Tämä antaa mahdollisuuden arvioida ensimmäisen iteraation laajuutta (Kimball et al. 2008, s. 105; Velicanu & Matei 2007). Projektisuunnitelman toteutumista ja aktiviteettien koordinointia tuetaan projektinhallinnan keinoin projektisuunnitelman, toteutettavuuden arvioinnin ja vaatimusmäärittelyn jälkeen. Tämä tarkoittaa projektin kokonaisvaltaista hallintaa, jossa huomioidaan eri aktiviteettien integroituminen yhdeksi johdonmukaiseksi kokonaisuudeksi (Kimball et al. 2008, s. 3-4).

Liiketoiminnan analyysin ja vaatimusmäärittelyn jälkeen Velicanu & Matei (2007) esittää seuraavia vaiheita: tietovarastoarkkitehtuurin suunnittelua, teknologisten ratkaisujen valitsemista, iteraatiokierrosten suunnittelua ja yksityiskohtaisempaa suunnittelua sekä kehitystä. Kun tätä verrataan Kimball et al. (2008) esittämään elinkaarimalliin, huoma-

taan molempien noudattavan käytännössä samanlaista kaavaa. Kimball et al. (2008) elinkaarimallissa vaatimusmäärittelyn jälkeen siirrytään teknologian suunnitteluun ja kehitykseen, mikä tarkoittaa arkkitehtuurisuunnnittelua, teknologioiden valitsemista, tietomallinnusta, fyysistä suunnittelua, ETL-kehitystä sekä raportointityökalujen suunnittelua ja kehitystä. Velicanu & Matein (2007) malli tosin pyrkii vaiheistamaan kehitysprosessia Kimball et al. (2008) elinkaarimallia enemmän. Elinkaarimallissa nämä kehitysvaiheet ovat periaatteessa rinnakkaisia ja sisältävät tehtäviä tai aktiviteetteja, joiden välillä vallitsee jatkuvia riippuvuussuhteita (Kimball et al. 2008, s. 5).

Velicanu & Matein (2007) mukaan tietovarastoarkkitehtuurin suunnittelu aloitetaan, kun ensimmäisen iteraation laajuus on tiedossa ja liiketoiminnan analyysin tulokset ovat selvillä. Tässä vaiheessa tunnistetaan olemassa olevat ja puuttuvat tietovarastokomponentit, jotka pitää integroida, rakentaa tai hankkia tietovarasto- ja raportointijärjestelmää varten. Ensimmäiseksi määritetään looginen arkkitehtuuri, joka ei ota kantaa komponenttien sijoitteluun, vaan havainnollistaa, miten järjestelmä toteutetaan (Mäkinen 2000; Velicanu & Matei 2007). Tämän jälkeen määritetään tietoarkkitehtuuri, sovellusarkkitehtuuri, tekninen arkkitehtuuri ja tukiarkkitehtuurit. Tietoarkkitehtuurin tarkoitus on organisoida tietolähteet ja määrittää asianmukaiset laatu- ja hallintastandardit datalle ja metadatalle. Sovellusarkkitehtuuri havainnollistaa ohjelmistokomponentteja, jotka tarjoavat järjestelmän eri toiminnallisuudet kuten myös ETL-prosessin. Tekninen arkkitehtuuri tarjoaa vastaavasti tarkoituksenmukaisen infrastruktuurin tieto- ja sovellusarkkitehtuurien käytettäväksi. Se on fyysinen malli, joka sisältää palvelimet, työasemat, tietoverkon komponentit, laitteiston ja ohjelmistokomponentit verkkoyhteyksien muodostamista ja yhteydenpitoa varten. Tukiarkkitehtuuri kuvaa työkaluja tai komponentteja järjestelmän käytön ja kehittämisen hallitsemiseksi sekä suorituskyvyn valvomiseksi. Elinkaarimallissa nämä aktiviteetit sisältyvät arkkitehtuurisuunnitteluun.

Perusperiaate on, että arkkitehtuurit vastaavat laajennettavuuden, suorituskyvyn, saavutettavuuden, vakauden ja tietoturvan asettamiin vaatimuksiin kussakin tapauksessa (Velicanu & Matei 2007). Kun arkkitehtuurit on saatu luonnosteltua, siirrytään valitsemaan teknologiset ratkaisut. Vaiheen tarkoituksena on tunnistaa potentiaaliset teknologiat tieto- ja sovellusarkkitehtuurien toteuttamiseksi sekä toisaalta tarjota sopivat ratkaisut teknisen arkkitehtuurin ja tukiarkkitehtuurin totetuttamiseksi. Tässä kohtaa on huomioitava muun muassa tietovaraston laajuus ja koko, tietovarastoalustan skaalautuvuus ja tapa, jolla teknologian halutaan tukevan valittuja ohjelmistokomponentteja, operatiivisia järjestelmiä, tietokantojen hallintajärjestelmiä, kehitystyökaluja ja tiedon analysointityökaluja. (Kimball et al. 2008; Velicanu & Matei 2007.)

Teknologioiden valinnan jälkeen aiemmin tunnistetut liiketoiminnan sekä loppukäyttäjien tarpeet ja järjestelmän tekniset vaatimukset jalostetaan riittävän yksityiskohtaisiksi varsinaista kehitystyötä varten. Lisäksi arvioidaan järjestelmän laajuus uudelleen, tarkennetaan edellä suoritettujen vaiheiden tuloksia sekä dokumentoidaan tietolähteinä

käytettävien tietojärjestelmien eli lähdejärjestelmien asettamat rajoitukset, jotka vaikuttavat kehitystapaan. Kaikki tämä on osa suunnittelua. (Kimball et al. 2008; Velicanu & Matei 2007.) Kun iteraatioiden laajuus on selvillä, siirrytään yksityiskohtaisempaan suunnitteluun ja kehitykseen. Tässä kohtaa määritetään tietovaraston fyysinen malli ja rakenne, jonka on vastattava tunnistettuja informaatiotarpeita. Lähdetietueet transformoidaan kohdetietueiksi ja linkitykset lähdetaulujen kenttien ja kohdetaulun kenttien välillä muodostetaan, kun kohdetaulussa oleva data populoidaan useasta eri tietolähteestä. Lisäksi muodostetaan säännöt, joiden mukaan tietolähteistä saatava data poimitaan, muunnetaan ja ladataan tietovarastoon. Myös muita proseduureja, kuten tiedon puhdistusta, tietoturvaa, pääsyoikeuksia, varmuuskopiointia, arkistointia, testausta tai tuotantoon vientiä, yksityiskohtaistetaan ja tehdään tarkempia suunnitelmia toteutusta varten. Yksityiskohtaisten mallien, sääntöjen ja prosessien avulla aloitetaan varsinainen ETL- ja raportointityökalujen kehitystyö ja toteutetaan tietty osakokonaisuus prioriteettien mukaisesti. (Kimball et al. 2008; Velicanu & Matei 2007.)

Kehitystyön jälkeen seuraa tietovaraston testaus ja implementointi, jota voidaan pitää keskeisimpänä kehitysvaiheena tämän tutkimuksen kannalta. Tässä vaiheessa suoritetaan testaus, asennetaan laitteisto- ja ohjelmistokomponentit ja tehdään viimeisimmät konfiguroinnit datan poimintaa, puhdistusta, muuntelua, latausta ja säännöllisiä päivityksiä varten. Lisäksi voidaan perustaa erillinen testausympäristö testausta varten. Kun tietovaraston looginen ja fyysinen arkkitehtuuri on valmis, tulisi myös tietovaraston tietokantojen rakenteen kunkin iteraatiokierroksen määrittelyitä kohden valmiina siten, että tarkoituksenmukaiset indeksit on luotu ja mahdollinen osiointi on toteutettu. Tässä vaiheessa ainakin osalla loppukäyttäjistä on pääsy dataan, käyttöliittymä on valmis ja tärkeimmät raportit saatavilla testausta varten. Testaus aloitetaan kehittäjien toimesta, jolloin testitapauksia ajetaan tietokantoja vasten ja tulokset validoidaan testaussuunnitelman mukaisesti. Tämän jälkeen testauksessa otetaan myös loppukäyttäjät mukaan, jolloin he käyttävät sitä kuin tuotannossa on tarkoitus. Näin voidaan löytää ja korjata järjestelmässä piileviä virheitä tai vikoja, tunnistaa ongelmakohtia suorituskyvyssä ja tuoda loppukäyttäjät tutuiksi uuden järjestelmän ja käyttöliittymän kanssa. Ennen varsinaista käyttöönottoa ja tuotantoon siirtoa on siis vielä mahdollista tehdä muutoksia, jotka katsotaan tarpeelliseksi testauksen perusteella. (Velicanu & Matei 2007.) Kimball et al. (2008) elinkaarimallissa testausta ei ole kuitenkaan esitetty omana vaiheenaan, vaan testausta pidetään enemmänkin projektinhallinnan osana, jolloin se liittyy käytännössä kaikkiin vaiheisiin omana osa-alueenaan.

Toteutuksen ja testauksen jälkeen on vuorossa varsinainen käyttöönotto eli *roll-out* ja siirtyminen ylläpitoon. Tällöin loppukäyttäjät alkavat viimein hyödyntää järjestelmää ylläpidon ja pienkehityksen ollessa mukana. Käyttöönotto sisältää käyttökoulutuksen ja hallinnointiprosessien perustamisen tarkoituksenmukaisella laajuudella. Lisäksi aloitetaan eri mittareiden ja tilastollisten indikaattoreiden seuraaminen ja valvonta muutostarpeiden varalta. (Kimball et al. 2008; Velicanu & Matei 2007.)

3 JOHDATUS TESTAUKSEEN

3.1 Testauksen määritelmä

Testaus on laaja käsite ja se voidaan yhdistää lähes mihin tahansa kokeilemiseen ja kokeiluista saatujen tulosten analysointiin. Kokeilua ei kuitenkaan mielletä testaamiseksi ohjelmisto- ja järjestelmäkehityksen yhteydessä, vaan ohjelmistotestauksessa on kyse järjestelmällisestä ja ennalta suunnitellusta virheiden tai puutteiden etsimisestä ja ohjelman luotettavuuden arvioinnista dynaamisten tai staattisten testausmenetelmien avulla (Haikala & Märijärvi 2006, s. 287). *International Software Testing Qualification Board* (2012) kuvailee testausta prosessiksi, joka kattaa kehityselinkaaren määrittelystä tulosten arviointiin ja jonka avulla pyritään varmistumaan, että ohjelmistotuote täyttää sille asetetut vaatimukset ja on kaikilta osin tarkoituksenmukainen. Lisäksi ISTQB (2012) pitää virheiden havaitsemista ja niiden hallintaa osana testausta. Näin ollen, testauksella pyritään saavuttamaan luottamus siitä, että ohjelmistotuote toimii määritysten mukaisesti, mikä tarkoittaa luotettavuuden arviointia eri menetelmin (Haikala & Märijärvi 2006, s. 284; Hetzel 1988, s. 4). Tällainen lähestymistapa olettaa, että testaamalla varmistetaan vain ohjelmistotuotteen toimivan oikein. Toisaalta, Myers et al. (2004, s. 6) on lähestynyt testauksen määritelmää oletuksesta, että ohjelmistotuote sisältää aina virheitä: testaus on prosessi, jonka tarkoituksena on havaita ohjelmistotuotteessa esiintyvät virheet tai puutteet.

Testattaessa voidaan seurata esimerkiksi havaittujen virheiden kokonaismäärää kutakin ohjelmistokomponenttia kohden tai korjaamattomien virheiden määrää koko järjestelmää kohden. Tällaisia tilastoja hyödynnetään ohjelmistotuotteen luotettavuuden arvioinnissa. Schaeferin (2008, s. 41-42) mukaan eräs testauksen tarkoitus onkin juuri ohjelmistotuotteen laadun mittaaminen. Toisaalta Haikalan & Märijärven (2006, s. 287) tulkinnan mukaisesti testauksen tarkoitus on ohjelmistotuotteen luotettavuuden arviointi ja parantaminen. Järjestelmällinen ja ennalta suunniteltu toiminta virheiden, poikkeamien tai puutteiden havaitsemiseksi ja korjaamiseksi on luonteeltaan eräänlaista riskien hallintaa, mikä voidaan nostaa yhdeksi tärkeäksi testauksen tavoitteeksi. Näin ollen, testauksen keskeiset tavoitteet voidaan määrittää seuraavasti:

- ohjelmiston luotettavuuden arviointi ja parantaminen,
- laadun mittaaminen,
- riskien hallinta.

3.1.1 Testaustasot

Testaustasolla tarkoitetaan testausta tietyn tyyppisen testaustavoitteen ja testauskohteen ympärillä. Testaustasolla ei viitata testausvaiheeseen, vaan kukin testaustaso sisältää useampia testausvaiheita. (Pyhäjärvi & Pöyhönen 2005.) Testaustasot ovat tapa jakaa testausta pienempiin osiin, missä jokaisella osalla on erityinen tavoite ja kohde. Myös toimintatavat ja testaustekniikat vaihtelevat aina testaustason mukaan. Tyypillisimmät testaustasot ovat yksikkötestaus, integraatiotestaus, järjestelmätestaus ja hyväksyntätestaus (Black 2009, s. 502; Haikala & Märijärvi 2006, s. 288).

Yksikkötestauksessa testataan yksittäistä järjestelmän komponenttia ja testaustuloksia verrataan kyseisen komponentin määrittelyyn. Yksikkötestauksen suorittaminen voi edellyttää simuloitujen järjestelmäympäristön osien, testiajureiden ja tynkämoduulien toteutusta testattavan moduulin ympärille, jos sen toimintaa ei voida tarkastella erillisenä kokonaisuutena. Testiajureiden avulla voidaan tehdä pyyntöjä moduulin toteuttamiin palveluihin ja tarkastella tuloksia. Tynkämoduuleilla vastaavasti voidaan korvata sellaiset järjestelmän osat, joiden olemassa olosta testattava moduuli on riippuvainen. (Haikala & Märijärvi 2006, s. 288-289.)

Integraatiotestaus on yksikkötestauksesta seuraava taso, jossa yhdistetään järjestelmän komponentteja tai komponenttiryhmiä ja tarkastellaan komponenttien muodostaman kokonaisuuden toimintaa erityisesti komponenttien rajapintojen ja yhteensopivuuden kannalta. Testauksen tuloksia verrataan arkkitehtuurisuunnittelun lähtökohtiin. Integraatiotestaus etenee usein *bottom-up* –periaatteen mukaisesti, jolloin alimman tason komponentteja integroidaan suuremmaksi kokonaisuudeksi. Vastaavasti jäsentävässä tai osittavassa integraatiotestauksessa etenemissuunta on päinvastainen, jolloin testauksessa lähdetään liikkeelle suuremmista kokonaisuuksista ja edetään kohti alemman tason komponentteja. (Haikala & Märijärvi 2006, s. 290.)

Järjestelmätestauksen kohde on käytännössä koko järjestelmä ja testauksen tuloksia verrataan järjestelmän vaatimusmäärittelyyn, joka sisältää järjestelmältä edellytettävät ominaisuudet ja toiminnallisuudet. Usein järjestelmätestausta laajennetaan ei-toiminnallisten ominaisuuksien testauksella, jolloin järjestelmän toimintaa voidaan arvioida laajemmin ja perusteellisemmin. Kun yksikkö- ja integraatiotestaus suoritetaan usein järjestelmän kehittäjien toimesta, järjestelmätestauksen suorittaa tyypillisesti kehitystyöstä riippumaton taho. (Haikala & Märijärvi 2006, s. 290)

Hyväksyntätestaus on läheisesti sidoksissa järjestelmätestaukseen. Erona on käytännössä se, että hyväksyntätestauksen suorittavat loppukäyttäjät, jolloin testauksen tuloksia verrataan teknisen määrittelyn ja järjestelmävaatimusten sijasta loppukäyttäjien asettamiin liiketoimintavaatimuksiin ja toiminnalliseen määrittelyyn. (Haikala & Märijärvi 2006, s. 290.) Hyväksyntätestauksen tarkoitus on varmistaa, ettei järjestelmässä ole

merkittäviä virheitä ja että se täyttää järjestelmälle asetetut hyväksymiskriteerit erityisesti tilaajan näkökulmasta (Bhat 2007; Brahmkshatriyan 2007). Toisin sanoen hyväksyntätestaus suoritetaan, kun halutaan tutkia, täyttääkö testauksen kohteena oleva valmis järjestelmä loppukäyttäjien asettamat tarpeet ja vaatimukset ja toimiiko järjestelmä kelvollisesti muun muassa käytettävyyden näkökulmasta. Hyväksyntätestauksen tarkoitus on kaksiosainen. Sen ensisijainen tarkoitus on varmistaa, että järjestelmä kykenee suoriutumaan tehtävistä, joita varten se on suunniteltu. Toissijainen tarkoitus on saada käyttäjiltä palautetta, varsinkin käytettävyyteen liittyen. (Rainardi 2008, s. 486.)

3.1.2 Virheiden luokittelu

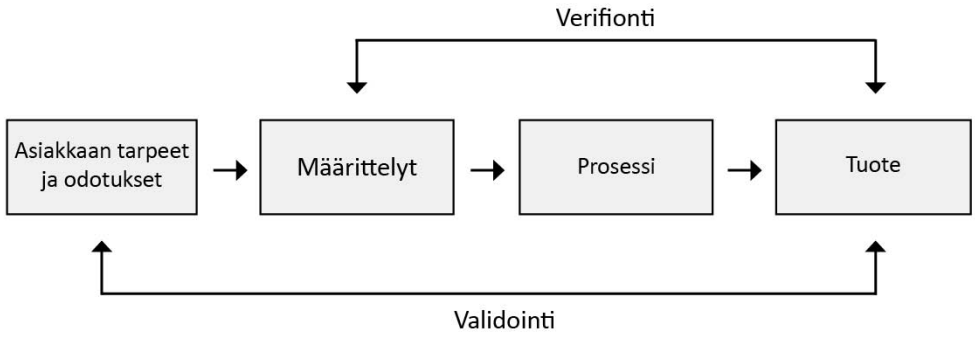
Virheellä viitataan tyypillisesti ihmisen toimintaan, joka tuottaa väärän tuloksen (ISTQB 2012). Ohjelmistotuotannossa toiminta liittyy tyypillisesti johonkin ohjelmistotuotteen elinkaaren vaiheeseen, kuten vaatimusmäärittelyyn, suunnitteluun, kehitystyöhön tai testaukseen. Virheellä voidaan näin ollen tarkoittaa ihmisen toiminnan kautta syntynyttä vikaa tai puutetta ohjelmistossa tai järjestelmässä. ISTQB:n (2012) määritelmän mukaan vika on järjestelmän komponentissa tai järjestelmässä oleva virhe, jonka johdosta järjestelmä ei kykene suorittamaan siltä edellytettävää toimintoa asianmukaisesti. Haikalan & Märijärven (2006) mukaan myös puute voi olla virhe. Puute on jokin järjestelmälle määritelty ominaisuus tai toiminnallisuus, jota ei löydy varsinaisesta ohjelmistosta (Haikala & Märijärvi 2006, s. 287). Virheeseen törmääminen ohjelmiston suorituksen aikana voi johtaa järjestelmän komponentin tai koko järjestelmän häiriöön, jolloin ohjelmiston toiminta poikkeaa jollain tavoin odotetusta toiminnasta. (ISTQB 2012) Koska virheet ovat peräisin ihmisen toiminnasta, on täysin luonnollista, että virheitä esiintyy jokaisen ohjelmistoprojektin yhteydessä. Virhe voi aiheutua esimerkiksi virheellisestä ajattelutavasta, huonosta kommunikaatiosta tai vääristä toimintatavoista. Näin ollen jo organisaatiokulttuuri, yksilöiden henkilökohtaiset ominaisuudet ja yksilöiden välinen kommunikointi yhdessä vaikuttavat olennaisesti vikojen ja puutteiden syntyyn. Ongelmallista virheiden luokittelussa on tulkinnanvaraisuus eri näkökulmien välillä. Esimerkiksi asiakkaan mieltämä virhe voi toimittajan näkökulmasta olla ohjelmiston ominaisuus (Haikala & Märijärvi 2006, s. 287).

Jokainen ohjelmistossa oleva virhe muodostaa riskin. Testauksen avulla havaitut virheet voidaan luokitella niiden vakavuuden ja prioriteetin mukaan. Virheiden vakavuus vaihtelee virheittäin, mikä edellyttää harkintaa jatkotoimenpiteitä pohdittaessa. Pahimmassa tapauksessa yksi virhe voi estää koko järjestelmän toiminnan. Toisaalta virhe voi olla järjestelmän toiminnan kannalta merkityksetön, jolloin sen korjaamatta jättäminen ei vaikuta järjestelmän toimintaan, eikä korjaustoimenpiteitä kannata välttämättä tehdä resurssien niukkuuden ja olemattomien hyötyjen vuoksi (Haikala & Märijärvi 2006, s. 287). Virheiden priorisointi voidaan tehdä arvioimalla vakavuutta ja vaikutuksia. Vaikka virheitä ei voida ennustaa etukäteen, voidaan potentiaalisten virheiden esiintymistodennäköisyyttä arvioida. Näin voidaan suunnitella ja suorittaa testaus siten, että vakavuudeltaan suurimmat virheet voitaisiin löytää mahdollisimman aikaisessa vaiheessa

ohjelmistotuotteen kehityselinkaarta ennen kustannusvaikutusten realisoitumista. Resurssien niukkuuden ja ohjelmistojen kompleksisuuden johdosta testausta ei voida suorittaa 100 % kattavuudella (van Veenendaal 1997). Tämä tarkoittaa sitä, että valmiit ja pitkään käytössä olleetkin ohjelmistot tai järjestelmät sisältävät jonkinasteisia virheitä. Osa virheistä on toiminnan kannalta täysin merkityksettömiä, mutta osa voi aiheuttaa hankaluuksia esimerkiksi poikkeustilanteissa. Muutama prosentti virheistä jää kokonaan havaitsematta suuren syöteavaruuden tai testitapausjoukon vuoksi. Tähän vaikuttaa lisäksi se, että virheen sisältävän kohdan suorittaminen järjestelmässä ei välttämättä johda virheelliseen toimintoon tai tulokseen, vaan johtaa laajempaan järjestelmävikaan (Haikala & Märijärvi 2006, s. 287-288). Järjestelmävika voi kumoutua jonkin toisen virheen tai toiminnon myötä. Pahimmassa tapauksessa se voi kuitenkin johtaa häiriöön, jonka vaikutus ilmenee järjestelmän ulkoisessa toiminnassa (Haikala & Märijärvi 2006, s. 287-288).

3.2 Verifiointi vs. validointi

Tämän tutkimuksen osalta testaus jaetaan luonteeltaan kahteen kategoriaan: verifiointiin ja validointiin.



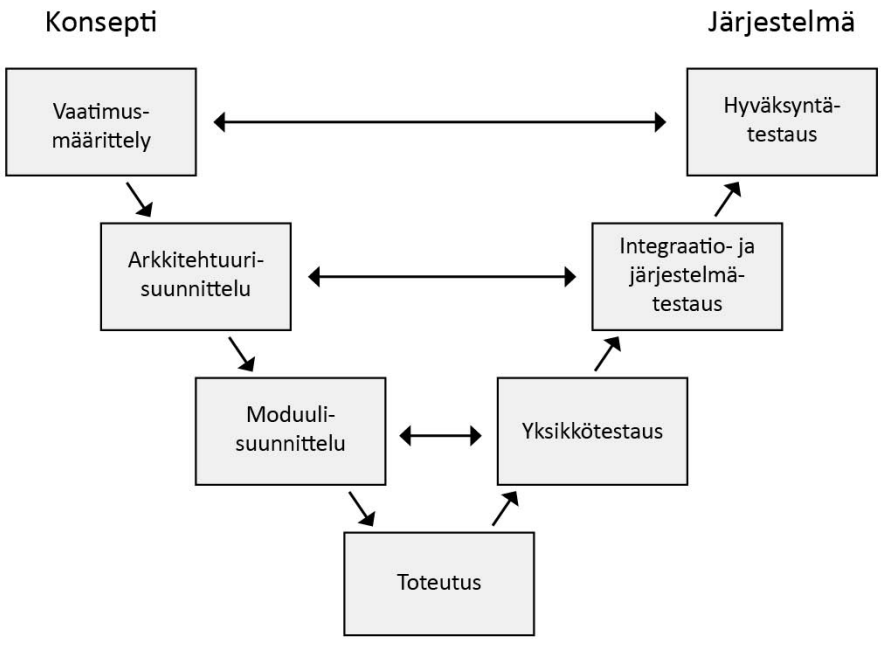
Kuva 3.1. Verifiointi ja validointi

International Software Testing Qualifications Board (2012) määrittelee verifioinnin määrättyjen vaatimusten täyttymisen vahvistamiseksi kokeellisesti ja objektiivisen todistusaineiston avulla. Validointi vastaavasti on määrättyä käyttöä varten tai määrätylle sovellukselle asetettujen vaatimusten täyttymisen vahvistamista kokeellisesti ja objektiivisen todistusaineiston avulla. Näin ollen, verifioinnissa on kyse siitä, rakennetaanko ohjelmistoa oikein, ja validoinnissa siitä, rakennetaanko oikeaa ohjelmistoa. Kuvassa 3.1 havainnollistetaan, miten verifioinnissa tuotetta verrataan määrittelyihin ja validoinnissa liiketoiminnan asettamiin vaatimuksiin ja odotuksiin. Usein verifioinnilla tarkoitetaan katselmuksia tai tarkistuksia ja validoinnilla testausta, mutta tämän työn osalta testauksen voidaan katsoa sisältävän sekä verifiointia ja validointia eli staattista dokumentaation tai ohjelmakoodin läpikäyntiä sekä dynaamista ohjelmiston analysointia.

Testaussuunitelmaa laadittaessa tulisi selvittää, miten järjestelmää testataan eli mitä halutaan verifioida tai validoida (Leffingwell & Widrig 2000, s. 361). Haasteena on kuitenkin testauksen riittävyyden arvioiminen, kun testauksen kustannukset halutaan minimoida laadusta tinkimättä. Leffingwell & Widrig (2000) esittävät kaksi lähestymistapaa määrittämään, mitä tulisi verifioida tai validoida. Pienemmissä projekteissa tulisi lähtökohtaisesti harkita verifioinnin ja validoinnin soveltamista jokaiseen projektin osa-alueeseen (Leffingwell & Widrig 2000, s. 362). Tällainen lähestymistapa on yksinkertainen ja käsittelee projektin eri osa-alueita tasavertaisesti mutta johtaa tyypillisesti selektiiviseen menettelyyn. Tämä tarkoittaa oletuksia siitä, kannattaako jotain triviaalilta tuntuvaa asiaa testata. Selektiivisyys voi osoittautua hyväksi menetelmäksi, jos ymmärretään valintaan liittyvät riskit ja syyt siihen, miksi näin tehdään. (Leffingwell & Widrig 200, s. 362.) Toisaalta, parempi tapa voi olla lähestyä asiaa riskianalyysin näkökulmasta. Näin ollen, testauksen laajuus sekä verifioinnin tai validoinnin tarve voidaan määrittää riskien vakavuuden ja esiintymistodennäköisyyden perusteella. Priorisoimalla riskejä voidaan priorisoida myös testausta erityisesti silloin, kun testaukselle osoitetut resurssit ovat niukat (Leffingwell & Widrig 200, s. 364).

3.3 Testaus osana järjestelmän kehityselinkaarta

Testauksen V-malli (kuva 3.2) noudattaa ohjelmistokehityksestä tuttua vesiputousmallia ja testaus tapahtuu ohjelmistokehityksen ehdoilla (Haikala & Märijärvi 2006, s. 288). V-mallissa ohjelmiston kehitys tapahtuu peräkkäisissä vaiheissa vaatimusmäärittelystä toteutukseen. Vastaava testaustaso ja siihen liittyvät aktiviteetit määräytyvät aina kehitysvaiheen mukaan.



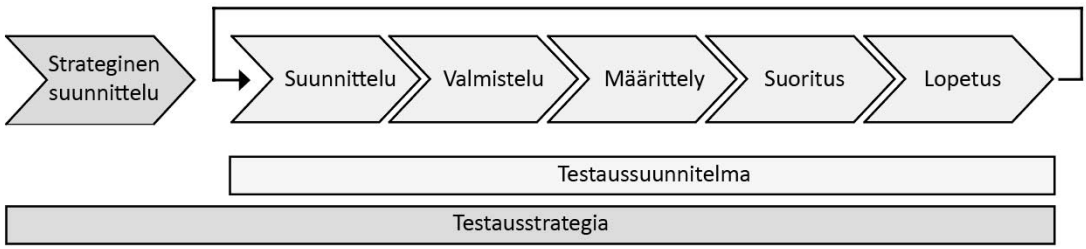
Kuva 3.2. Testauksen V-malli (Black 2009, s. 502; Haikala & Märijärvi 2006, s. 288)
V-mallissa validointia suoritetaan mallin oikealla puolella, kun taas verifointi suoritetaan osana kehitystä V-mallin konseptin puoleisissa vaiheissa. V-malli vaiheistaa testauksen liittämällä testauksen osaksi vesiputousmallin mukaista ohjelmiston kehitysprosessia. Näin esimerkiksi vaatimusmäärittely toimii hyväksyntätestauksen perustana, jolloin hyväksyntätestauksen valmistelu voidaan aloittaa heti tämän vaiheen alkaessa (Black 2009, s. 502).

V-mallin suurin ongelma on heikko ennustettavuus. Tämä tarkoittaa sitä, että vaiheita on vaikea suunnitella tarkasti etukäteen. Jos jokin kehitysvaiheista viivästyy tai vie odotettua enemmän aikaa, V-mallin koko vasen kehitysvaiheiden sarja siirtyy hiljalleen oikealle käyttöönoton viivästyessä. Toisaalta V-malli on usein tiukasti sidoksissa testauspuolen osalta, eikä mallin oikeanpuoleinen testausvaiheiden sarja siirry kehityspuolen mukana. Tämä johtaa siihen, että aikataulu- ja kustannuspaineiden lisääntyessä testaukseen kiinnitetään vähemmän huomiota ja resurssipulan vuoksi osasta testauksesta voidaan jopa joutua luopumaan. (Black 2009, s. 502-503.) Blackin (2009, s. 504) mukaan tämän vuoksi eri kehitysvaiheiden aloitus- ja lopetuskriteereihin on kiinnitettävä erityistä huomiota V-mallin kohdalla.

Leffingwellin & Widrigin (2000) mukaan testaus on jatkuvaa iteratiivisessa kehityksessä, eikä sitä voida irroittaa erilliseksi kokonaisuudeksi (kuva 2.10). Siinä toistuvat kuitenkin V-mallista tutut testaustasot eri testityyppeineen kustakin iteraatiosta ja projektista riippuen.

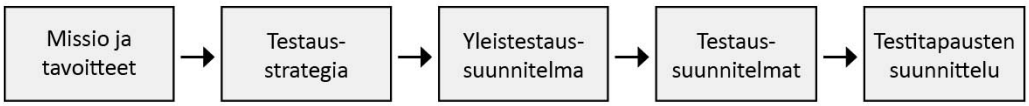
3.4 Testausprosessin kuvaaminen

Testausprosessi muodostuu viidestä testausvaiheesta, joita ovat suunnittelu, valmistelu, määrittely, suoritus ja lopetus (Black 2009; van Veenendaal & Pol 1997). Testausvaiheet organisoidaan tavallisesti kuvassa 3.3 esitetyn elinkaarimallin mukaisesti. Testausvaihe on testaustason sisäinen tai useiden testaustasojen yhteinen tehtäväkokonaisuus, joka on tyypiltään kertaluonteinen eikä jatkuva. (Pyhäjärvi & Pöyhönen 2005.)



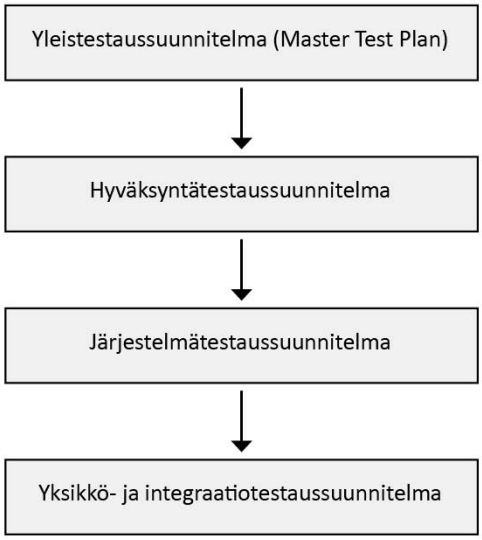
Kuva 3.3. Testauksen elinkaari (mukailtu lähteistä Black 2009, s. 535; van Veenendaal & Pol 1997)

Van Veenendaalin & Polin (1997) mukaan suunnitteluun tulisi käyttää 20 %, valmisteluun 40 % ja suoritukseen ja lopetukseen yhteensä loput 40 % kokonaispanoksesta, mikä korostaa suunnittelu- ja valmisteluvaiheen merkitystä testausprosessissa. Suunnitteluvaihetta voidaan pitää tärkeimpänä vaiheena, jossa fokus on testaukseen liittyvien realiteettien tunnistamisessa, testauksen suuntaviivojen asettamisessa ja testaussuunnitelmien laadinnassa (Black 2009, s. 534). Näin ollen, se muodostaa perustan hallittavalle ja laadukkaalle testausprosessille. Suunnitteluvaiheessa tutustutaan vaatimusmäärittelyyn, ohjelmiston toiminnallisuuksiin ja projektiorganisaatioon. Koska ohjelmiston täydellinen testaaminen on käytännössä mahdotonta, on suunniteltava ja priorisoitava huolellisesti mitä testataan ja miten testataan. (van Veenendaal & Pol 1997.) Kuvassa 3.4 havainnollistetaan suunnitteluvaiheen keskeisimmät tehtävät.



Kuva 3.4. Testauksen suunnittelu (mukailtu lähteestä Pyhäjärvi & Pöyhönen 2005)

Suunnitteluvaiheessa ensimmäinen tehtävä on mission ja tavoitteiden määrittäminen. Missio määrää testauksen tarkoituksen, ja sen avulla testauksen onnistumista voitaisiin arvioida eri sidosryhmien näkökulmista (Pyhäjärvi & Pöyhönen 2005). Odotukset testausta kohtaan vaihtelevat sidosryhmittäin, joten kaikkien sidosryhmien osallistaminen on järkevää. Mission ja tavoitteiden määrittämisen jälkeen siirrytään testausstrategian laadintaan. Se on prosessi, johon kaikkien sidosryhmien tulisi osallistua, jotta osataan arvioida mitä osa-alueita testauksessa pitäisi korostaa laatuominaisuudet huomioiden (van Veenendaal & Pol 1997).



Kuva 3.5. Testaussuunnitelmien hierarkia (van Veenendaal & Pol 1997)

Testausstrategian pohjalta voidaan muodostaa yleistestaussuunnitelma (Master Test Plan), joka havainnollistaa mitä testataan, miten testataan, kuka testaa ja milloin testataan. Ideaalitapauksessa yleistestaussuunnitelma sisältää kaiken tyyppiset testit, mutta usein yleistestaussuunnitelmaa joudutaan rajaamaan koskemaan vain joko musta- tai valkolaatikkotestausta (van Veenendaal & Pol 1997). Yleistestaussuunnitelman myötä luodaan yksityiskohtaisemmat testaussuunnitelmat kutakin testaustasoa ja –tyyppiä kohden, minkä jälkeen laaditaan testitapaukset. Hierarkia eri suunnitelmien suhteen havainnollistetaan kuvassa 3.5. Näin ollen, suunnitteluvaiheessa otetaan huomioon:

- tavoitteet
- riskit,
- lähestymistapa,
- resurssointi,
- testauksen seuranta,
- korjaustoimenpiteet,
- ongelmien ratkaisu,
- testauksen laadun arviointi, ja
- kommunikointi eri sidosryhmien kesken (Pyhäjärvi & Pöyhönen 2005).

Valmisteluvaihe voidaan aloittaa valmiiden testaussuunnitelmien pohjalta. Vaatimuksia ja muita määrittelyitä katselmoidaan testattavuuden näkökulmasta (van Veenendaal & Pol 1997; Pyhäjärvi & Pöyhönen 2005). Katselmointien jälkeen voidaan aloittaa testiympäristöjen ja –aineistojen valmisteleminen sekä tarvittavan tiedon kerääminen (Pyhäjärvi & Pöyhönen 2005). Valmistelun aikana ja rakennetaan testausjärjestelmä eli

kootaan testaajat, pystytetään testiympäristöt, asennetaan tarvittavat työkalut ja määritetään testausprosessit (Black 2009, s. 80).

Valmisteluvaiheen jälkeen siirrytään määrittelyvaiheeseen, jossa määritellään ja priorisoidaan aiemmin suunnitellut testitapaukset, määritellään virallinen testiaineisto, luodaan ja konfiguroidaan testiympäristöt ja automatisoidaan testejä tarpeen mukaan (van Veenendaal & Pol 1997; Pyhäjärvi & Pöyhönen 2005). Testitapausten määrittely sisältää kuvauksen muun muassa syötteistä, prosessista ja odotettavista tuloksista. Lopulta testitapaukset käännetään fyysiseen muotoon eli esimerkiksi testiskripteiksi. (van Veenendaal & Pol 1997). Pyhäjärven & Pöyhösen (2005) mukaan määrittelyvaiheessa on otettava huomioon myös mahdolliset riskit, joita testien suoritukseen voi liittyä.

Suoritusvaihe voidaan aloittaa, kun testauksen kohteena olevat komponentit ovat saatavilla. Testaus aloitetaan esitesteillä, joiden avulla voidaan arvioida testitapausten soveltuvuutta ja ohjelmiston toiminnallisuuksien testattavuutta käytännössä. Esitestien jälkeen voidaan suorittaa testitapaukset tai –skriptit. Tähän liittyy myös testitapausten uusintatestaus ja uudelleentestaus eli regressiotestaus. (van Veenendaal & Pol 1997; Pyhäjärvi & Pöyhönen 2005) Testien suorituksen yhteydessä tulisi kirjata ja raportoida testien tuloksia ja testauksen etenemistä kuvaavia asioita, joiden pohjalta voidaan laatia ennusteita tai arvioita testausprosessiin liittyen (van Veenendaal & Pol 1997). Erot odotettujen tulosten ja todellisten tulosten välillä voivat ennakoida virhettä järjestelmässä tai komponentissa, mutta toisaalta myös virhettä tai puutetta vaatimuksissa, tai testausympäristössä. (van Veenendaal & Pol 1997.)

Lopetusvaihe on viimeinen osa testausprosessia. Kun testit on suoritettu, on arvioitava testausprosessia sekä ohjelmiston laatua kokonaisuudessaan ja päättää prosessi järjestelmällisellä tavalla. Tämä tarkoittaa tilastotietojen ja väliraporttien tietojen yhdistämistä tuloksiksi loppuraportille, jonka perusteella voidaan tehdä tarvittavat päätökset jatkotoimenpiteitä ajatellen. (van Veenendaal & Pol 1997.) Lopetusvaiheessa tehdään yhteenveto testattavuudesta, tuloksista ja kokemuksista sekä päivitetään testaukseen liittyvät materiaalit (Pyhäjärvi & Pöyhönen 2005). Yhteenvetojen avulla pitäisi pyrkiä oppimaan testausprosessista, jotta tulevaisuudessa sen läpivienti olisi tehokkaampaa.

4 TIETOVARASTO- JA RAPORTOINTIJÄRJESTELMIEN TESTAUS

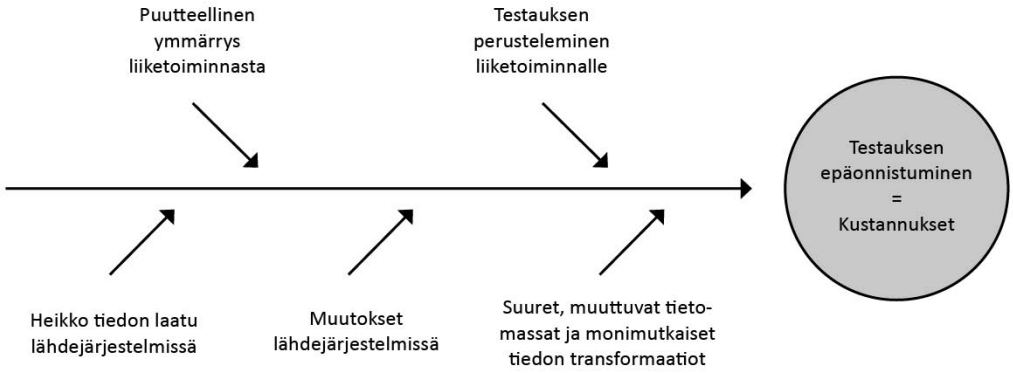
4.1 Keskeiset haasteet tietovarasto- ja raportointijärjestelmien testauksessa

Organisaatioiden tuottama ja hankkima tieto on jatkuvassa muutoksessa. Tieto muuttuu yhä monimutkaisemmaksi ja monimuotoisemmaksi myös sen määrän kasvaessa voimakkaasti vuodesta toiseen. Tämä asettaa haasteen myös tietovarasto- ja raportointijärjestelmien testaukselle, jossa fokus on järjestelmän tietosisällön validoinnissa (Mathen 2010).

Tiedon suuret volyymit ja lähdejärjestelmien heterogeenisuus aiheuttavat helposti ongelmia testauksen kannalta erityisesti tilanteissa, joissa lähdejärjestelmien tiedon laatu on heikkoa. Tämä aiheuttaa ongelmia ainakin testauksen suunnittelun ja tulosten kannalta, mikäli lähdejärjestelmien tietosisällön kartoittaminen ja profilointi jää suorittamatta (Mathen 2010). Englishin (1999, s. 4) mukaan epäonnistuneet tietovarasto- ja raportointijärjestelmät ovat kärsineet juuri taustalla olevan tiedon heikosta laadusta, mikä tarkoittaa puutteita, epäjohdonmukaisuuksia ja merkityseroja tiedossa, tiedon väärinkäyttöä ja toisaalta järjestelmän kyvyttömyyttä yhdenmukaistaa eri lähdejärjestelmistä poimittua tietoa asianmukaisesti myös häiriötilanteissa. Tiedon laatua voidaan tarkastella kahdesta näkökulmasta: luontaisesta ja pragmaattisesta. Luontaisesta näkökulmasta tarkasteltuna tiedon laadulla tarkoitetaan muun muassa tiedon oikeellisuutta, paikkaansapitävyyttä ja johdonmukaisuutta. Vastaavasti pragmaattisesta näkökulmasta tiedon laadulla tarkoitetaan sitä, kuinka hyvin tieto on hyödynnettävissä liiketoiminnan näkökulmasta ja kuinka paljon se tuottaa lisäarvoa liiketoiminnalle (English 1999, s. 22). Testauksessa on otettava huomioon nämä molemmat näkökulmat, jotta lopputulos on tilaajan kannalta paras mahdollinen. Pragmaattinen laatu perustuu siis liiketoimintavaatimusten toteutettavuuteen käytettävissä olevalla tietosisällöllä. Tämän seurauksena liiketoiminnan tarpeiden ymmärtäminen on oleellista, jotta voidaan arvioida käytettävissä olevan lähdedatan riittävyyttä näiden tarpeiden täyttämiseksi. Keskeisenä ongelmana onkin liiketoiminnan puutteellinen ymmärrys, joka johtaa helposti väärien asioiden testaukseen ja lopulta testausprosessin epäonnistumiseen (Mathen 2010). Haaste piilee siinä, miten toimittaja onnistuu hankkimaan riittävän ymmärryksen asiakkaan liiketoiminnasta ja löytämään olennaiset asiat, jotka ohjaavat tarpeita.

Testaus ei ole vain teknologian tuntemusta ja mekaanista työtä. Se on luovaa työtä, joka edellyttää osaamisen lisäksi ennen kaikkea asianmukaista asennetta ja testausta suosivaa organisaatiokulttuuria. Testauksen onnistumiseen vaikuttavat myös erilaiset oletukset esimerkiksi virheiden esiintyvyyden osalta. Vucevic & Zhang (2011, s. 127) määrittävät onnistuneen testauksen havaittujen virheiden, poikkeamien tai puutteiden aikaansaannokseksi. Toisaalta oletuksia siitä, että järjestelmästä löytyy virheitä tai puutteita, vähätellään turhan helposti. Eräs merkittävä haaste onkin siinä, miten testauksen merkitys saadaan perusteltua liiketoiminnalle, joka on käytännössä järjestelmän ensisijainen asiakas.

Tietovarasto- ja raportointijärjestelmien testaus sisältää muitakin haasteita. Esimerkiksi suuret tietomassat aiheuttavat sen, että tietosisältöjen täydellinen validointi käy usein lähes mahdottomaksi (Kamal & Nakul 2013; Mathen 2010). Lisäksi heterogeeninen lähdedata päivittyy usein eri aikaisesti, mikä saattaa aiheuttaa toistuvia epäjohdonmukaisuuksia ja vaikuttaa testauksen tuloksiin negatiivisesti. Tosiasiassa asiantuntemus ja kokemus yhdessä eivät estä virheiden syntyä, vaan virheitä syntyy aina manuaalista työtä tehdessä. Virheiden aiheuttamat kustannukset kasvavat sitä mukaa, mitä myöhemmässä elinkaaren vaiheessa ne havaitaan (Mathen 2010). Järjestelmäprojektien erilaisuuden ja yksilöllisyyden vuoksi testauksen riittävyyttä ja asianmukaisia testauksen suoritustapoja on erittäin hankala arvioida. Toisaalta testauksen hyvien käytäntöjen soveltaminen helpottaa testauksen läpivientiä ja auttaa hankalasti arvioitavien asioiden arvioinnissa (Pyhäjärvi & Pöyhönen 2005). Kuvassa 4.1 tiivistetään keskeisimmät haasteet tietovarasto- ja raportointijärjestelmien testauksessa.



Kuva 4.1. Keskeiset haasteet tietovarasto- ja raportointijärjestelmien testauksessa

4.2 Tietovarasto- ja raportointijärjestelmien testauksen erityispiirteet

Perinteinen ohjelmistotestaus on tärkeä osa tietovarasto- ja raportointijärjestelmän testausta, mutta tietovarasto- ja raportointijärjestelmän testaukseen sisältyy paljon erityispiirteitä, jotka erottavat sen merkittävissä määrin perinteisestä ohjelmistotestauksesta (Gupta et al. 2012; Vucevic & Zhang 2011, s. 42). Tietovarasto- ja raportointijärjestelmän testauksen fokus on sen tuottamassa strategisessa informaatiossa, jota liiketoiminta hyödyntää. Tämän vuoksi datan puhdistaminen, verifiointi, validointi ja muuntaminen on merkittävässä roolissa tietovarasto- ja raportointijärjestelmän testauksessa. (Gupta et al. 2012.)

Toisaalta, asiaa voidaan lähestyä myös liiketoiminnan näkökulmasta: tietovarasto- ja raportointijärjestelmän kehitys saa alkunsa kohdasta, jossa liiketoiminta ja informaatioteknologia kohtaavat. Tällaisen kahden toimialueen välillä tasapainoilevan järjestelmän kehitystä ei voida luonnehtia projektiksi, vaan hankkeeksi, jossa kehitystoimenpiteet ja ylläpito ovat jatkuvasti läsnä. (Vucevic & Zhang 2011, s. 42) Tietovarasto- ja raportointijärjestelmän rakentaminen ja kehittäminen ei ole kertaluonteinen ponnistus vaan edellyttää jatkuvaa ja pitkäjänteistä työtä määritysten, vaatimusten, ympäristön ja liiketoiminnan muutosten myötä. Eräs keskeinen yhdistävä tekijä ohjelmistotestauksen ja tietovarasto- ja raportointijärjestelmien testauksen välillä onkin se, että molemmissa vaatimusten määrittelyvaiheella on olennainen merkitys (Gupta et al. 2012). Jos järjestelmän vaatimuksia on vaikea ymmärtää tai ne ovat epäselviä ja epäjohdonmukaisia, testaus epäonnistuu suurella todennäköisyydellä. Tämä on yleistä tietovarasto- ja raportointijärjestelmien tapauksessa, sillä liiketoiminnan asettamat vaatimukset ovat usein vaikeasti määriteltävissä tai rajattavissa hankkeen alkuvaiheessa. Tämä aiheuttaa epävakautta ja altistaa vaatimukset muutoksille. Näin ollen, kehitys on usein luonteeltaan vahvasti iteratiivista ja elinkaarimalli eroaa perinteisten ohjelmistotuotteiden elinkaarimalleista (Vucevic & Zhang 2011, s. 42-43).

Eroja ohjelmistotestauksen ja tietovarasto- ja raportointijärjestelmien testauksen välillä on löydettävissä melko paljon, mikä tekee tietovarasto- ja raportointijärjestelmien testauksesta oman taitolajinsa. Testaus on olennainen osa tietovarasto- ja raportointijärjestelmien kehitysprosessia ja sen tulisi olla läsnä koko elinkaaren ajan. Sen tavoitteet, ominaispiirteet ja laajuus eroavat kuitenkin perinteisestä ohjelmistotestauksesta. Tietovarasto- ja raportointijärjestelmän testaus keskittyy informaation tarjoamiin liiketoimintahyötyihin ohjelmistotestauksen fokuksen ollessa tätä suppeampi (Vucevic & Zhang 2011, s. 42). Lisäksi tietovarasto- ja raportointijärjestelmän vaatimusten muutosherkkyys vaikuttaa testauksen ominaispiirteisiin aina vaatimusten määrittelystä alkaen. On esimerkiksi tärkeää, että liiketoiminnan edustajat osallistetaan kehitysprosessiin ja sitä kautta testaukseen heti alusta alkaen, jotta he ymmärtäisivät teknologian potentiaalin ja osaisivat liittää sen osaksi liiketoimintaa (Vucevic & Zhang 2011, s. 43).

Taulukkoon 4.1 on listattu keskeisimpiä eroja perinteisen ohjelmistotestauksen ja tietovarasto- ja raportointijärjestelmien testauksen välillä.

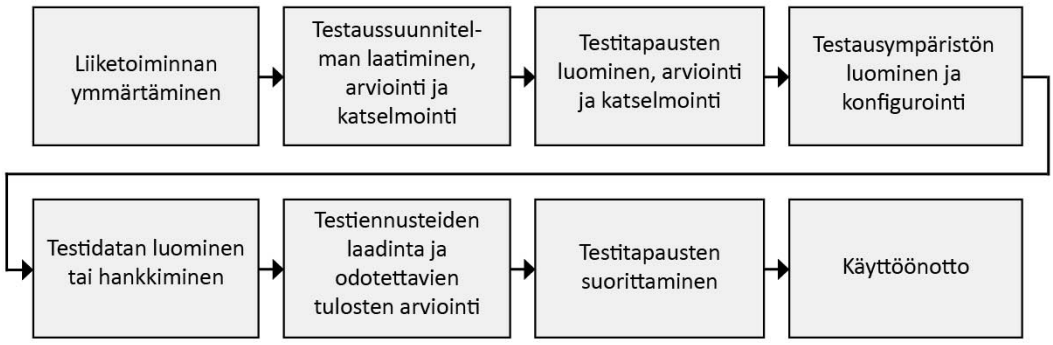
Taulukko 4.1. Merkittävimmät erot ohjelmistotestauksen ja tietovarasto- ja raportointijärjestelmien testauksen välillä (Gupta et al. 2012; Vucevic & Zhang 2011)

Ohjelmistotestaus	Tietovarasto- ja raportointijärjestelmien testaus
Toiminnallisuuksien testaaminen ja ohjelmakoodin verifiointi on keskiössä	Tietosisällön validoiminen on keskiössä
Liiketoiminnan edustajien osallistaminen ei ole merkittävässä roolissa	Liiketoiminnan edustajien osallistaminen on merkittävässä roolissa
Määrittelyt ovat usein selkeitä ja yksiselitteisiä, muutoksia tehdään suhteellisen vähän	Määrittelyt ovat usein epämääräisiä ja saattavat muuttua jatkuvasti (myös käyttöönoton jälkeen)
Suoritetaan pääosin ennen käyttöönottoa	On jatkuvaa ja painopiste on käyttöönoton jälkeisissä elinkaaren vaiheissa
Keskiössä on eri käyttötapausten testaaminen testitapausten mukaisesti	Keskiössä ovat erilaiset hakuoperaatiot tai kyselyt, jotka kohdistuvat käytettävissä olevaan testidataan
Tietokantoja voidaan testata ilman liiketoimintasääntöjen tai -logiikan läpikäyntiä	Tietokantoja ei voida testata ilman liiketoimintasääntöjen tai -logiikan läpikäyntiä, sillä jokaisen testin tulisi perustua jonkin liiketoimintasäännön tai -logiikan tarkistukseen
Tehdään usein suoraan käyttöliittymästä	Käyttöliittymää ei tyypillisesti käytetä tai sitä ei ole olemassa
Testi on käyttäjän käynnistämä tapahtua, jossa käyttäjä antaa järjestelmälle syötteen	Testi on järjestelmän käynnistämä tapahtuma
Nopeat tulokset mahdollisia	Prosessointiin ja tulosten saamiseen voi kulua paljon aikaa
Fokus on virheiden aiheuttamien kustannusten minimoimisessa	Testauksen fokus on oikeellisessa tiedossa, jonka pohjalta voidaan tehdä liiketoimintakriittisiä päätöksiä

4.3 Testausprosessin teoreettinen malli

Mookerjea & Malisetty (2008) esittävät prosessimallin, jossa tietovarasto- ja raportointijärjestelmän testausprosessi muodostuu kahdeksasta kriittisestä vaiheesta. Tämä prosessimalli esitetään kuvassa 4.2. Kuvan mukaisesti testausprosessin tulisi lähteä liikkeelle vaiheesta, jossa testausta lähestytään liiketoimintalähtöisesti melko karkealla tasolla. Tämä tarkoittaa liiketoimintavaatimusten analysointia ja liiketoimintasääntöjen validointia, mikä antaa mahdollisuuden muodostaa tarkoituksenmukaisen lähestymistavan testaukseen ja arvioida testien edellyttämiä resursseja. (Mookerjea & Malisetty 2008.)

Tämä antaa lähtökohdat kattavien määrittelyiden luomiseksi. Liiketoimintavaatimukset ovat tyypillisesti ominaisuuksia, joita käyttäjät odottavat järjestelmältä, jotta he pystyisivät suoriutumaan määrätyistä tehtävistä määritetyllä tavalla (Rainardi 2008, s. 480). Näin ollen liiketoimintavaatimusten tulisi olla olennainen osa testausta. Kuva 4.3 havainnollistaa liiketoimintavaatimusten ja testauksen suhteen neljällä eri tasolla.



Kuva 4.2. Tietovarasto- ja raportointijärjestelmien testausprosessin tyypillinen vaihejako (Mookerjea & Malisetty 2008)

Kuva 4.3. Liiketoimintavaatimusten yhteys testaukseen (mukailtu lähteestä Itkonen & Kauppinen 2011)

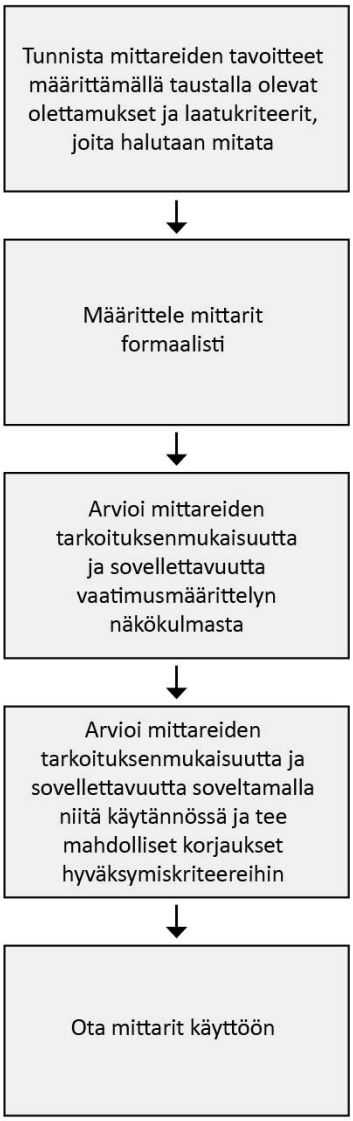
Organisaatioissa tulisi pyrkiä kohti tilannetta, jossa liiketoimintavaatimukset ja testaus olisivat tiiviisti liitoksissa toisiinsa. Tämä tarkoittaa sitä, että testit ovat osa liiketoimintavaatimuksia ja valtaosa hyväksyntätesteistä suunnitellaan jo vaatimusmäärittelyn yhteydessä. Näin vaatimuksia voidaan hyödyntää suoraan testauksessa, jolloin testit tukevat väistämättä paremmin projektin päämääriä (Itkonen & Kauppinen 2011). Toisaalta liiketoimintavaatimukset tulisi tunnistaa ja ymmärtää mahdollisimman laaja-alaisesti jo projektin alkuvaiheessa, sillä ne toimivat järjestelmäkehityksen keskeisinä ajureina

(Brahmkshatriya 2007; Rainardi 2008, s. 480). Brahmkshatriya (2007) tulkitsee toiminnallisen testauksen liiketoimintavaatimusten testaukseksi, jossa käydään läpi vaatimusmääritysten asianmukaisuutta. Liiketoimintavaatimusten tulisi olla selkeästi, yksiselitteisesti ja johdonmukaisesti määriteltyjä (Brahmkshatriya 2007). Näin ollen testauksessa huomiota tulisi kiinnittää erityisesti liiketoimintavaatimusten mahdollisiin puutteisiin, ymmärrettävyyteen ja siihen, vastaavatko ne todellisuudessa liiketoimintaa. Lisäksi tulisi selvittää, ovatko liiketoimintavaatimukset ylipäätään toteutettavissa ja testattavissa eli saadaanko lähdejärjestelmistä tarvittava data esimerkiksi raportoinnin vaatimuksia ajatellen (Brahmkshatriya 2007). Tämä voi edellyttää tiedon profilointia ja kattavaa lähdejärjestelmäanalyysia ennen järjestelmäprojektin aloitusta. Tiedon profiloinnissa tunnistetaan muun muassa tietokenttien arvoalueet ja arvoalueiden jakaumat (Vucevic & Zhang 2011, s. 86). Ensimmäisessä vaiheessa tulisi myös validoida erinäiset oletamukset, joita projektiin liittyy (Vucevic & Zhang 2011, s. 48).

Tietovarasto- ja raportointijärjestelmien testausprosessin toinen vaihe on testaussuunnitelman laatiminen, arviointi ja katselmointi. Testaussuunnitelman sijasta vaiheen tulisi lähteä testausstrategian määrittelystä, jolloin joudutaan pohtimaan karkealla tasolla muun muassa seuraavanlaisia kysymyksiä:

- miksi ylipäätään testataan eli mitä testauksella tavoitellaan?
- mitä testataan eli mikä on olennaista projektin kannalta?
- miten testataan eli millä tavoin päästään parhaisiin tuloksiin?

Näin ollen testauksen tulisi olla tavoitteellista toimintaa ja sen olemassaolo on pystyttävä perustelemaan. Testausstrategian tulisi tukea projektin päämääriä ja toisaalta organisaation liiketoimintastrategiaa, jolloin tärkeintä on selvittää liiketoiminnan avaintekijät ja yhdistettävä nämä testausstrategiaan, mikä edellyttää jatkuvaa kommunikointia eri osapuolten välillä (Vucevic & Zhang 2011, s. 47). Testausstrategian tulisi määrittää eri kehityselinkaaren vaiheissa suoritettavat prosessit, vastuut ja ympäristöt, jotta toimitettava järjestelmä tuottaisi odotetunlaista arvoa tilaajalleen. Testausstrategian tulisi käydä ilmi testaussuunnitelmasta eli huolellisesti laadittu testaussuunitelma ottaa kantaa testaukseen myös strategisesta näkökulmasta. Toisaalta testaussuunnitelmasta tulisi käydä ilmi myös käytettävissä olevat resurssit, kuten testaajat ja työkalut, riskit ja niihin varautuminen, aikataulu, testityypit ja menetelmät sekä testiympäristöt (Vucevic & Zhang 2011, s. 48).



Kuva 4.4. Prosessimalli mittareiden määrittämiseksi ja käyttöönottamiseksi

Testaussuunnitelman osalta eräs olennainen tekijä on selvittää, mitä onnistumisella tarkoitetaan ja millaisten mittareiden avulla onnistumista voidaan arvioida. Lisäksi tehokas testaus edellyttää testin ja toisaalta testauksen läpäisyyn vaadittavien kriteerien määrittelyä jo aikaisessa vaiheessa testausprosessia. Nämä niin kutsutut hyväksymiskriteerit tulisivat olla verifioitavissa ja mitattavissa, mikä taas edellyttää asianmukaisten mittareiden valintaa. Golfarellin & Rizzin (2009) mukaan tehokas prosessimalli mittareiden ja hyväksymiskriteerien määrittämiseksi on viisivaiheinen. Tämä prosessimalli esitetään kuvassa 4.4. Tanuska et al. (2009) pitävät tärkeänä tällaisen mittariston määrittämistä osana testausprosessia. Kyseessä on kuitenkin haastava tehtävä, sillä eri käyttäjät tyypillisesti suosivat eri ominaisuuksia tai suoritusarvoja (Kamal & Nakul 2013; Tanuska et al. 2009). Hyväksymiskriteerit voivat liittyä esimerkiksi tiedon laatuun tai suorituskykyyn. Suorituskykytestauksen kohdalla hyväksymiskriteerillä voidaan tarkoittaa esimerkiksi tietyn kyselyn maksimivasteaikaa tai ETL-testauksen yhteydessä aikaikkunaa,

jossa tietyn ETL-latauksen on suoriuduttava sille asetetusta tehtävästä. Hyväksymiskriteerien määrittäminen ja kustannusten kurissa pitäminen on kuitenkin erittäin haastava tehtävä, sillä testauksen riittävyyttä on hankala arvioida.

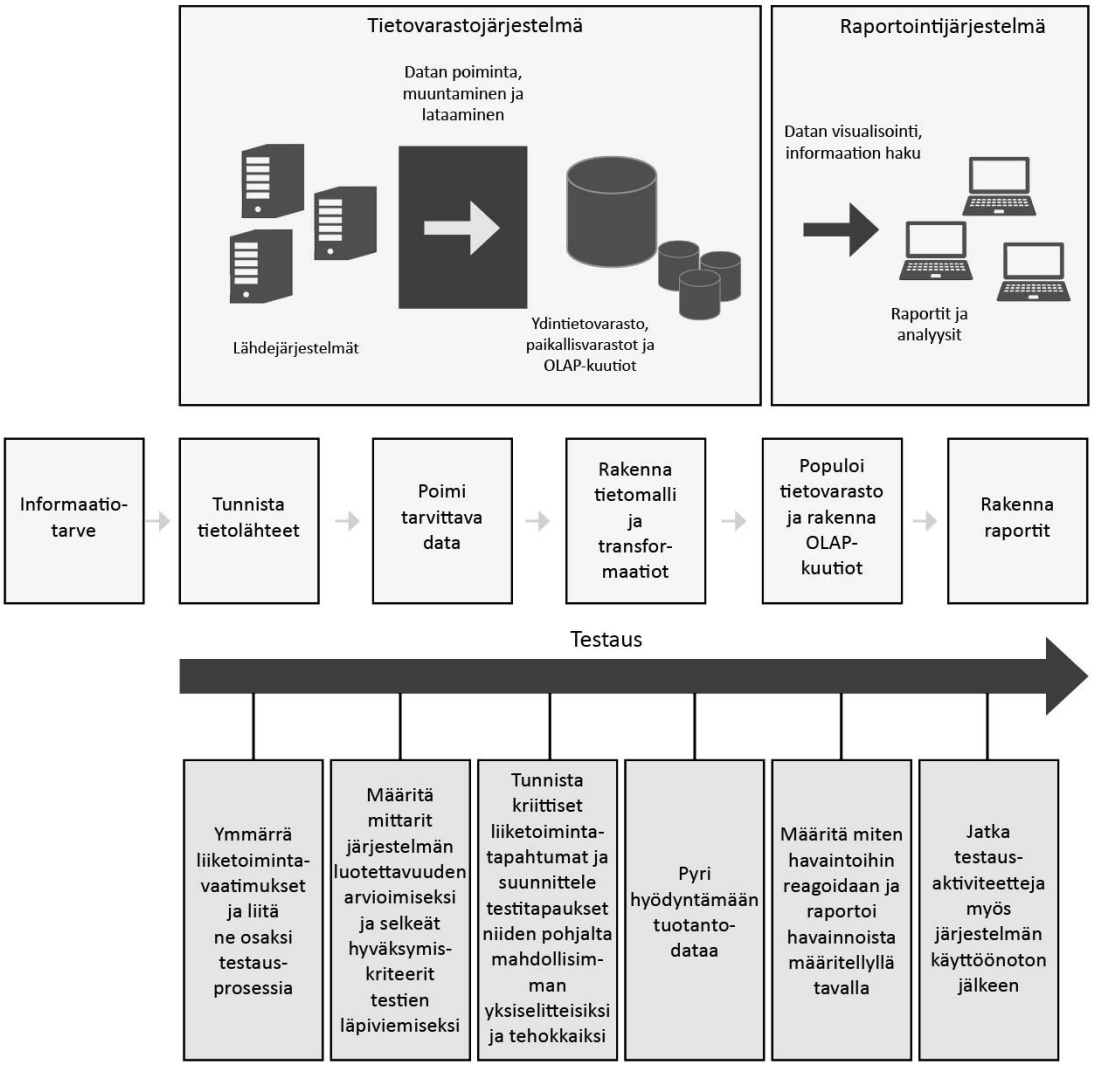
Esimerkiksi Bill Hetzel on kirjoittanut aikanaan kirjassaan *The Complete Guide to Software Testing*, että testaus saatetaan loppuun vasta, kun varsinainen hyväksyntätestaus voidaan suorittaa ongelmitta. Vucevic & Zhang (2011, s. 129) sen sijaan esittävät, että testauksen loppuun saattaminen pohjautuu käsitykseen testauksen kattavuudesta. Tämä tarkoittaisi käytännössä sitä, että testaus lopetetaan ja siirrytään hyväksyntätestaukseen, kun tietty prosenttiosuus testitapauksista on suoritettu. Vaarana tällaisessa lähestymistavassa on kuitenkin se, että testitapaukset suunnitellaan mahdollisimman nopeasti ja helposti suoritettaviksi (Vucevic & Zhang 2011, s. 129).

Testitapausten laatiminen on testausprosessin kolmas keskeinen vaihe. Se on tehtäväkokonaisuus, jossa kirjataan tarvittavia tehtäviä tai aktiviteettejä tietyn testin suorittamiseksi. Testitapausten tulisi antaa vastaus siihen, mitä testataan, miten testataan ja millaisia tuloksia voidaan odottaa - testaus ilman tietoa odotettavista tuloksista on kokeilua. (Vucevic & Zhang 2011, s. 127.) Testitapausten laadinnassa tulisi ottaa huomioon syöteavaruuden epäkelvolliset tapaukset, jolloin testauksella pystytään arvioimaan tekeekö järjestelmä jotain sellaista, mitä sen ei pitäisi määrittelyiden mukaan tehdä (Vucevic & Zhang 2011, s. 127). Testitapausta ei kuitenkaan tulisi suunnitella siten, että se sisältäisi vain epäkelvollisia syötteitä. Tällöin olisi vaikea tunnistaa eri syötteiden synnyttämät vasteet toisistaan (Vucevic & Zhang 2011, s. 128). Näin ollen testitapauksen tulisi sisältää sekä kelvollisia että epäkelvollisia syötteitä, jotka valitaan rajapintojen läheisyydestä. Mathenin (2010) mukaan rajatapausten käsittely on ehdottoman tärkeää testitapauksissa ja muun muassa tietokenttien arvoalueet ja -rajat tulisi selvittää suorittamalla tiedon profilointia jo kehitysprojektin alkuvaiheessa. Työmäärän ja testitapausten vähentämiseksi, testitapausten tulisi testata useampia toimintoja saman aikaisesti (Vucevic & Zhang 2011, s. 128). Tällainen lähestymistapa antaa mahdollisuuden havaita virheitä eri toimintojen välillä. Testitapausten suunnittelun ja laatimisen tulisi olla systemaattinen prosessi, jossa pystytään tunnistamaan kriittiset testitapaukset ja eliminoimaan ekvivalentteja testitapauksia ja joka lähtee liikkeelle liiketoiminnan vaatimuksista, käyttötapauksista ja teknisistä määrittelyistä (Vucevic & Zhang 2011, s. 128).

Testausprosessin neljäs ja viides vaihe liittyvät melko tiiviisti toisiinsa. Testausympäristön rakentaminen ja konfiguroiminen sekä testidatan hankinta on kokonaisuus, jossa voidaan helposti tehdä kriittisiä virheitä. Testausympäristön pystyttämiseen ei ole teoriassa otettu kantaa, vaan huomio kiinnittyy yleensä testidatan hankintaan. Testidatan hankinta on usein haastava tehtävä, sillä testidatan tulisi kattaa kaikki mahdolliset liiketoimintatapahtumat liiketoimintavaatimusten piirissä (Tanuska et al. 2009; Vucevic & Zhang 2011, s. 182). Tämä tarkoittaa sitä, että testitapauksia laadittaessa tulisi ymmärtää, ettei tietovarastoa pystytä käytännössä testaamaan täydellisesti kohtuullisin panok-

sin (Tanuska et al. 2009). Jos tarvittavaa testidataa ei saada lähdejärjestelmistä, voidaan Mookerjean & Malisettyn (2008) mukaan hyödyntää esimerkiksi simuloitua dataa, joka muistuttaa tuotantodataa. Ennen kuin dataa simuloidaan tyhjästä, tulisi tunnistaa dataan liittyvät rajoitteet kuten arvoalueet ja riippuvuudet (Kamal & Nakul 2013). Kamal & Nakul 2013 ja Tanuska et al. (2009) suosittelevat testidatan rajoittamista usein jonkin tietyn parametrin suhteen, jotta testidatan hallittavuus ja ylläpitäminen olisi helpompaa. Tällainen parametri voi olla esimerkiksi prosenttiosuus tai aikaväli. Pienempi otos mahdollistaa ETL-kerroksen muunnosprosessien tehokkaamman testauksen. Tulosten katselmoinnin ja mahdollisten korjausten jälkeen tulisi vastaavasti käyttää koko tuotantodataa testidatana (Tanuska et al. 2009).

Ennen testien suoritusta laaditaan vielä testiennusteet ja arvioidaan odotettavia tuloksia, mikä tarkoittaa käytännössä sopivan vertailuaineiston määrittämistä. Tieteellisessä kirjallisuudessa ei kuitenkaan anneta vastausta siihen, mitä tällaisen vertailuaineiston määrittämisessä tulisi erityisesti ottaa huomioon. Jokatapauksessa, tämän jälkeen testit ovat valmiita suoritettaviksi. Tässä kohtaa tulisi muistaa, että testitapausten suorittamiseen kuuluu tulosten arviointi ja raportointi. Raportoinnilla tarkoitetaan käytännössä sitä, miten testauksen tulokset (havaitut virheet) analysoidaan ja kommunikoidaan eri osapuolille (Vucevic & Zhang 2011, s. 182).



Kuva 4.5. Kokonaiskuva testausprosessista (mukailtu lähteestä Kamal & Nakul 2013)

Viimeisessä eli käyttöönottovaiheessa järjestelmän toimintaa tulisi seurata tuotantoympäristössä ja jatkaa testejä mahdollisten ongelmien havaitsemiseksi (Mookerjea & Malisetty 2008). Kuvassa 4.5 havainnollistetaan testausprosessin kokonaiskuva.

4.4 Lähestymistavat järjestelmäkerrosten testaamiseksi testausvaiheittain

Tietovarasto- ja raportointijärjestelmien testauksesta on tieteellisesti tutkittua tietoa vielä suhteellisen vähän, eikä aihe ole saanut yleisesti kovin suurta huomiota. Tutkitun tiedon määrä kuitenkin lisääntyy sitä mukaa, kun kyseisten järjestelmien liiketoimintakriittisyys kasvaa ja yritykset alkavat hiljalleen tunnistaa muun muassa tietovarasto- ja raportointijärjestelmiensä puutteet. (ElGamal et al. 2011; Gupta et al. 2012.) Lähestymistavat testaukseen vaihtelevat hieman tutkimuksesta riippuen, mutta kaikista löytyy yhtäläisyyksiä toistensa suhteen. Eri lähestymistavat ottavat eri tavalla kantaa siihen, millai-

sia testauksen eri osa-alueita tietovarasto- ja raportointijärjestelmän testaukseen tulisi sisällyttää, mitä testauksessa tulisi ottaa huomioon eri testaustasoilla, ja mihin järjestelmän komponentteihin, kokonaisuuksiin tai tasoihin eri testityyppejä kannattaisi soveltaa.

Testityypillä tarkoitetaan tämän tutkimuksen osalta joukkoa testausaktiviteettejä, joilla on toistensa suhteen samankaltaisia tai yhteisiä erityispiirteitä tai ominaisuuksia ja jotka on ryhmitelty siten, että ne arvioivat jotain tiettyä laatuominaisuutta (Pyhäjärvi & Pöyhönen 2005). Pyhäjärven & Pöyhösen (2005) mukaan testityyppi voi sijoittua yhdelle tai useammalle testaustasolle ja liittyä useampaan testausvaiheeseen. Oleellista on ymmärtää, etteivät kaikki testityypit ole aina tarpeellisia, vaan ne muodostavat ainoastaan potentiaalisen tarkistuslistan järkevistä vaihtoehdoista tai tapauksista, jotka tulisi kattaa testauksessa.

Lähestymistavan muodostamisessa ja eri testaustyyppien soveltamisessa tulisi kuitenkin huomioida aina tapauskohtaiset järjestelmän laatuominaisuuksien suhteelliset painoarvot. Tieteellisessä kirjallisuudessa ja tutkimuksissa esiin nousevat erityisesti järjestelmän toiminnallisuus, luotettavuus, suorituskyky, käytettävyys ja turvallisuus (Bhat 2007; Brahmkshatriya 2007; Golfarelli & Rizzi 2009; Golfarelli & Rizzi 2011; Gupta et al. 2012; Mathen 2010; Mookerjea & Malisetty 2008; Rainardi 2008; Srivastava 2011; Tanuska et al. 2009; Vucevic & Zhang 2011). Mookerjea & Malisetty (2008) korostavat liiketoiminnan ymmärrystä ja liiketoimintavaatimusten merkitystä testaussuunnitelman ja testitapausten laadinnassa. He korostavat myös liiketoimintavaatimusten merkitystä ETL-kerroksen testauksessa ja esittävät muutamia olennaisia perussääntöjä, joita tulisi noudattaa yleisesti erityisesti tietohäviöiden eliminoinnissa, ETL-logiikassa ja tiedon validoinnissa. Toisaalta Vucevic & Zhang (2011, s. 48) tunnustavat eri testaustyyppien merkittävyyden, mutta pitävät arvokkaimpina tietovarasto- ja raportointijärjestelmien testauksessa yksikkötestejä, toiminnallisia testejä, integraatiotestejä ja regressiotestejä.

Yaddowin (2009) mukaan tehokkaassa tietovarasto- ja raportointijärjestelmien testauksessa tulisi taas keskittyä ETL-kerroksen, tietovaraston ja raporttien testaukseen. Niiden tulisi sisältää sekä toiminnallisia että ei-toiminnallisia testejä, joissa tarkastellaan kerralla eri osia erillisinä kokonaisuuksina tai kaikkia osia yhtenä kokonaisuutena. Rainardi (2008) esittää Yaddowin (2009) kaltaisesti kuusi osa-aluetta, joihin testauksessa tulisi kiinnittää huomiota. Näitä osa-alueita ovat ETL-kerroksen testaus, tietovaraston toiminnallinen testaus, järjestelmän suorituskyvyn testaus, järjestelmän tietoturvan testaus, hyväksyntätestaus ja järjestelmän end-to-end-testaus. Melko samanlaista jaottelua seuraa myös Brahmkshatriya (2007), joka noudattaa ohjelmistotestauksessa yleisesti hyväksyttyjä testaustasoja ja luokittelee testauksen neljään eri kategoriaan: vaatimusten testaukseen, yksikkötestaukseen, integraatiotestaukseen ja hyväksyntätestaukseen. Vaatimusten testauksessa huomio tulisi kiinnittää liiketoimintavaatimuksiin. Yksikkötestauksen painopiste on ETL-kerroksen testauksessa ja logiikassa, jolla tieto integroidaan

tietovarastoon. Integraatiotestauksessa Brahmkshatriya (2007) korostaa alkulatauksen ja inkrementaalisten latausten toiminnan testaamista. Hyväksyntätestausta pidetään viimeisenä testausvaiheena, eikä ei-toiminnallisia osa-alueita näin ollen oteta huomioon.

Vastaavasti Bocarsly (2011) korostaa automaation merkitystä testauksessa ja nostaa esiin kaksi olennaista tietovarasto- ja raportointijärjestelmien testauksen osa-aluetta: tiedon validoinnin ja suorituskykytestauksen. Bhat (2007) taas jakaa tietovarasto- ja raportointijärjestelmien testausmetodologian V-mallin mukaisesti yksikkötestaukseen, integraatiotestaukseen, järjestelmätestaukseen ja hyväksyntätestaukseen. Lisäksi huomiota kiinnitetään tekniseen testaukseen ja toimintavalmiustestaukseen. Testausmetodologian keskeinen teema on testidatan manipuloiminen ja liiketoimintavaatimusten jäljitettävyyden hyödyntäminen laajan testikattavuuden saavuttamiseksi sekä asianmukaisten työkalujen soveltaminen testauksen suorittamisen ja arvioinnin nopeuttamiseksi.

Hyvien käytäntöjen mukaisessa tietovarasto- ja raportointijärjestelmien testauksessa tulisi siis käydä läpi kolme kokonaisuutta, joita ovat ETL-kerroksen testaus, varastointikerroksen testaus ja raportointikerroksen testaus. Järjestelmäkerroksia tulisi testata yksikkötasolla ja lisäksi koko järjestelmän osalta tulisi ainakin ottaa huomioon integraatio- ja hyväksyntätestaus. Lisäksi testauksessa tulisi huomioida suorituskyvyn ja tietoturvan testaus sekä mahdollisuudet testien automatisoimiseksi.

4.5 ETL-kerroksen testaus

ETL-kerroksen testaus on olennainen testauksen osa-alue, sillä ETL-kerroksella integroidaan tietoa eri lähdejärjestelmistä tietovarastoon (Rainardi 2008, s. 478; Tanuska et al. 2009). Jos transformaatiot eivät noudata liiketoimintasääntöjä tai ne sisältävät virheitä ja puutteita, tietovarastoon saattaa kertyä kelvotonta tietoa, mikä voi vaarantaa koko tietovarasto- ja raportointijärjestelmän toiminnan (Rainardi 2008, s. 478).

ETL-kerroksen kehitys vie tyypillisesti suurimman osan työpanoksesta suhteessa muihin järjestelmäkerroksiin ja edellyttää näin suhteessa suurempaa panosta testaukseen. Brahmkshatriya (2007) mieltää ETL-kerroksella tapahtuvan testauksen tyypillisesti yksikkötestaukseksi luonnehtien sitä valkolaatikkotestaukseksi, joka suoritetaan kehittäjän toimesta. Tanuska et al. (2009) tukevat tätä teoriaa ja esittävät, että ETL-kerroksen testaus kokonaisuutena koostuu skriptien tai moduulien (yksikköjen) testauksesta, mikä pitää sisällään tiedon laadun eri ulottuvuudet kuten tiedon eheyden ja johdonmukaisuuden.

ETL-kerroksen testauksessa tulisi ensisijaisesti kiinnittää huomiota lähdetiedon ajantasaisuuteen, ETL-logiikan oikeellisuuteen ja toiminnan luotettavuuteen, massa- ja päivittäislatausten toiminnan luotettavuuteen toimintaperiaatteesta riippumatta, tietohäviöihin sekä järjestelmän palautettavuuteen (Kamal & Nakul 2013; Mookerjea & Malisetty

2008; Rainardi 2008, s. 478-479). ETL-logiikat tulisi validoida poimittaessa tietoa, muunnettaessa tietoa sekä ladattaessa tietoa (Kamal & Nakul 2013). Rainardin (2008, s. 479) mukaan ETL-logiikkaa ja transformaatioiden toimintaa tarkasteltaessa tulisi testatessa tarkastella yksittäisiä ETL-töitä sekä niiden työnkulkua ja suoritusjärjestystä riippuvuussuhteet huomioiden. Toisaalta tietueiden ja erilaisten transformaatioiden korkea lukumäärä voi osoittautua testauksen kannalta haasteelliseksi (Srivastava 2011). Koska suuren tietomassan validoiminen tietue kerrallaan on käytännössä mahdotonta, tulisi huomio kiinnittää ensisijaisesti transformaatioihin ja niiden logiikan oikeellisuuteen ja toiminnan luotettavuuteen.

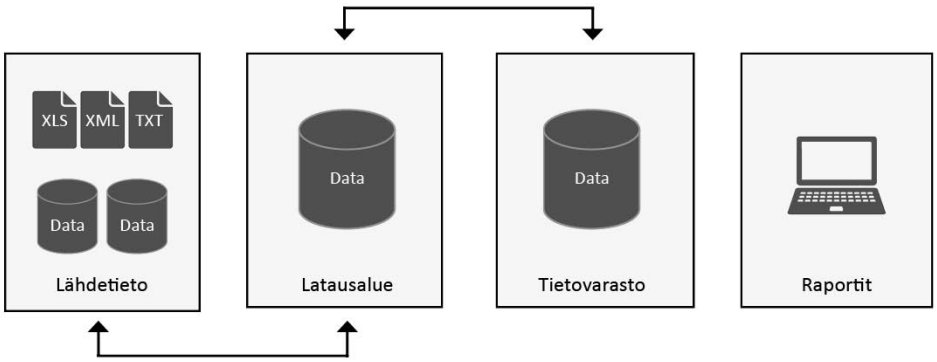
4.5.1 Transformaatioiden ja tiedon validointi

ETL-kerroksen testauksessa tulisi siis varmistaa, että transformaatiot eli muunnosprosessit toimivat kokonaisuudessaan asianmukaisesti ja noudattavat liiketoimintasääntöjä (Mookerjea & Malisetty 2008), mikä voi usein olla ETL-kerroksen testauksen vaativin osuus (Theobald 2007). Mathen (2010) pitää tärkeänä erityisesti sitä, miten liiketoimintasäännöt muunnetaan transformaatioiksi eli millä tavalla liiketoimintasäännöt implementoidaan osaksi ETL-prosesseja. ETL-prosessi voi käytännössä sisältää useita, ETL-sekvenssejä. Yksittäinen ETL-sekvenssi voi olla esimerkiksi suora siirto, yksinkertainen muunnosprosessi tai monimutkainen muunnosprosessi, johon voi liittyä useita erillisiä ETL-töitä. Koska liiketoiminta määrittää käytettävät liiketoimintasäännöt, eivätkä ETL-kehittäjät tunne lähtökohtaisesti hyvin liiketoimintaa, on riski virheiden syntymiseen korkea. Srivastava (2011) suositteleekin, että kaikki transformaatiot testattaisiin virheiden varalta. Myös kertaluonteiset erityistapaukset tulisi käsitellä (Mathen 2010). Tämä johtuu muun muassa siitä, että moni liiketoimintaprosessi sietää virheitä huonosti, eikä liiketoimintakriittisten päätösten tekeminen virheellisen tiedon pohjalta johda välttämättä hyviin lopputuloksiin.

Näin ollen mitä monimutkaisempi transformaatio on kyseessä, sitä enemmän testausta se edellyttää (Srivastava 2011). Testauksessa tulisi kuitenkin muistaa, ettei keskeneräisten ETL-töiden tai transformaatioiden testaaminen ole järkevää (Mookerjea & Malisetty 2008; Srivastava 2011).

Transformaatioiden testauksen ohella Theobald (2007) ja Vucevic & Zhang (2011) pitävät tiedon validointia keskeisimpänä testausmenetelmänä ETL-kerroksen yksikkötestauksessa. Itseasiassa ETL-kerroksen testaus on suurimmaksi osaksi juuri tiedon validointia (Kamal & Nakul 2013; Mathen 2010). Tämä tarkoittaa käytännössä tietovarastoon ladatun datan oikeellisuuden, johdonmukaisuuden ja täydellisyyden arviointia muun muassa vertailemalla rivimääriä lähde- ja kohdetaulujen välillä sekä validoimalla summatietoja tietovarastossa lähdetietojen pohjalta (Kamal & Nakul 2013; Mathen 2010; Vucevic & Zhang 2011). Mathenin (2010) mukaan ensisijaisena tavoitteena tulisi olla tiedon täydellisyyden validoiminen. Näin varmistetaan, että tietovarastoon ladattujen tietueiden tietosisältö on siirtynyt kokonaisuudessaan, eikä tietoa ole esimerkiksi

hävinnyt. Tiedon täydellisyys on otettava huomioon myös erityistapauksissa, jotka voivat liittyä esimerkiksi NULL-arvoihin tai tahattomasti katkaistuihin merkkijonoihin. Tiedon validointia tulisi tehdä tietovarasto- ja raportointijärjestelmän arkkitehtuurin mukaisesti ainakin kahden eri pisteen välillä. Kuvassa 4.6 havainnollistetaan tietovarasto- ja raportointijärjestelmän keskeiset järjestelmäkerrokset ja niiden välillä tapahtuva tiedon validointi.



Kuva 4.6. Tiedon validointi eri järjestelmäkerrosten välillä (mukailtu lähteestä Bocarsly 2011)

ETL-kerroksella tietoa tulisi ensisijaisesti validoida lähdejärjestelmien ja latausalueen sekä latausalueen ja tietovaraston välillä (Bocarsly 2011). Tilanteesta riippuen tiedon validoinnin tarpeellisuutta tulisi myös arvioida latausalueella sijaitsevien eri välivaiheiden välillä. Lähde- ja kohdetaulujen välisiä vertailuja voidaan toteuttaa hyödyntämällä esimerkiksi suoria SQL-kyselyitä tai tarkistussummia (Mathen 2010). Mikäli tiedon siirroissa hyödynnetään tiedostoja, voidaan vertailu toteuttaa esimerkiksi perinteisissä taulukkolaskentaohjelmistoissa erilaisten sisäänrakennettujen funktioiden avulla.

Tiedon validoinnissa voidaan hyödyntää useampia tekniikoita tai menetelmiä myös samanaikaisesti (Mathen 2010). Eräs tehokas tiedon validoinnin menetelmä on näytteenotto (engl. sampling), jossa poimitaan testidatasta ennalta määritelty otos vertailtavaksi ja muunnosten validointia varten. Mathen (2010) suosittelee tiedon profiloinnin yhdistämistä osaksi tiedon validointia, mikä mahdollistaa laajemman testikattavuuden saavuttamisen.

Myös Srivastava (2011) suosittelee ETL-kerroksen testauksessa käytettäväksi näytteenottomenetelmää ja toisaalta myös tiedon validointia varten SQL-funktioiden hyödyntämistä. Näytteenotto suuresta tietomassasta on yleisesti hyvä tekniikka, jossa testaaja poimii tietomassasta muutamia tietueita ja varmistaa, että ne noudattavat liiketoimintasääntöjä ja muunnosprosessien määrityksiä (Srivastava 2011). Esimerkkinä voidaan käyttää tapausta, jossa jokin tietty ETL-prosessi muuntaa työntekijöiden tietoja

sisältävään kohdetauluun sijainniksi vakioarvon ”Helsinki” niille tietueille, joiden yksilöivä tunniste on välillä 1000-2000. Testaaja poimii tietomassasta kymmeniä tietueita sattumanvaraisesti siten, että liiketoimintasäännön rajatapaukset otetaan huomioon. Tämä tarkoittaa tietueita, joiden yksilöivä tunniste on lähellä raja-arvoja 1000 ja 2000. Jos valittujen tietueiden validointi ei paljasta ongelmia, tietomassa on todennäköisesti populoitu oikein, eikä ETL-logiikkaa tarvitse muuttaa tai korjata. Testauksessa tulisikin aina kiinnittää erityistä huomiota rajatapauksiin, sillä sieltä löytyy todennäköisimmin virheitä. Tällainen lähestymistapa edellyttää kuitenkin paljon manuaalista työtä ja toisaalta syvällistä ymmärrystä transformaatioista sekä liiketoiminnasta. Näin ollen pitkällä tähtäimellä paras ratkaisu olisikin automatisoida tällaisten transformaatioiden validointeja.

Näytteenoton ohella toinen tapa on tarkastella kohdetietokannan tauluja ja verrata arvoja liiketoimintasääntöihin. Edellisen esimerkin tapauksessa tämä voisi tarkoittaa esimerkiksi seuraavanlaisen, yksinkertaisen SQL-hakuoperaation osoittamista kohdetauluun, mikä listaisi kaikki sijainnin saamat arvot yksilöivän tunnisteen ollessa välillä 1000 ja 2000:

> ***select*** *distinct (sijainti)* ***from*** *kohdetaulu* ***where*** *tunniste between (1000 and 2000);*

Tiedon validoinnissa tulisi varmistaa myös, ettei tietovarastoon ole päässyt syntymään duplikaatteja eli identtisiä tietueita ja että datan eheys säilyy aina muunnosprosesseista ja latausten toimintaperiaatteista huolimatta.

4.5.2 Tietohäviöiden ja järjestelmän palautettavuuden testaaminen

Aina, kun tietoa poimitaan lähdejärjestelmien relaatiotietokannoista, tulisi kiinnittää huomiota ETL-kerroksella tapahtuviin mahdollisiin tietohäviöihin, sillä puutteellinen tieto laskee merkittävästi tietovaraston luotettavuutta (Mookerjea & Malisetty 2008; Rainardi 2008, s. 187). Mookerjea & Malisettyn (2008) mukaan tämä tarkoittaa tietueiden eli rivien ja tietueiden sarakkeiden mukaisen sisällön validointia. Ajatellaan esimerkiksi tilannetta, jossa lähdejärjestelmässä suoritetaan tietovarastoinnin kannalta merkittäviä tapahtumia, jotka aiheuttavat muutoksia datassa. ETL-prosessissa poimitaan, muunnetaan ja ladataan dataa tietovarastoon tietyllä frekvenssillä eli ajallisesti tietyllä taajuudella. Kun kyseinen prosessi on suoritettu riittävän monta kertaa, voidaan ryhtyä vertailemaan lähdejärjestelmiä ja tietovarastoa keskenään. Hyvä keino tunnistaa puuttuvia tietueita on käyttää apuna tarkistussummia. Jos esimerkiksi *TAULU_1* sisältäisi tietovarastoon ladattua dataa ja *TAULU_2* sisältäisi lähdejärjestelmästä löytyvää dataa, tarkistussummia voitaisiin verrata taulujen kesken tarkistussumman osoittaman sarakkeen *CHECKSUM* avulla:

> **select * from** *taulu_2* **where** not exists (**select * from** *taulu_1* **where** *taulu_1*.checksum=*taulu_2*.checksum)

Vertailu voitaisiin tehdä myös päinvastaisesti eli niille tietueille tai riveille, jotka ovat taulussa *taulu_1*, mutta eivät taulussa *taulu_2*.

> **select * from** *taulu_1* **where** not exists (**select * from** *taulu_2* **where** *taulu_1*.checksum=*taulu_2*.checksum)

Poikkeamat antavat viitteitä virheistä tai puutteista ETL-kerroksella. Jos puutteita tarkistussummien vertailussa ei ilmene, voidaan ETL-prosessia käytännössä pitää luotettavana tietovuotojen suhteen. Rainardin (2008, s. 187) mukaan hyviin käytäntöihin kuuluu, että tietovuotojen testaamista jatketaan kehityksen ohessa. Näin pystytään seuraamaan ETL-prosessien toimintaa ja tekemään vertailuja suuremmilla volyymeilla. Rainardin (2008) mukaan eräajojen lisäksi ETL-kerroksen testauksessa tulisi käydä läpi myös alkulataus ja mahdolliset muut massalataukset tietovuotojen varalta.

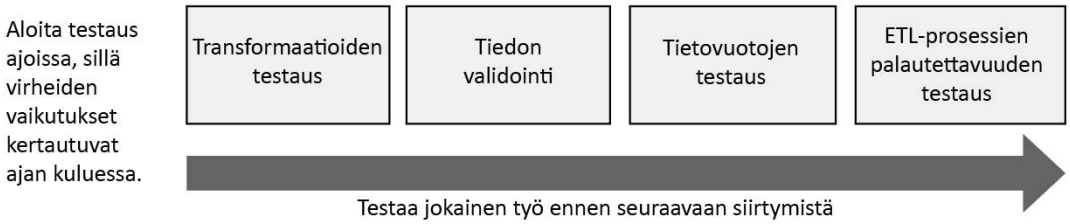
Tietohäviöihin liittyy olennaisesti palautettavuus. Palautettavuus tarkoittaa ETL-prosessin robustisuutta eli järjestelmän palautumiskykyä virhetilanteista ilman tietohäviöitä tai vahingollisia muutoksia tietovarastoon ladatussa tiedossa (Golfarelli & Rizzi 2009). Ennen palautettavuuden testausta on määritettävä tapahtumatyypit, joilta halutaan erityisesti suojautua. (Rainardi 2008, s. 479.) Tällainen voi olla esimerkiksi olla ETL-prosessin ajon aikainen sähkökatkos tai tallennuslaitteen häiriö. Tapahtumatyyppien määrityksen jälkeen tulisi testata ne ETL-kerroksen prosessityypit, joissa tapahtumat (esim. tiedon poiminta ja populointi) voivat aiheuttaa virheitä. Jokaisen tapahtuma- ja prosessityypin välinen kombinaatio tulisi simuloida, minkä jälkeen kyseisen prosessityypin sisältävä ETL-prosessi ajetaan uudelleen. Tämän jälkeen voidaan tarkistaa, mitä ajossa tapahtui ja ilmaantuiko tietohäviöitä. Jos ETL-kerroksella käytetään virhetilanteiden automatisoitua raportointia, täytyy varmistua siitä, että oikeat henkilöt saavat asiasta tiedon oikealla hetkellä. (Rainardi 2008, s. 479.)

Kuten jo aiemmin on todettu, ETL-kerroksen testaus on pitkälti yksikkötestausta. Se tulisi aloittaa tyypillisesti siinä vaiheessa, kun testaussuunnitelma on olemassa, testattavan moduulin rakenne on arvioitu ja tarvittava verifiointi koodin osalta on saatettu loppuun (Vucevic & Zhang 2011, s. 73). Jos edellä mainitut esivalmistelut toteutetaan asianmukaisesti, pystytään suurin osa virheistä havaitsemaan ja korjaamaan jo kehityselinkaaren aikaisessa vaiheessa. Toisaalta, jos virheitä ei havaita yksikkötestauksessa, ne aiheuttavat ongelmia myöhemmissä elinkaaren vaiheissa, ja tästä koituu suurella todennäköisyydellä myös huomattavia kustannuksia.

Testiympäristön pystyttäminen ei aina ole ongelmatonta, sillä tuotantoa vastaavia olosuhteita voi olla hankala luoda testiympäristössä, joka kattaisi kehityselinkaaren alkuvaiheen vaikeasti havaittavat rakenteelliset ongelmat (Vucevic & Zhang 2011, s. 73). Kehityksen alkuvaiheessa esiintyvät ongelmat ovat tyypillisesti kaikkein hankalimpia

tai vaarallisimpia, sillä ne vievät kehitystä tavallaan jatkuvasti väärään suuntaan ja johtavat suurempiin ongelmiin myöhemmissä vaiheissa. Tuotantoympäristön dynaamisuus aiheuttaa sen, että erilaiset virheet ilmaantuvat ennemmin tai myöhemmin, eikä erillisen testiympäristön avulla välttämättä saavuteta todellisuutta vastaavia olosuhteita ennen järjestelmän käyttöönottoa (Vucevic & Zhang 2011, s. 74). Tämä tulisi ottaa huomioon testausstrategiaa määritettäessä.

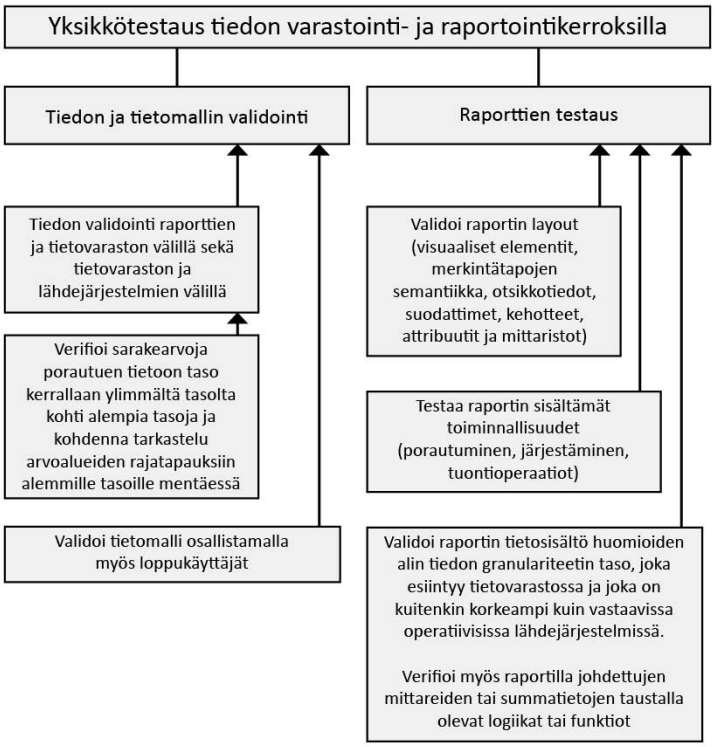
Kuvassa 4.7 tiivistetään keskeisimmät testauksen osa-alueet, jotka tulisi ottaa huomioon ETL-kerroksen yksikkötestauksessa.



Kuva 4.7. Yksikkötason testaus ETL-kerroksella

4.6 Tiedon varastointi- ja raportointikerrosten testaus

Tietovaraston testauksen avulla pyritään varmistumaan siitä, että järjestelmä täyttää sille asetetut toiminnalliset liiketoimintavaatimukset, mikä tarkoittaa käytännössä suurelta osin tietovarastossa olevan datan validointia (Rainardi 2008, s. 480). Tiedon validoinnin lisäksi Rainardin (2008, s. 481) mukaan on välttämätöntä, että testaaja ymmärtää kuinka tietovarastossa olevia arvoja verifioidaan. Tiedon varastointi- ja raportointikerroksen yksikkötestaukseen voidaan liittää myös tietomallin validointi ja raporttien testaus. Lisäksi Bocarsly (2011) esittää, että testausta tulisi automatisoida, sillä se parantaa järjestelmän laatua ja toisaalta vähentää testauksen läpivientiaikaa merkittävästi, kun automaatiosovelluksia hyödyntämällä tietoyksikköjä voidaan jäljittää aina lähdejärjestelmistä raportointisovelluksiin saakka. Jäljitettävyys helpottaa ongelmien paikallistamista ja näin edesauttaa ongelmien juurisyiden selvittämistä ja korjaamista, mikä vähentää jatkossa samantyyppisten ongelmien esiintyvyyttä (Bocarsly 2011). Kuvassa 4.8 esitetään tiivistettynä varastointi- ja raportointikerrosten yksikkötason testauksen keskeiset hyvät käytännöt.

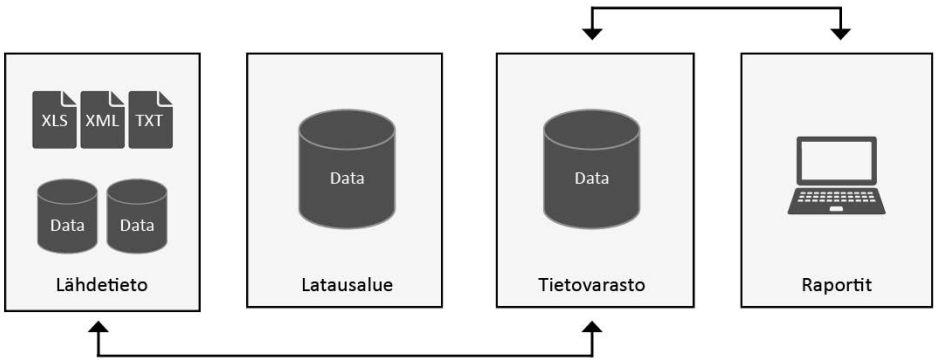


Kuva 4.8. Yksikkötason testaus varastointi- ja raportointikerroksilla

Myös OLAP-kuutioiden testaus on osa varastointi- ja raportointikerroksen testausta, mikä on valtaosin dimensioiden, hierarkioiden ja erilaisten laskentafunktioiden validointia. Myös erilaiset lennosta tehtävät

4.6.1 Tiedon ja tietomallin validointi

Tietovarasto- ja raportointijärjestelmän taustalla olevan tiedon tulisi aina olla ehdottoman luotettavaa, mikä edellyttää tiedon validointia eri järjestelmäkerrosten välillä. Tiedon validointi varastointikerroksella on tiedon vertailua ydintietovaraston ja lähdejärjestelmien välillä sekä tiedon vertailua raporttien ja ydintietovaraston välillä (Rainardi 2008, s. 480). Kuva 4.9 havainnollistaa tiedon validointia eri järjestelmäkerrosten suhteen.



Kuva 4.9. Tiedon validointi eri järjestelmäkerrosten välillä (mukailtu lähteestä Bocarsly 2011)

Koska jokaista erillistä arvoa ei voida tarkistaa, tulisi keskittyä verifioimaan summa-arvoja porautuen tietoon taso kerrallaan ylimmältä tasolta kohti alempia tasoja. Usein riittää, että alimmalla summatasolla käydään läpi ainoastaan muutamia esimerkkiarvoja (Rainardin 2008, s. 480). Mikäli esimerkiksi liiketoimintavaatimuksiin sisältyy kuukausittainen myynnin seuraaminen tuotteittain, tiedon validointi voidaan toteuttaa tarkastelemalla kokonaisliikevaihtoa, -kustannuksia ja -katetta tietyllä aikavälillä ja verifioida, että nämä kolme arvoa vastaavat lähdejärjestelmissä olevaa tietoa. Ylimmällä tasolla voidaan tuotteiden tapauksessa tarkoittaa esimerkiksi tuotekategoriaa, jolloin tietyn aikavälin tarkastelu kohdistuu tiettyä tuotekategoriaa koskeviin summa-arvoihin. Vastaavasti alimmalla tasolla voidaan tarkoittaa yksittäistä tuotekoodia, joka vastaa tiettyä tuotteen konfiguraatiota tai komponenttia. Koska hierarkiassa alemmille tasoille siirryttäessä toimialueiden määrä kasvaa, on alimmalla tasolla järkevintä kohdentaa tarkastelu arvoalueen rajatapauksiin. Tämä voi tarkoittaa tuotehierarkian tapauksessa sellaisia tuotekoodeja, joita myydään suhteessa eniten ja vähiten. Kun arvot on tarkistettu, tulisi dimensioiden jäsenten määrä verifioida kaikilla tasoilla (Rainardi 2008, s. 481). Tämä tarkoittaa yksinkertaisesti sitä, että sekä lähdejärjestelmistä että tietovarastosta voidaan tunnistaa sama määrä esimerkiksi tuotekategorioita, tuotetyyppejä ja tuotteita.

Tietovaraston osalta testauksen tulisi kattaa mahdollisimman monia liiketoimintatapahtumia. Testaus tulisi aloittaa tunnistamalla keskeisimmät liiketoimintatapahtumat, joiden puitteissa järjestelmän luotettavuudesta halutaan erityisesti varmistua. (Rainardi 2008, s. 482.) Keskeinen liiketoimintatapahtuma voi olla esimerkiksi valuuttakurssien päivitys tai resurssien uudelleenallokointi. Lisäksi testauksessa tulisi varmistua siitä, että merkittävät liiketoimintatapahtumat voidaan suorittaa tietovarastossa liiketoimintavaatimuksia rikkomatta. Tämä edellyttääkin juuri tiedon validointia lähdejärjestelmien ja tietovaraston välillä (Mookerjea & Malisetty 2008; Rainardi 2008, s. 482).

Tiedon verifiointi liittyy olennaisesti tiedon validointiin. Verifiointia tehdään vertaamalla arvoja lähdejärjestelmien ja tietovaraston välillä hyödyntäen joko suoria hakuja tai raportteja. Näin ollen testaajan tulisi tietää, mistä tietovarastoa vastaava arvo saadaan ja miten se noudetaan lähdejärjestelmistä (Rainardi 2008, s. 481). Jos esimerkiksi kokonaisliikevaihto tuotteelle X joulukuun 24. päivänä on 20 000 euroa, testaajien on selvitettävä, miten lähdejärjestelmästä voidaan hakea vastaava kokonaisliikevaihto kyseiselle päivälle ja millaisia poikkeuksia (valuuttamuunnokset, palautukset, peruutukset, konsernin sisäinen myynti, ilmaiskappaleet) siihen mahdollisesti liittyy. Näin ollen eri liiketoiminta-alueilla tarvitaan erilaisia testaajia ja henkilöitä, jotka osaavat laskea luvut lähdejärjestelmien puolella (Rainardi 2008, s. 481).

Tiedon varastointikerroksen testauksessa tulisi keskitettyä tiedon validoinnin ohella myös tietomallin validointiin. Tietovaraston noudattama tietomalli tulisi validoida loppukäyttäjien toimesta, sillä on tärkeää varmistaa, että malli täyttää käyttäjien vaatimukset ja käyttäjät kykenevät ymmärtämään miten tietovarasto toimii. (Tanuska et al. 2009.) Jos käyttäjät eivät ymmärrä tietomallia, he eivät välttämättä osaa käsitellä dataa riittävällä tasolla. Tanuska et al. (2009) mukaan testaukseen tulisi liittyä myös paljon suunnitteluvaiheen verifiointia, jolloin tulisi verifioida ainakin, että faktataulujen vierasavaimet on indeksoitu huolellisesti, yhteen faktatauluun liittyy aina yksi tai useampia dimensiotauluja ja eri hierarkiatasojen sisältämät faktat eivät sisällä NULL-arvoja.

4.6.2 Raporttien testaus

Raporttien testaus on käytännössä raporttien ulkoasun, tietosisällön ja laskukaavojen validointia. Ulkoasussa tärkeintä on kiinnittää huomiota raportilla näkyviin tietoihin, visuaalisiin elementteihin sekä eri merkintätapojen semantiikkaan. (Srivastava 2011.) Raportin visuaalisen ilmeen tulisi siis vastata määrityksiä ja kuvata raportilta vaadittuja asioita oikein, mikä tarkoittaa esimerkiksi sitä, että raportti sisältää odotetunlaiset visuaaliset elementit, otsikkotiedot, suodattimet (engl. filter), kehotteet (engl. prompt), attribuutit ja mittaristot (Kamal & Nakul 2013; Srivastava 2011; Yaddow 2009). Kamal & Nakul (2013) nostavat esiin myös raporttien toiminnallisuuksien testaamisen esimerkiksi web-pohjaisessa käyttöliittymässä. Tämä pitää sisällään muun muassa porautumisen, tietojen järjestämisen ja erilaisten tuontioperaatioiden (engl. export) läpikäyntiä.

Brahmkshatriya (2007) ja Srivastava (2011) korostavat kuitenkin raportilla olevan tiedon validointia ulkoasua tärkeämpänä tekijänä. Tiedon validointi voidaan suorittaa yksinkertaisesti tutkimalla arvoja raportilla ja etsimällä mahdollisia poikkeamia suhteessa odotettuihin arvoihin. Testaajien tulisi ottaa huomioon alin tiedon granulariteetin taso, joka esiintyy tietovarastossa ja joka on kuitenkin korkeampi kuin vastaavissa operatiivisissa lähdejärjestelmissä. Näin ollen tiedon validointi edellyttääkin kykyä jäljittää raportilla näkyvä tieto tarvittaessa aina lähdejärjestelmiin saakka. (Kamal & Nakul 2013.) Tiedon validoinnin ohella testaajien tulisi myös verifioida raportilla esiintyvien johdettujen mittareiden tai summatietojen taustalla olevat logiikat tai funktiot, jotta tulosten

oikeellisuudesta ja luotettavuudesta voitaisiin varmistua kaikissa tapauksissa (Kamal & Nakul 2013; Srivastava 2011).

4.7 Integraatiotestaus

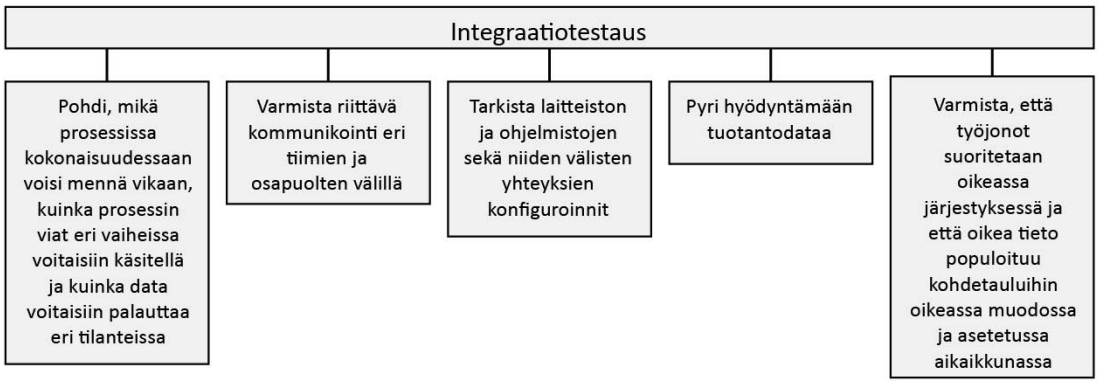
Integraatiotestauksen tarkoituksena on osoittaa, että tietovarasto- ja raportointijärjestelmä toimii kokonaisuutena luotettavasti ja tarkoituksenmukaisesti (Mathen 2010; Vucevic & Zhang 2011, s. 90). Integraatiotestauksessa nostetaan esiin ongelmia, joita syntyy eri järjestelmäkerrosten ja -sovelluskomponenttien välillä (Mathen 2010; Theobald 2007). Rainardi (2008, s. 487) kutsuu integraatiotestausta myös end-to-end-testaukseksi, jossa järjestelmä testataan ensimmäistä kertaa kokonaisuutena ennen varsinaista hyväksyntätestausta, mutta fokus on erityisesti rajapinnoissa (Theobald 2007).

Valtaosa integraatiotestauksessa löydetyistä virheistä liittyy joko järjestelmän tietosisältöön (Mathen 2010; Yaddow 2011, s. 3), tai ne ovat seurausta vääristä oletuksista komponenttien arkkitehtuurin suhteen (Mathen 2010; Theobald 2007; Yaddow 2011, s. 3). Testitapauksia laadittaessa tulisi erityisesti miettiä sitä, mikä prosessissa kokonaisuudessaan voisi mennä vikaan, kuinka prosessin viat eri vaiheissa voitaisiin käsitellä ja kuinka data voitaisiin palauttaa tai tuhota tarvittaessa (Theobald 2007; Yaddow 2011, s. 3). Vastuu integraatiotestauksesta ja siinä havaittavista ongelmista ei tulisi olla vain testaajilla, vaan kommunikointi eri osapuolten välillä on tärkeää (Theobald 2007). Koska heikko kommunikointi voi johtaa integraatiotestauksen epäonnistumiseen, kommunikointikuilujen välttämiseksi eri tiimien jäsenet tulisi koota yhteen muodostamaan testitapaukset ja keskustelemaan siitä, mikä voisi mennä vikaan tuotannossa (Theobald 2007; Yaddow 2011, s. 3).

Bhat (2007) esittää näiden lisäksi vielä hieman teknisemmän näkökulman, jonka mukaan huomiota tulisi samaan aikaan kiinnittää järjestelmän taustalla olevaan laitteistoon ja ohjelmistoihin. Bhatin (2007) ja Rainardin (2008, s. 487) mukaan integraatiotestauksessa tulisikin varmistaa ainakin, että laitteisto on asennettu ja konfiguroitu asianmukaisesti, tarvittavat yhteydet eri komponenttien välillä on konfiguroitu asianmukaisesti, ohjelmistot on siirretty asianmukaisesti testiympäristöihin ja laitteiston, tietoverkon sekä ohjelmistojen konfiguraatiot on dokumentoitu riittävällä tasolla.

Mathen (2010), Theobald (2007) ja Yaddow (2011, s. 3) korostavat tuotantodatan merkitystä integraatiotestauksessa. Lähtökohtaisesti integraatiotestauksessa tulisi hyödyntää tuotantodataa, mutta joissain tapauksissa liiketoiminnan asettamat tietoturvapolitiikat saattavat estää tuotantodatan suoran käytön testauksessa, mikä edellyttää kyseisten kenttien tai taulujen sattumanvaraista generointia (Yaddow 2011, s. 3). Tällöin oleellista on se, kuinka hyvin pystytään jäljittelemään tuotantodataa (Mathen 2010; Theobald 2007).

Integraatiotestauksessa tulisi varmistaa, että työjonot suoritetaan oikeassa järjestyksessä ja oikea tieto populoituu kohdetauluihin oikeassa muodossa asetetussa aikaikkunassa (Bhat 2007; Brahmkshatriya 2007; Tanuska et al. 2009). Tässä kohtaa tulisi validoida myös rivimäärät ja analysoida tiedon eheys siirryttäessä kerroksilta toisille (Brahmkshatriya 2007). Toisaalta oleellista on muistaa, että integraatiotestauksessa koko tietovarastointi- ja raportointiprosessi ajetaan alusta loppuun samoin riippuvuussuhtein kuin tuotannossa (Yaddow 2011, s. 3). Tämä tarkoittaa muun muassa työjonojen ajamista tuotannon mukaisessa järjestyksessä ja aikataulussa sisältäen myös aikataulutettujen töiden suorituksen valvomisen, virhe- tai häiriötilanteiden käsittelyn validoinnin sekä suorituskyvyn analysoinnin eri prosessin vaiheissa (Bhat 2007; Theobald 2007; Yaddow 2011, s. 3). Rainardi (2008, s. 487) vastaavasti korostaa automatisoitujen ja ajastettujen ETL-prosessien ajoa muutamien päivien ajan simuloimaan tuotannon olosuhteita varmistaen, että tarpeellinen tieto populoituu tietovarastoon asianmukaisesti ja tiedon eheys säilyy. Kuvassa 4.10 esitetään tiivistettynä integraatiotestauksen keskeiset hyvät käytännöt.



Kuva 4.10. Integraatiotestauksen hyvät käytännöt

4.8 Hyväksyntätestaus

Hyväksyntätestaus on järjestelmän käyttäjien suorittama testausvaihe, jossa ryhmä ennalta valittuja henkilöitä käyttää järjestelmää siten kuin sitä käytettäisiin tuotannnossa. Hyväksyntätestauksessa käyttäjien tulisi arvioida, täyttääkö järjestelmä sille asetetut vaatimukset eli suoriutuuko se päivittäisistä käyttötapauksista tai liiketoimintatapahtumista asianmukaisesti niin toiminnallisuuden kuin käytettävyydenkin näkökulmasta. (Brahmkshatriya 2007; Rainardi 2008 s. 486; Vucevic & Zhang, s. 99.) Tarkoituksena on arvioida luotettavuutta, eikä etsiä systemaattisesti virheitä. Hyväksyntätestaus tulisi suorittaa, kun järjestelmä on toimitusvalmiina ja muilta osin testattu, joten se ajoittuu kehityselinkaaren loppuun. (Rainardi 2008 s. 486; Vucevic & Zhang, s. 99.)

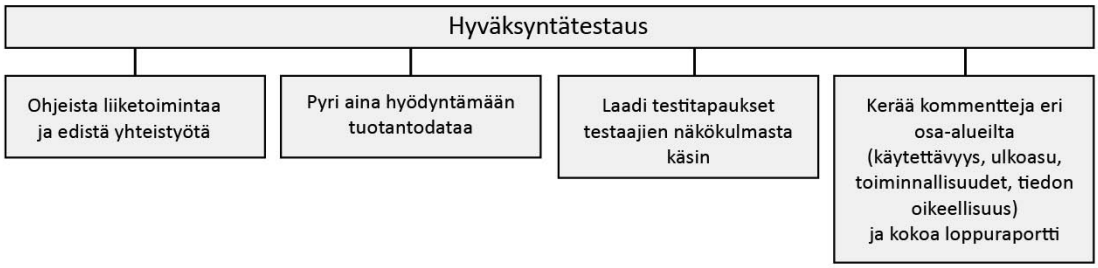
Käyttäjien osallistaminen testaukseen on ehdottoman tärkeää järjestelmän luotettavuuden ja kehitysprojektin onnistumisen kannalta, sillä he tuntevat tietosisällön ja osaavat arvioida liiketoimintavaatimusten tai tarpeiden täyttymistä parhaiten päivittäisissä rutiineissa (Theobald 2007; Vucevic & Zhang 2011, s. 99; Yaddow 2011, s. 4). Tämä ei edellytä heiltä esimerkiksi ETL-kerroksen tuntemusta, sillä fokus on toiminnallisuuksien ja järjestelmän käyttäytymisen arvioimisessa (Vucevic & Zhang 2011, s.99). Rainardin mukaan (2008, s. 486) ennen hyväksyntätestauksen aloittamista käyttäjiä tulisi ohjeistaa järjestelmän toiminnallisuuksista ja arkkitehtuurista, jotta he ymmärtäisivät millaisessa ympäristössä he operoivat. Hyväksyntätestauksen aikana vastaavasti järjestelmätoimittajan tulisi tukea käyttäjiä ohjeistamalla heitä tarvittaessa ja vastaamalla käyttäjien esittämiin kysymyksiin esimerkiksi siitä, miten data populoidaan tietovarastoon tai miten ETL-kerros toimii (Theobald 2007; Vucevic & Zhang 2011, s. 99; Yaddow 2011, s. 4). Tavallisesti fokus on tietovaraston tietosisällössä ja siitä johdetuissa näkymissä (Theobald 2007; Yaddow 2011, s. 4).

Rainardi (2008, s. 486), Vucevic & Zhang (2011, s. 99) ja Yaddow (2011, s. 4) korostavat, että hyväksyntätestauksessa tulisi käyttää ehdottomasti suoraan tuotantodataa tai ainakin sitä läheisesti muistuttavaa testidataa, jota päivitetään hyväksyntätestauksen aikana käyttäjien toiveiden mukaisesti. Theobaldin (2007) ja Yaddowin (2011, s. 4) mukaan käyttäjät löytävät ongelmia vasta, kun he kohtaavat todellisuutta vastaavan järjestelmäympäristön. Testaajien tulisi pohtia, millaista dataa hyväksyntätestauksessa tarvitaan ja kuinka usein tämä data tulisi hyväksyntätestauksen aikana päivittää (Theobald 2007; Yaddow 2011, s. 4), mikä voi edellyttää yhteistyötä liiketoiminnan eri osa-alueiden edustajien kanssa. Oleellista on se, että käytettävä data kattaa hyväksymiskriteerien edellyttämät testitapaukset ja liiketoimintatapahtumat.

Hyväksyntätestauksessa testitapaukset tulisi suunnitella pohjautuen liiketoiminnan suorittamiin päivittäisiin rutiineihin tai aktiviteetteihin (Rainardi 2008, s. 486). Oleellista olisi ottaa huomioon myös näkymien sisältö suhteessa odotettuun sisältöön (Yaddow 2011, s. 4). Rainardin (2008, s. 486) mukaan testitapausten suunnittelussa tulisi ymmärtää, että hyväksyntätestauksessa keskeistä on, kuinka yksinkertaista tai helppoa eri roolissa toimivien käyttäjien on hyödyntää tietovarasto- ja raportointijärjestelmää suoriutuakseen tehtävistään. Esimerkiksi käytettävyys ei ole koskaan absoluuttista, vaan riippuu roolista ja käyttäjäprofiilista vastuiden ja tarpeiden vaihtelusta johtuen.

Hyväksyntätestauksen aikana loppukäyttäjien tulisi kirjata ylös kommentteja, mielipiteitä ja huomioita järjestelmän toiminnallisuudesta, käytettävyydestä ja ulkoasusta sekä koota nämä yhtenäiseksi loppuraportiksi (Rainardi 2008, s. 486; Vucevic & Zhang 2011, s. 99). Lopuksi, loppuraportti tulisi hyväksyttää liiketoiminnan edustajilla, jotka ymmärtävät loppuraportin sisällön (Vucevic & Zhang 2011, s. 99). Jos merkittäviä puutteita tai ongelmia ei ilmene, järjestelmä voidaan hyväksyä tuotantokäyttöön

(Brahmkshatriya 2007). Kuvassa 4.11 esitetään tiivistettynä hyväksyntätestauksen keskeiset hyvät käytännöt.



Kuva 4.11. Hyväksyntätestauksen hyvät käytännöt

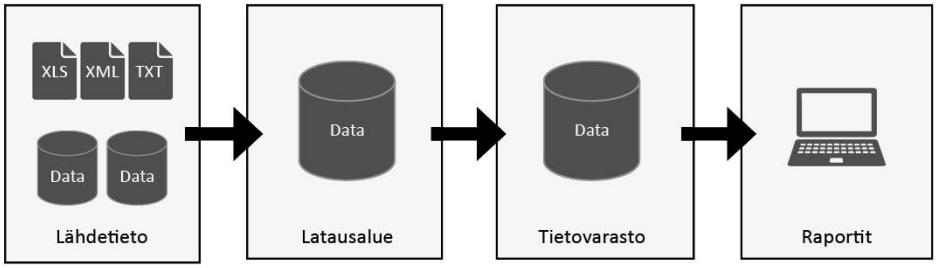
4.9 Suorituskykytestaus

Suorituskyvyn testauksella varmistetaan, että eri tyyppiset latausoperaatiot, samanaikaiset kyselyt tai hakuoperaatiot ja muu ajon aikainen prosessointi ei kaada järjestelmän toimintoja tai hidasta järjestelmän toimintaa siten, että järjestelmän suorituskyvylle asetetut vaatimukset alitetaan. Tavoitteena on osoittaa potentiaalisia heikkouksia ja huonosti optimoituja prosessointivaiheita eri osissa järjestelmää. (Rainardi 2008, s. 483; Mathen 2010; Vucevic & Zhang 2011, s. 97.) Kun volyymit tietovarasto- ja raportointijärjestelmässä lisääntyvät, järjestelmä altistuu suuremmalle kuormitukselle ja esimerkiksi ETL-prosessien ajaminen voi hidastua huomattavasti (Theobald 2007). Toisaalta kehitystyön aikana tietomäärät ovat suhteellisen pieniä (Rainardi 2008, s. 483). Järjestelmän on kuitenkin toimittava mutkattomasti tuotantoympäristössä, jossa yksittäiset faktataulut voivat sisältää miljoonia rivejä ja jossa voidaan tehdä lukuisia rinnakkaisia operaatioita. Näin ollen suorituskyvyn testaukseen on sisällytettävä myös stressi- ja skaalautuvuustestausta vähintään tuotantoa vastaavilla volyymeilla (Vucevic & Zhang 2011, s. 97). Näin varmistetaan, että järjestelmä pysyy tuotannossa suorituskykyisenä tietomäärien ja prosessoinnin (hakuoperaatiot, kyselyt) kasvusta huolimatta (Rainardi 2008, s. 483). Vucevicin & Zhangin (2011, s. 97) mukaan suorituskykyä tarkasteltaessa tulisi ottaa huomioon kaksi tekijää:

- **tiedon prosessoinnin ja latausten tehokkuus,**
- **hakuoperaatioiden tehokkuus.**

Suorituskykyä tulisi testata kolmessa eri pisteessä eri järjestelmäkerrosten välillä kuvan 4.12 mukaisesti. Rainardi (2008, s. 483) ja Bocarsly (2011) korostavat ETL- ja varastointikerroksen sekä varastointi- ja raportointikerrosten välisiä pisteitä. Golfarelli & Rizzi (2009) lisäisivät tähän vielä lähdejärjestelmien ja ETL-kerroksen välisen pisteen

eli sen, miten järjestelmä suoriutuu lähdetiedon poimimisesta ja lataamisesta ETL-kerroksen latausalueelle. Käyttäjän kannalta olennaisinta on vaatimusten mukaiset vasteajat raportointikerroksella (raportointityökaluissa tai esityskerroksella), sillä ne vaikuttavat suoraan käyttäjien arkeen (Golfarelli & Rizzi 2009). Tanuska et al. (2009) nostavat esiin myös käsittelyaikojen johdonmukaisuuden yhtenä tärkeänä tekijänä tietovarasto- ja raportointijärjestelmissä, mikä tarkoittaa käytännössä mahdollisten järjestelmän pullonkaulojen eliminointia.

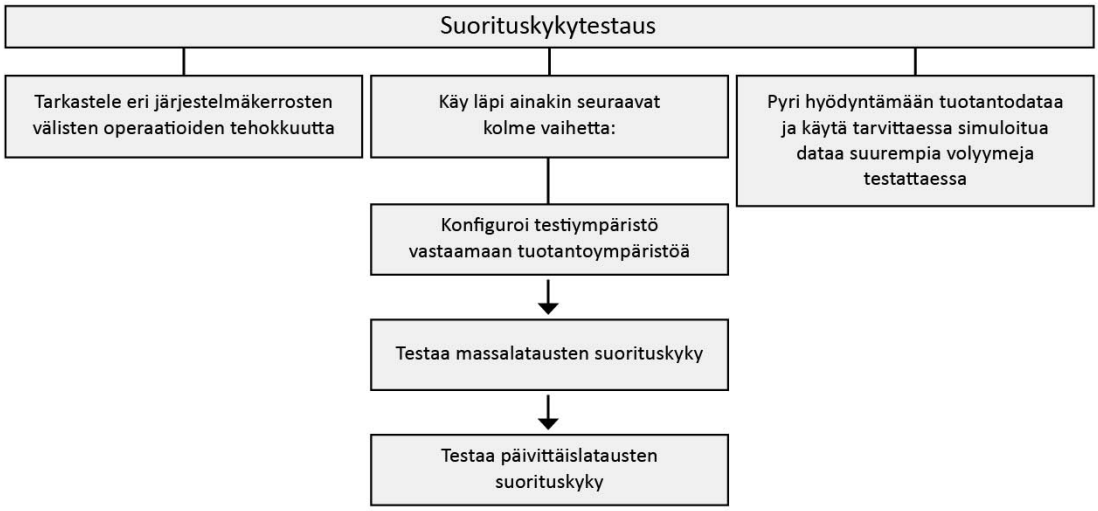


Kuva 4.12. Suorituskykytestaus järjestelmäkerrosten välillä (mukailtu lähteestä Bocarsly 2011; Golfarelli & Rizzi 2009)

Rainardin (2008, s. 483) mukaan ETL-kerroksen suorituskyvyn testaamiseksi tulisi suorittaa kolme vaihetta: testiympäristön pystyttäminen ja konfigurointi, massalatausten ja päivittäislatausten ajaminen. Ensimmäisessä vaiheessa tulisi konfiguroida testiympäristö tuotannon konfiguraatiota vastaavaksi ja suorittaa koko ETL-prosessi alusta loppuun saakka hyödyntäen suhteellisen pieniä datavolyymeja, jotta varmistutaan konfiguraation asianmukaisuudesta (Rainardi 2008, s. 483). Toisessa vaiheessa tulisi suorittaa massalataus tietovaraston populoimiseksi testiympäristössä ja mitata kyseisen massalatauksen suoritusarvoja eli useimmiten latausaikaa. Tässä kohtaa voidaan hyödyntää esimerkiksi ETL-työlogia ja -työkaluja (MS SQL Server Profiler) tai taulujen aikaleimoja suoritusarvoiltaan heikkojen kohtien tunnistamiseksi. Kolmannessa vaiheessa tulisi testata päivittäislataukset ja mitata niiden suoritusarvot samalla tavalla kuin massalatausten kohdalla. Jos jokin latauksista edellyttää lähes reaaliaikaisuutta, on mitattava myös aikaviivettä, joka kuluu lähdejärjestelmässä tapahtuvasta transaktiosta tiedon siirtymiseen tietovarastoon. (Rainardi 2008, s. 483-484.) Raportointikerroksella keskeinen mitattava suorituskykyarvo on vasteaika eli aika, joka kuluu kyselyn tulosten saamiseen siitä hetkestä, kun käyttäjä osoittaa haun raportointityökalussa.

Yleisesti ottaen suorituskyvyn testaus voidaan suorittaa joko suoraan tuotantodatalla tai simuloidulla datalla (engl. mock data), mutta volyymien tulisi vastata aina tuotantoa (Golfarelli & Rizzi 2009; Tanuska et al. 2009). Erilaiset rajoitteet testidatassa tulisi ottaa huomioon siten, että yksittäinen moduuli tai järjestelmä testataan näiden rajojen sisä- ja ulkopuolella (Tanuska et al. 2009). Toisaalta stressitesteissä tulisi käyttää lähtökohtai-

sesti simuloitua dataa, jotta saadaan testattua järjestelmän toimintaa tuotantodataa merkittävästi suuremmilla volyymeillä (Golfarelli & Rizzi 2009). Kuvassa 4.13 esitetään tiivistettynä suorituskykytestauksen keskeiset hyvät käytännöt.



Kuva 4.13. Suorituskykytestauksen hyvät käytännöt

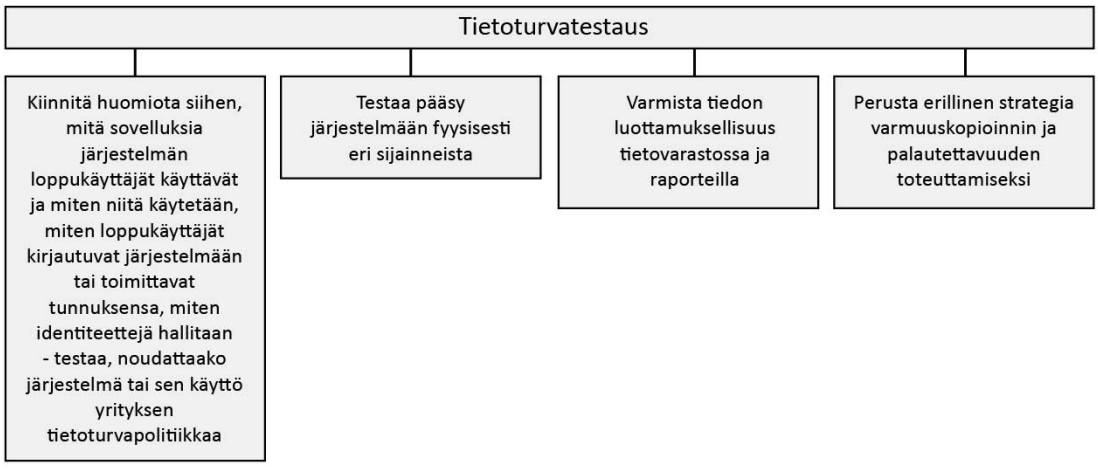
4.10 Tietoturvatestaus

Tietovarasto- ja raportointijärjestelmän tietoturvatestauksessa pyritään varmistumaan asianmukaisesta pääsystä tietoon tai sovelluksiin eri järjestelmäkerroksilla (Golfarelli & Rizzi 2009; Rainardi 2008, s. 485). Tietoturvatestauksessa on kiinnitettävä huomiota siihen, mitä sovelluksia järjestelmän loppukäyttäjät käyttävät ja miten niitä käytetään, miten loppukäyttäjät kirjautuvat järjestelmään tai toimittavat tunnuksensa ja miten identiteettejä hallitaan tai millaista politiikkaa pääsyoikeuksien hallinnassa noudatetaan (Rainardi 2008, s. 485). Raportointikerroksella tulisi käydä läpi käyttäjäroolit ja käyttäjäroolien oikeudet, minkä yhteydessä tulisi tarkistaa myös kertakirjautumispolitiikat. Kertakirjautuminen sujuvoittaa loppukäyttäjien työtä mutta voi vaarantaa tietoturvan (Golfarelli & Rizzi 2009). Golfarelli & Rizzi (2009) korostavat myös ETL-kerroksen tietoturvaa, jolloin testauksessa tulisi ottaa kantaa erityisesti tiedon eheyteen ETL-kerroksella ennen tietovaraston populointia ja toisaalta tietoverkkoinfrastruktuurin turvallisuuteen. Tiedon eheydellä tarkoitetaan sitä, että tieto pysyy koskemattomana ja muuttumattomana muiden kuin tarkoituksenmukaisten transformaatioiden toimesta. Vastaavasti tietoverkkoinfrastruktuurin turvallisuudella tarkoitetaan käytännössä sitä, onko ulkopuolisen henkilön tai tahon mahdollista tunkeutua järjestelmään ja millaista tietoturvapolitiikkaa asian suhteen noudatetaan (Golfarelli & Rizzi 2009; Rainardi 2008, s. 486).

Liiketoiminnan vaatimuksista riippuen, tietovarasto- ja raportointijärjestelmään voi olla pääsy joko organisaation sisäisestä tietoverkosta, organisaation ulkoisesta tietoverkosta

tai mahdollisesti näistä molemmista. Toisaalta pääsy järjestelmään voi olla rajoitettu tiettyyn osaan tietoverkkoa. Näin ollen tietoturvatestauksessa pääsy järjestelmään tietoverkon eri pisteistä tulisi testata perusteellisesti muun muassa järjestelmän sisältämän arkaluonteisen ja luottamuksellisen tiedon vuoksi (Rainardi 2008, s. 486). Rainardin (2008, s. 486) mukaan tämä kannattaa toteuttaa tietoturvapolitiikan ja liiketoiminnan vaatimusten mukaisesti siten, että tietoverkon eri osista otetaan yhteys järjestelmään. Testauksen yksinkertaistamiseksi tietoliikenne voidaan esimerkiksi reitittää tarkasteltavan tietoverkon toimialueen kautta. Tietoturvavaatimukset sanelevat sen, mitä loppukäyttäjät voivat tehdä järjestelmässä ja mihin järjestelmän osiin heillä on pääsy. Nämä vaatimukset on verifioitava siten, että eri käyttäjäroolien pääsyoikeudet testataan käytännössä. Esimerkiksi luottamukselliseksi osoitetun tiedon näkyvyys eri käyttäjäroolien välillä on ehdottomasti testattava. Tanuska et al. (2009) vastaavasti suosittelevat pääsyoikeuksien testausta yksinkertaisilla, esivalmistelluilla raporteilla, mikäli valmiita raportteja ei ole saatavilla. Näin voidaan varmistaa, että käyttäjät näkevät raporteilla vain ne tiedot, joihin heillä on oikeus.

Tietoturvatestaus käsittää muutakin kuin esimerkiksi pääsyoikeuksien testaamisen. Tanuska et al. (2009) korostavat erityisesti tiedon varmuuskopioinnin ja palautettavuuden testaamista. Tätä varten tulisi perustaa erillinen strategia, jossa määritellään miten varastointikerroksen tietojen varmuuskopiointi toteutetaan ja kuinka palautettavuuden tulisi toimia. Testauksessa olennaista on käyttää tuotantoa vastaavia tietovolyymeja, jotta kyetään näkemään tiedon palautukseen liittyvät vaikutukset todellisuutta vastaavassa tilanteessa. (Tanuska et al. 2009.) Kuvassa 4.14 esitetään tiivistettynä tietoturvatestauksen keskeiset hyvät käytännöt.

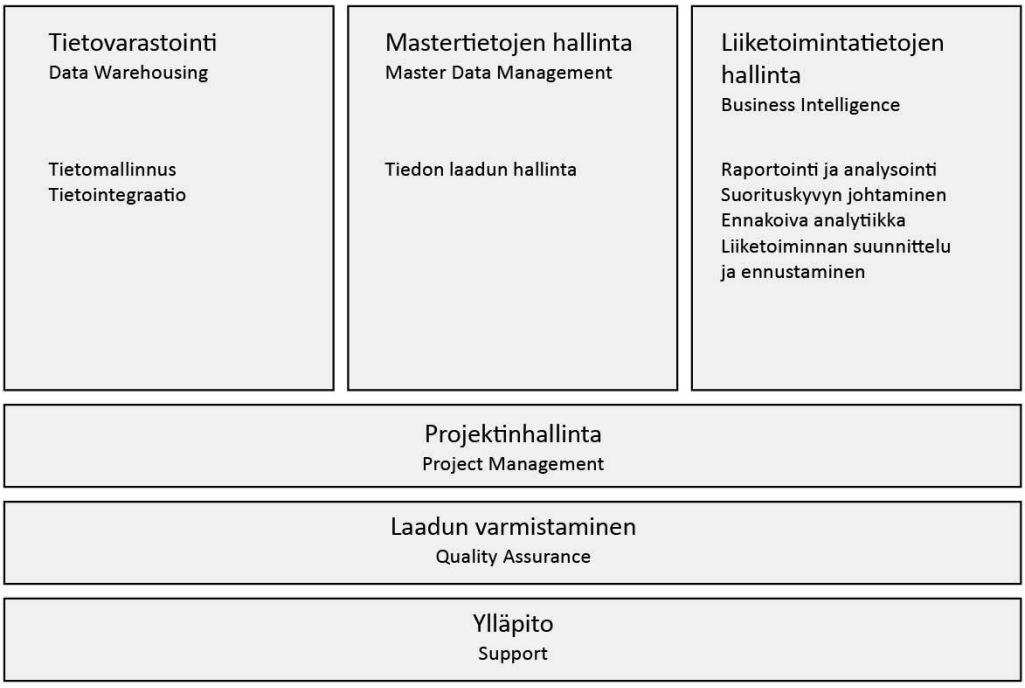


Kuva 4.14. Tietoturvatestauksen hyvät käytännöt

5 TUTKIMUKSEN KOHDE, MENETELMÄT JA TOTEUTUS

5.1 Kohdeyrityksen esittely

Rongo Oy on vuonna 2006 perustettu informaation hallintaan ja sen konsultointiin erikoistunut yritys. Tietovarastoinnin, mastertietojen ja liiketoimintatietojen hallinnan, suorituskyvyn johtamisen ja ennakoivan analytiikan palveluin se auttaa asiakkaitaan hallitsemaan, jalostamaan ja hyödyntämään informaatiota menestyksekkäämmin. Lisäarvoa se tuottaa asiakkailleen luomalla tilannekohtaisesti rakennettuja ratkaisumalleja erilaisiin informaation hallinnan tarpeisiin. Rongo on arvoiltaan suomalainen, itsenäinen kasvuyritys, joka toimii lähellä asiakasta. Vuonna 2012 liikevaihtoa kertyi noin 7 miljoonaa euroa ja kasvu on pysynyt edelleen vahvana. Rongon palveluksessa työskentelee tällä hetkellä noin 60 informaationhallinnan asiantuntijaa.



Kuva 5.1. Rongon palvelukokonaisuudet

Rongon palvelut (kuva 5.1) kattavat käytännössä kaikki informaation hallinnan osa-alueet, kuten tietovarastoinnin, mastertietojen hallinnan, liiketoimintatietojen hallinnan, projektinhallinnan, laadun varmistuksen ja ylläpidon. Rongo on teknologiariippumaton

yritys, jossa teknologiat palveluiden toteuttamiseksi valitaan aina asiakkaan tarpeiden ja etujen mukaisesti.

Tämän tutkimuksen kannalta olennaisimmat palvelualueet ovat tietovarastointi ja laadunvarmistus. Rongo pyrkii noudattamaan järjestelmästäprojekteissa hyvien käytäntöjen mukaisia suunnitteluperiaatteita ja vaalii korkeaa laatua, mikä edellyttää osaamista ja hyvien käytäntöjen tuntemusta myös testauksesta ja sen hallinnasta. Testauksen hallinta on Rongon kokonaisvaltainen palvelu, joka muodostuu testauksen suunnittelusta, koordinoinnista ja raportoinnista. Sen avulla voidaan antaa takuu järjestelmällisestä ja tehokkaasta testausprosessista, jossa huolehditaan resursoinnista ja raportoinnista. Tässä tutkimuksessa keskitytään sekä testauksen, hyvien käytäntöjen että suunnitteluperiaatteiden tunnistamiseen ja vakiinnuttamiseen osaksi Rongon laadunvarmistustoimintaa testauksen hallinnan eri osa-alueilla.

5.2 Teemahaastattelut tutkimuksen tukena

Tutkimusongelma voi olla hyvin moniulotteinen ja saattaa edellyttää syvällistä tarkastelua, jota on käytännössä mahdotonta tuottaa pelkkien strukturoitujen haastatteluiden tai kyselyiden avulla. Lisäksi vastausvaihtoehtojen luominen voi tuottaa ongelmia, koska haastattelijan ennakkokäsitys ja tietojen sekä kapea-alaisen näkökulman perusteella luodut kysymykset eivät välttämättä tuota oikeita tuloksia tai vastaavat vääriin kysymyksiin. Näin ollen, tässä tutkimuksessa hyödynnetään haastattelumenetelmänä teemahaastatteluja, joissa yhdistyvät sekä avoimen haastattelun että strukturoidun haastattelun ominaispiirteet (Koskinen et al. 2005, s. 104-105). Toisin sanoen, teemahaastattelu on puolistrukturoitu haastattelumenetelmä, jossa ennalta määritetyt haastattelun teemat eli näkökulmat tutkimusongelman ratkaisemiseksi ohjaavat haastattelua oikeaan suuntaan mutta korostavat haastateltavan kokemusta ja asiantuntemusta aiheesta (Koskinen et al. 2005, s. 104-105; Vuorela 2005, s. 39-40). Näin ollen, teemahaastattelun keinoin saadut tulokset ovat käytännössä aina sidoksissa tutkimusongelmaan. Taulukossa 5.1 tehdään vertailu strukturoidun haastattelun eli lomakehaastattelun, teemahaastattelun ja avoimen haastattelun ominaispiirteiden välillä.

Taulukko 5.1. Vuorelan (2005, s. 40) esittämä haastattelumenetelmien vertailu (Hirsjärvi & Hurme 1995)

	Lomakehaastattelu	Teemahaastattelu	Avoin haastattelu
Kysymysten muotoilu	Kiinteä	Suosituskysymyksiä	Vapaa
Kysymysalue	Tiukasti määritelty	Pääpiirteittäin määritelty	Vapaa
Osallistujamäärä	Suuri	Melko pieni	Pieni
Kustannus / yksikkö	Pieni	Suuri	Suuri
Analyysin työmäärä	Melko pieni	Suuri	Suuri
Tutkijan tiedot	Voivat olla vähäiset	Tulisi olla laajat	Tulisi olla laajat
Saatu tieto	Pintapuolinen	Syvällinen	Syvällinen

Haastattelurunko rakennetaan valittujen teemojen pohjalta, ja sitä sovelletaan kussakin haastattelutilanteessa. Toisaalta, haastattelun ei tarvitse edetä haastatelurungon osoittamassa järjestyksessä, vaan teemoja voidaan käsitellä haastattelutilanteelle ominaisessa luontaisessa järjestyksessä. (Hirsjärvi & Hurme 2008; s. 47-48; Koskinen et al. 2005, s. 104-105.) Tällainen lähestymistapa antaa haastateltavalle mahdollisuuden pohtia, jäsennellä ja analysoida vastauksia riittävästi. Vuorelan (2005, s. 40) mukaan teemahaastattelu voikin olla luonteeltaan hyvin vapaamuotoista keskustelua, jossa on kuitenkin olennaista huolehtia siitä, että samat aiheet käsitellään kaikkien haastateltavien kanssa (Preece et al. 2002). Kuten taulukosta 5.1 nähdään, teemahaastattelu soveltuu hyvin kyseessä olevan tutkimusongelman ratkaisemiseksi, sillä pienestä osallistujamäärästä huolimatta saadut tiedot ovat melko syvällisiä ja antavat näin paremman tuloksen kuin strukturoitu haastattelu. Toisaalta taas kysymysten muotoilu ja kysymysalueet rajoittavat tai ohjaavat haastattelua avointa haastattelua tiukemmin oikeaan suuntaan.

Kysymykset laadittiin yhteistyössä tutkimuksen ohjaajien kanssa ja ne pyrittiin muodostamaan mahdollisimman avoimiksi siten, että ne toisivat haastateltavan omat havainnot ja kokemukset parhaiten esiin ja tukisivat analyyttisia vastauksia. Toisaalta, kysymysten asettelussa pyrittiin välttämään johdattelua tai haastattelijan omia mielipiteitä. Käytännössä haastattelun teemat jakautuivat kolmeen osaan: testausprosessiin, testauksen haasteisiin ja testauksen osa-alueisiin. Näiden teemojen avulla pyrittiin avaamaan haastateltavien kokemuksia onnistuneista ja epäonnistuneista projekteista ja selvittämään, miten tietovarasto- ja raportointijärjestelmiä tulisi testata.

5.3 Haastatteluiden suorittaminen ja tutkimusaineiston analysointi

Teemahaastattelut toteutettiin kohdeyrityksessä maaliskuussa 2013 ja ennalta valituissa kohdeyrityksen asiakasorganisaatioissa huhtikuussa 2013. Kohdeyrityksen sisäisesti haastateltaviksi valitut henkilöt tunnistettiin tutkimuksen ohjaajien avustuksella ja perusteena käytettiin asiantuntemusta sekä kokemusta tietovarasto- ja raportointijärjestelmien projektinhallinasta, kehitystyöstä sekä testauksesta. Haastateltujen henkilöiden

valintaan vaikuttivat olennaisesti tämän tutkimuksen ohjaajien suositukset ja haastateltavien oma mielenkiinto sekä osaaminen tutkimuksen osa-alueelta.

Kohdeyrityksestä haastateltiin kuutta asiantuntijaa ja kohdeyrityksen kahdesta asiakasorganisaatiosta yhtä henkilöä kustakin. Kaikilla kohdeyrityksestä haastatelluilla asiantuntijoilla on pitkä kokemus tietovarastoinnin ja raportoinnin osa-alueelta. Suurin osa heistä kuuluu kohdeyrityksen johtoryhmään ja toimii johtavina konsultteina. Kukin asiantuntija tarjoaa hieman erilaisen näkemyksen tutkimuksen aiheeseen, sillä taustat ja lähtökohdat eroavat luonnollisesti hieman toisistaan. Karkeasti ottaen osa haastatelluista on keskittynyt työssään enemmän tietovarastointiin, osa raportointiin. Tällainen näkemysten monimuotoisuus on hyödyllinen tekijä tutkimuksen kannalta, kun asioita pystytään tarkastelelemaan eri näkökannoilta eri puolilta järjestelmää. Lisäksi tutkimuksessa haastateltiin kahta asiakkaan (tai integraattorin) näkökulmaa edustavaa henkilöä, joista toinen oli tietovarasto- ja raportointijärjestelmien testaukseen perehtynyt liiketoiminnan testaaja ja toinen testauksesta vastaavan yksikön päällikkö. Molemmat työskentelivät Suomen valtion eri organisaatioissa. Näiden henkilöiden eriävät taustat antoivat hyvät lähtökohdat tarkastella tutkimusongelmaa sekä hallinnollisesta että käytännönläheisestä näkökulmasta. Haastatellut henkilöt kuvataan taulukossa 5.2.

Taulukko 5.2. Tutkimuksessa haastallut henkilöt

Näkökulma	Henkilö	Asema	Kokemus
Toimittaja (tutkimuksen kohdeyritys)	Henkilö A	Johtava konsultti, johtoryhmän jäsen	> 8 v
	Henkilö B	Johtava konsultti, johtoryhmän jäsen	> 8 v
	Henkilö C	Johtava konsultti, johtoryhmän jäsen	> 8 v
	Henkilö D	Johtava konsultti	> 8 v
	Henkilö E	Johtava konsultti, johtoryhmän jäsen	> 8 v
	Henkilö F	Johtoryhmän jäsen	> 8 v
Asiakas / integraattori	Henkilö G	Testauksesta vastaavan yksikön päällikkö	> 8 v
	Henkilö H	Liiketoiminnan testaaja	> 8 v

Haastatteluiden toteuttaminen oli monivaiheinen prosessi. Vuorela (2005, s. 45) toteaa, että ennen varsinaisten haastatteluiden aloittamista tulisi tehdä esihaastatteluita useassa vaiheessa. Tämän tutkimuksen osalta nähtiin yhdessä tutkimuksen ohjaajien kanssa tarpeelliseksi toteuttaa yksi esihaastattelu, jossa keskityttiin erityisesti kysymysten ymmärrettävyyteen, rajauksiin ja asetteluun. Esihaastattelun tulosten pohjalta tehtiin lopulta pieniä muutoksia muun muassa kysymysten asetteluun. Esihaastattelun tarkoituksena oli ensisijaisesti testata haastattelurungon toimivuutta ja selvittää haastattelun keston suuruusluokka. Toisaalta, esihaastattelun avulla haastattelijaa harjaannutettiin toimimaan haastattelutilanteessa.

Haastattelutilanteissa läsnä olivat vain haastattelija ja yksi haastateltava. Haastattelijan tavoitteena oli luoda vuorovaikutteinen tilanne, jossa haastateltava sai ilmaista ajatuksiaan ja kokemuksiaan vapaasti teemojen rajoissa. Myöhempää tutkimusaineiston sisällönanalyysia varten, haastattelut, yhtä haastateltavaa hänen omasta pyynnöstään lukuunottamatta, äänitettiin älypuhelimella. Tutkimusaineisto litteroitiin äänitteeltä välittömästi haastattelun jälkeen haastattelupöytäkirjaksi. Litterointiin päädyttiin siksi, että onnistunut tulosten analysointi edellyttää huolellista perehtymistä tutkittavaan aineistoon (Vuorela 2005, s. 48).

Myös tutkimusaineiston analysoiminen oli monivaiheinen prosessi. Hirsjärvi & Hurme (2001) ovat esittäneet tutkimusaineiston kvalitatiivisen analyysin kolmivaiheisena prosessina (Vuorela 2005, s. 48), jota noudatettiin myös tässä tutkimuksessa. Tutkimusaineiston kuvailu on ensimmäinen vaihe, joka toimi analyysin perustana. Vuorelan (2005, s. 48) mukaan siinä etsitään tyypillisesti vastauksia kysymyksiin mitä, missä, milloin, kuka ja kuinka usein. Tämän tutkimuksen kohdalla kuvailu tarkoitti tutkimusaineiston karkeaa tarkastelua ja arviointia. Pyrkimyksenä oli kartoittaa, millainen tutkimusaineisto kokonaisuudessaan oli käytettävissä. Kuten kolmivaiheiseeen prosessiin kuuluu, kuvailuvaiheen jälkeen siirrytään luokitteluvaiheeseen, jossa luodaan viitekehys tutkimusaineiston tulkitsemiseksi. Hirsjärven & Hurmeen (2001) mukaan luokitteluvaiheen keskeinen ajatus on jäsentää luokiteltu tutkimusaineisto siten, että eri osia voidaan myöhemmin vertailla toisiinsa nähden ja tulkita yksinkertaisemmin. Luokitteluvaiheessa tutkimusaineistoa jäsenneltiin ja arvioitiin karkealla tasolla siten, että aineistosta tunnistettiin valittujen teemojen mukaisia selkeitä, vertailtavissa olevia kokonaisuuksia, joiden avulla pystyttiin tunnistamaan tutkimusongelman kannalta keskeiset asiat yleisellä tasolla. Prosessin periaatteiden mukaisesti luokittelun jälkeen tutkimusaineisto tulisi vielä yhdistellä. Vuorelan (2005, s. 48) mukaan yhdistelyvaiheessa pyritään löytämään luokittelujen välillä esiintyviä samankaltaisuuksia, yhtymäkohtia ja säännönmukaisuuksia. Yhdistelyvaiheessa teemojen mukaan valmiiksi jäsenneltyjä kokonaisuuksia tarkasteltiin yksi kerrallaan. Kustakin kokonaisuudesta pyrittiin tunnistamaan edellä mainittuja piirteitä sekä mahdollisia syy-seuraussuhteita. Tämän jälkeen jäsennelty ja yhdistelty tutkimusaineisto purettiin kirjallisesti analyyttiseen muotoon ja lopulliset tulokset muodostettiin.

Tutkimusongelman laaja-alaisuus ja moniulotteisuus edellyttää tarkastelua, jossa pyritään tunnistamaan tutkimuksen kannalta oleellisiä, isoja linjoja, joiden osalta valtaosa haastateltavista jakaa saman suuntaisen näkemyksen. Luvussa 6 esitetyt haastattelutulokset edustavat hyvin sekä haastateltavakohtaista että tilastollista keskimääräistä kausaalisuutta, koska haastateltavien näkemykset tukevat toisiaan. Sellaisten kysymysten kohdalla, joissa näkemys jonkun haastateltavan osalta poikkesi selvästi muiden haastateltavien näkemyksistä, on käsitelty tutkimuksessa erikseen. Tämän tutkimuksen kohdalla, otoksen koko huomioiden, haastatteluiden yksilökohtaisempi tarkastelu voisi vaarantaa vastaajien anonyymiteetin, eikä tällainen lähestymistapa antaisi juurikaan lisäar-

voa kokonaisuutta ajatellen. Näin ollen tutkimuksen tulosten käsittely on toteutettu anonyymiyyden periaatteita noudattaen. Haastattelutuloksissa viitataan tyypillisesti joko toimittajan tai asiakkaan näkökulmaan. Kun jotain asiaa on haluttu erikseen korostaa, on käytetty suoraan yksilöiviä viittauksia. Tällainen lähestymistapa on ollut perusteltua, sillä toimittajan näkökulma muodostaa melko homogeenisen ryhmän kokemuksen ja aseman kannalta tarkasteltuna. Toisaalta, asiakkaan näkökulmasta näin yksinkertaista ryhmittelyä on hankalampi toteuttaa, ja siksi haastattelutuloksissa on usein mainittu erikseen, mikä on esimerkiksi testaajan ja mikä päällikön näkemys.

6 HAASTATTELUTULOKSET

6.1 Erityispiirteet testauksen näkökulmasta

Eräs haastatteluiden teema käsitteli sitä, miten tietovarasto- ja raportointijärjestelmät eroavat muista tietojärjestelmistä testauksen näkökulmasta. Kuten teoreettiset tulokset osoittavat, kyseisten järjestelmien kohdalla on tunnistettavissa erityispiirteitä, jotka erottavat sen perinteisestä ohjelmistotestauksesta. Haastateltujen henkilöiden kokemus kyseisten järjestelmien kehityksestä antaa hyvän lähtökohdan tunnistaa ja analysoida erityispiirteitä tarkemmin.

Niin toimittajan kuin asiakkaankin näkökulmista tarkasteltuna haastatteluaineiston keskeisin havainto oli ehdottomasti tietosisällön merkitys. Kaikkien haastateltujen mukaan testauksessa tulisi keskittyä varmistamaan se, että eri järjestelmäkerroksilla sijaitseva tieto on luotettavaa, mikä on koko järjestelmän perusta. Useimpien haastateltavien kohdalla toistui näkemys siitä, että lopulta kaikki testit pyrkivät tavalla tai toisella ottamaan kantaa tiedon oikeellisuuteen. Kuten henkilö G totesikin haastattelussa, tietovarasto- ja raportointijärjestelmien testaus on ”*käytännössä kahden pisteen välistä tiedon vertailua*”, johon ei liity samalla tavoin toiminnallisuuksien tai käyttöliittymäpohjaista testausta kuten tyypillisesti perinteisten ohjelmistotuotteiden kohdalla. Tiedon (datan tai informaation) luotettavuus on merkittävä tekijä, koska sen pohjalta tehdään liiketoiminnan kannalta kriittisiä päätöksiä. Henkilön D tiivisti asian seuraavasti:

> ”*Merkittävin ero on se, ettei testata mitään koodattua toiminnallisuutta, vaan loppupeleissä kysymys on siitä, onko data oikein vai ei.*”

Toinen yleinen hyvin vahvasti esiin noussut erityispiirre oli integraatioiden suuri määrä, tiedon jalostusketjun monimutkaisuus ja lähdejärjestelmien tiedon laatu. Erityisesti toimittajan näkökulmasta viestittyi vahvasti näiden tekijöiden mukanaan tuomat ongelmat testauksen kannalta. Integraatioiden määrä lisää tyypillisesti lähdejärjestelmien tiedon laadun ja epäjohdonmukaisuuksien aiheuttamia ongelmia testauksessa. Toisaalta, raporteille tuotavien tietojen jalostusketju voi olla monimutkainen, jolloin yhden arvon muodostamiseksi voidaan joutua suorittamaan useita eri operaatioita. Tämä heikentää toimittajan näkökulmasta tiedon jäljitettävyyttä ja voi tehdä testauksesta sekä virheiden tai puutteiden korjauksesta työlästä ja hankalaa, kun joudutaan selvittämään, mistä tai miten jokin arvo on saatu raportille.

Eräs toimittajan näkökulmasta esiin noussut erityispiirre oli testiympäristön ominaisuudet. Siitä, miten testiympäristöä tulisi hyödyntää tietovarasto- ja raportointijärjestelmien tapauksessa, haastateltujen henkilöiden näkemykset vaihtelivat. Osa korosti erillisen testiympäristön merkitystä ja osa oli yhteisen testi- ja tuotantoympäristön kannalla. Haastatteluista kävi kuitenkin selvästi ilmi, että perinteisten tietojärjestelmien tapauksessa käytetään useammin selkeästi erillistä testiympäristöä, ja tietovarasto- ja raportointijärjestelmien kohdalla yhdistettyjen testi- ja tuotantoympäristöjen hyödyntäminen on täysin mahdollista. Lisäksi useimmat haastatellut henkilöt olivat sitä mieltä, että tietovarasto- ja raportointijärjestelmien tapauksessa testi- ja tuotantoympäristöjen välinen vertailukelpoisuus oli huomattavasti tärkeämpää kuin perinteisten tietojärjestelmien kohdalla.

Yleisistä näkemyksistä poiketen eräs mielenkiintoinen näkemys oli henkilön A esiin tuoma eräajopohjaisuus. Tämä tarkoittaa käytännössä sitä, että käyttäjät eivät syötä järjestelmään yksittäisiä tapahtumia, vaan data syötetään järjestelmän toimesta eräajoina. Näin ollen käyttäjien tekemiin operaatioihin tai toiminnallisuuksiin ei juurikaan tarvitse kiinnittää huomiota testauksen aikana.

6.2 Testauksen keskeiset haasteet

Tutkimuksessa pyrittiin selvittämään haastateltavien kokemuksia keskeisimmiksi koetuista haasteista, joihin testauksessa usein törmätään. Kaikilla haastateltavilla ei ollut laajaa omakohtaista, käytännönläheistä kokemusta testauksesta, mutta jokainen oli työskennellyt riittävän pitkään kehitystyössä sekä projektinjohdossa, joissa molemmissa testaus on olennainen osa työnkuvaa. Haastatteluaineiston perusteella keskeisimmiksi koettuja haasteita olivat oikean vertailuaineiston määrittäminen ja tulosten oikeellisuuden arvioiminen, puutteet osaamisessa ja resursoinnissa, ongelmat tiedon jäljitettävyydessä ja järjestelmän läpinäkyvyydessä, asenteet sekä määritysten puutteellisuus.

Toimittajan näkökulmasta erityisesti vertailuaineiston määrittäminen ja tulosten oikeellisuuden arvioiminen aiheuttaa lähes aina jonkin asteisia ongelmia. Tämä korostui vahvasti muun muassa henkilöiden C ja D haastatteluissa. Toisaalta, henkilö H ei pitänyt asiaa kovin ongelmallisena ja henkilö G vastaavasti suhtautui asiaan neutraalisti. Toimittajan näkökulmasta yleinen näkemys oli se, että tulosten arviointi on hankalaa, jos vertailuaineiston asianmukaisuudesta ei päästä yksimielisyyteen tai jos asianmukaista vertailuaineistoa ei osata hakea tai muodostaa lähdejärjestelmistä tai olemassa olevasta raportointiratkaisusta. Vertailuaineiston määrittämisessä ei välttämättä havaita kaikkia käsittelysääntöjä, joita olemassa olevan järjestelmän tietoihin liittyy. Myös näkemykset voivat vaihdella. Toisaalta, myös tarkkuudella on merkitystä eli mikä on riittävä tarkkuus vertailutietojen esittämiseksi. Näitä asioita on hankala arvioida käytännössä varsinkin tapauksissa, joissa testaajat eivät tunne liiketoimintaa ja testausympäristöä riittä-

vän hyvin. Henkilön H osaaminen oli todennäköisesti eräs iso tekijä, miksi hän ei kokenut asiaa kriittiseksi.

Yleisesti asiakkaan näkökulmasta yhdeksi keskeiseksi haasteeksi nousi resurssien niukkuus. Tämä oli myös toimittajan näkökulmasta hyvin yleinen huolen aihe. Henkilön H mukaan testaajia tulisi olla aina vähintään kaksi, jotta testauksessa ei tapahtuisi niin sanottua urautumista. Urautumisella tarkoitetaan sitä, että testaaja tulee sokeaksi oman tekemisensä suhteen, eikä osaa nähdä kaikkia mahdollisia variaatioita tai mahdollisuuksia. Lisäksi henkilön H mukaan toisesta testaajasta olisi ehdottomasti hyötyä tämän tuodessa asioihin uusia näkemyksiä. Useammat henkilöt toimittajan näkökulmasta arvioivat, että toimittajan kohdalla ongelmana on usein puutteellinen asiakkaan liiketoiminnan tuntemus, eikä konsulttien syvällinen perehtyminen asiakkaan liiketoimintaan ole mahdollista etenkään yksittäisten projektien laajuudessa. Lisäksi motivaatiota voi olla hankala löytää, kun aikaa on rajoitetusti, eikä sitä ole useinkaan resursoitu riittävästi perehtymiseen.

Toimittajan näkökulmasta asiakkaan ongelmat kohdistuvat usein tekniseen asiantuntemukseen sekä resursointiin. Tämä tarkoittaa sitä, että asiakkaalla ei ole välttämättä kykyä tai mahdollisuutta sitouttaa riittävää osaamista testaukseen, mikä johtaa usein testauksen merkityksen vähättelyyn. Toisaalta osaavat avainhenkilöt saattavat olla jo valmiiksi niin työllistettyjä, ettei testaukselle yksinkertaisesti riitä aikaa. Testauksen kannalta tärkeiden avainhenkilöiden löytyminen asiakkaan organisaatiosta ei ole itsestäänselvyys. Tätä osaamiseen liittyvää haastetta tukevat myös henkilöiden G ja H näkemykset. Esimerkiksi henkilön H haastattelusta kävi ilmi, että tietovaraston ymmärtäminen on välillä haasteellista, mikä hankaloittaa tai hidastaa testausta. Tietovaraston rakentaminen onkin usein asiakkaan tietohallinnon ja toimittajan välinen projekti, joten testaajat liiketoiminnan edustajina jäävät helposti vaille riittävää informaatiota tietovarastoratkaisusta. Vaikka koulutus järjestettäisiin jälkikäteen, liiketoiminnan testaaja ei välttämättä kykene ymmärtämään, miksi jokin tieto on viety tietovarastoon tietyllä tavalla. Tämä voi tehdä testauksesta vaikeaa, ja yhä useampien kysymysten herätessä, tulosten arviointiin liittyvä epävarmuus lisääntyy. Löydösten arvioiminen on hankalaa, kun syytä ei osata itsenäisesti selvittää.

Useimmat kohdeyrityksen haastatelluista henkilöistä näkivät ongelmalliseksi sen, että tietovarasto- ja raportointijärjestelmäprojektien yhteydessä vain harvoin käytetään varsinaisia testaajia, ja vastuu testauksesta hajautetaan kulloinkin vapaana oleville henkilöille, joilla ei aina ole riittäviä edellytyksiä tai motivaatiota vastata testauksesta. Haastatteluaineistosta oli pääteltävissä, ettei substanssiosaaminen välttämättä riitä, jos testauksesta ei ole aiempaa kokemusta. Toinen usein haastatteluissa toistunut seikka oli se, että liiketoiminnan näkökulmasta testauksen merkitys nähdään usein melko vähäisenä, eikä hyötyjä ei tunnisteta. Kohdeyrityksestä haastateltujen henkilöiden vallitsevan näkemyksen mukaan testauksen merkityksen vähättely johtaa usein siihen, että testausta

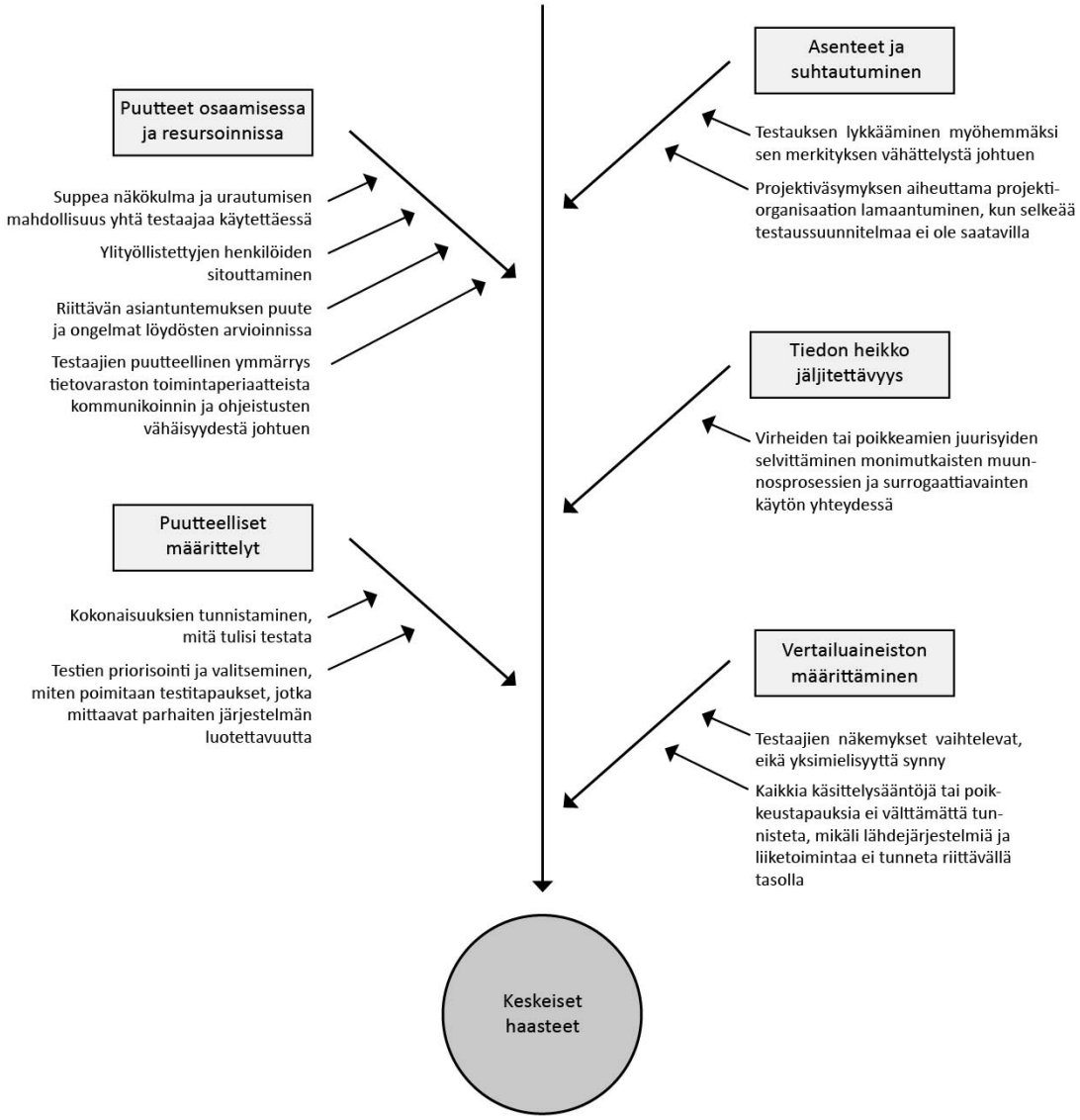
siirretään jatkuvasti myöhäisempään ajankohtaan, kunnes lopulta huomataan, että järjestelmä on siirretty tuotantoon ilman, että sitä on testattu kunnolla. Järjestelmän loppukäyttäjät saattavat törmätä tuotannossa ongelmiin, eikä järjestelmän käyttöönotto suju tällöin oletetulla tavalla. Jos järjestelmän ensivaikutelma on kehno, muutosvastarinta voi muodostua merkittäväksi ongelmaksi, ja sen vaikutukset voivat ulottua pitkälle tulevaisuuteen. Toisaalta testauksen siirtäminen myöhempään ajankohtaan aiheuttaa sen, että projektiväsymys lamaannuttaa testausaktiviteetit. Mikäli projektiorganisaatiolla ei ole tällöin käytettävissä selkeästi jäsenneltyä testaussuunnitelmaa, testaukseen tarttuminen voi olla äärimmäisen vaikeaa. Tätä asiaa korostettiin toimittajan näkökulmasta usean henkilön osalta. Vastaavasti, jos jo projektin alkuvaiheessa asiat on mietitty ennalta, aikataulupaine usein helpottaa ja testauksen läpivienti on tehokkaampaa.

Useimpien kohdeyrityksestä haastateltujen henkilöiden ja henkilön G antamien tietojen perusteella organisaatioiden suhtautuminen siihen, että tietovarasto- ja raportointijärjestelmiä tulisi testata, ei ole aina itsestäänselvyys. Aina ei välttämättä ymmärretä, että kuten ohjelmistokehityksessä, myös tietovarasto- ja raportointijärjestelmien kohdalla syntyy inhimillisiä virheitä.

Henkilö B nosti esiin surrogaattiavainten merkityksen testauksen kannalta. Haastatellun mukaan surrogaattiavainten käsitteleminen aiheuttaa haasteita, sillä tietojen jäljitettävyys ja läpinäkyvyys tietovarasto- ja raportointijärjestelmässä heikkenee. Raportilla havaitun virheen syyn selvittäminen ja korjaaminen voi olla työlästä, eikä ongelman selittäminen asiakkaalle ole aina helppoa. Jos tietovirrassa on paljon monimutkaisia käsittelyitä tai transformaatioita, voi olla vaikea selvittää, mistä esimerkiksi jokin poikkeava arvo tietovarastossa tai raportilla johtuu.

Useimmat kohdeyrityksestä haastatellut henkilöt sekä henkilöt G ja H kiinnittivät huomiota määrittelyihin ja niiden puutteellisuuteen. Haastatteluaineiston perusteella määrittelyt ovat hyvin usein joltain osin puutteellisia. Tämä on testauksen kannalta hankalaa siksi, että tällöin ei välttämättä pystytä tunnistamaan sitä, mitä oikeastaan tulisi testata. Testitapausten priorisoiminen on välttämätöntä, sillä testaukselle on asetettava rajat. Muutoin voidaan ajautua testaamaan asioita, jotka mittaavat vääriä tai epäolennaisia asioita, tai korjaamaan sellaisia yksityiskohtia, joilla ei ole merkitystä käytön tai kokonaisuuden kannalta. Kaikkea ei voida koskaan testata, joten on tärkeää löytää ne olennaiset testitapaukset, jotka todellisuudessa tuottavat lisäarvoa projektille. Henkilön G mukaan ongelmia on erityisesti ollut tiedon ja käsittelyiden määrittelyssä eli siinä, mistä tieto löytyy ja millä tavalla tietoa tulisi käsitellä.

Kuvaan 6.1 on koostettu haastatteluista esiin nousseet tietovarasto- ja raportointijärjestelmien testauksen keskeiset haasteet.



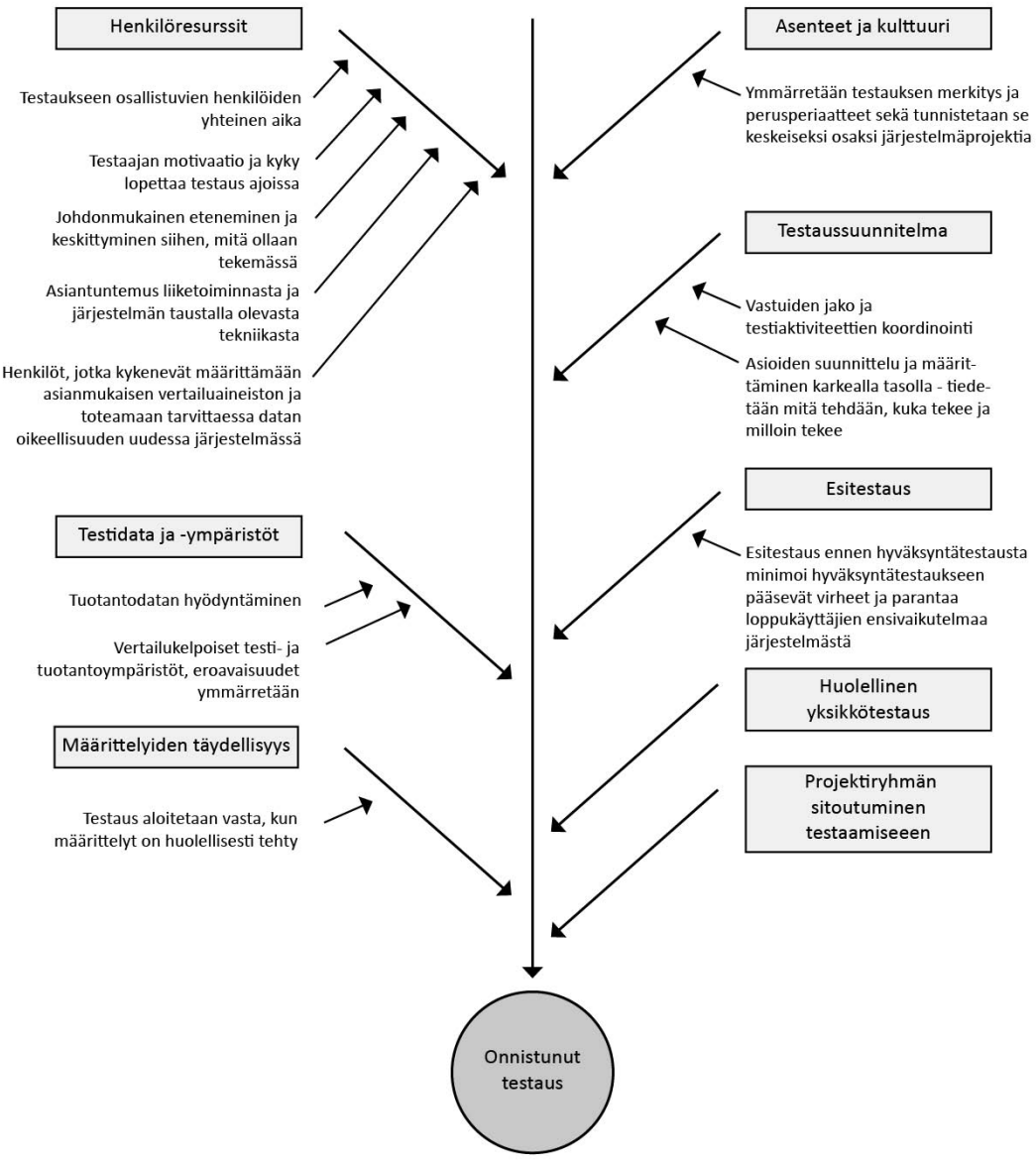
Kuva 6.1. Keskeiset haasteet testauksessa

6.3 Testauksen menestystekijät

Haastateltavia pyydettiin arvioimaan testauksen menestystekijöitä ja toisaalta syitä siihen, miksi testauksessa oli aiemmin epäonnistuttu. Tämä edellytti haastateltavilta yhtenäistä tulkintaa siitä, mikä koettiin testauksen kannalta onnistuneeksi tai epäonnistuneeksi projektiksi. Haastattelussa onnistunutta testausta kuvattiin prosessina, joka on ollut erityisen sujuva, saanut kiitosta tai tunnustusta, ollut merkittävässä roolissa projektin menestyksen kannalta tai esimerkiksi parantanut olemassa olevien lähdejärjestelmien tai olemassa olevan raportointiratkaisun laatua. Toisaalta onnistuneen testauksen seurauksena tuotannossa harvemmin esiintyy vakavia virheitä tai puutteita.

Testauksen menestystekijät kulminoituivat toimittajan näkökulmasta asenteisiin, testaukseen osallistuviin henkilöihin, testaussuunnitelmaan ja testauksen hallintaan sekä tes-

tiympäristöön ja -dataan. Yksi haastatelluista toi esiin myös liiketoiminnan esitestauksen, mikä on mielenkiintoinen näkökulma. Toisaalta, toisen haastatellun mukaan tekninen esitestaus oli tärkeää ennen varsinaista integraatio- tai järjestelmätestausta. Koska esitestaus mainittiin kahteen otteeseen ja perustelut olivat hyvin laajoja ja kokemusperäisiä, päätettiin tässä tutkimuksessa esitestaus liittää osaksi testauksen menestystekijöitä. Henkilön G haastattelussa korostuivat erityisesti huolellinen yksikkötestaus, projektiryhmän sitoutuminen ja määrittelyiden täydellisyys, mutta myös viitteitä muun muassa testaussuunnitelmaan, henkilöresursseihin ja testidataan oli löydettävissä. Kuvassa 6.2 esitetään tiivistettynä tietovarasto- ja raportointijärjestelmien testauksen menestystekijät, jotka nousivat esiin haastatteluaineistosta.



Kuva 6.2. Testauksen menestystekijät

Asenteet ja kulttuuri

Toimittajan näkökulmasta eri osapuolten asenteet testausta kohtaan, ainakin tietovarasto- ja raportointijärjestelmien osalta, vaikuttavat siihen, miten hyvin testauksessa onnistutaan. Yleisesti tulisi ymmärtää, että tietovarasto- ja raportointijärjestelmien tapauksessa data päätyy raportille ja tarjoaa informaatiota järjestelmän loppukäyttäjälle, minkä vuoksi myös testaus olisi tunnistettava keskeiseksi osaksi järjestelmäprojektia. Toimittajan kohdalla tämä tarkoittaa muun muassa proaktiivista otetta asiakkaan tukemiseksi testaukseen liittyvissä asioissa. Negatiivinen ajattelumalli testausta kohtaan syö motivaation ja lopulta johtaa testauksen epäonnistumiseen. Lisäksi huonot kokemukset ruokkivat entisestään tällaista ajattelumallia. Eräs kohdeyrityksen haastatelluista tiivisti tämän seuraavasti:

”Tulisi ajatella, että testaus on automaattisesti osa järjestelmäprojektia.”

Myös useat muut haastatellut tukivat tätä seikkaa. Kohdeyrityksestä haastateltujen mukaan testausta ei tulisi lähestyä liian IT-painotteisesti. Testauksen onnistuminen kuitenkin riippuu siitä, millä tasolla raportit vastaavat asiakkaan liiketoiminnan asettamiin kysymyksiin ja kuinka hyvin järjestelmä täyttää asiakkaan liiketoimintavaatimukset. Tämän johdosta liiketoiminnan osallistaminen testaukseen on erittäin tärkeää, eikä testausta sopisi siksi lähestyä liikaa teknisistä näkökulmista.

Henkilöresurssit

Toimittajan näkökulmasta tietovarasto- ja raportointijärjestelmien testauksessa liiketoiminnan tunteminen on kokonaisuutta ajatellen tärkempää kuin tekninen asiantuntemus, mutta molempia tarvitaan. Olennaisinta on, että testaukseen osallistuu sellainen henkilö, joka kykenee määrittämään asianmukaisen vertailuaineiston ja toteamaan tarvittaessa datan oikeellisuuden uudessa järjestelmässä. Tämä toistui lähes jokaisessa haastattelussa. Oikean vertailuaineiston generoiminen lähdejärjestelmistä tai kaikkien käsittelysääntöjen ymmärtäminen ei ole aina helppoa. Tällainen edellyttää melko kattavaa asiantuntemusta ainakin niiltä liiketoiminnan osa-alueilta, joita uuden järjestelmän on tarkoitus koskettaa. Henkilön tulisi ymmärtää myös järjestelmän taustalla olevaa tekniikkaa, eikä tällaisen moniosaajan löytäminen ei ole itsestäänselvää. Toisaalta erillisten asiantuntijoiden käyttäminen eri osa-alueilla voi hankaloittaa päätöksentekoa vertailuaineistoa ja tiedon validointia koskien. Osapuolten näkemyserot voivat viivästyttää testausprosessin ja sitä kautta projektin etenemistä.

Liiketoiminnallinen ja tekninen osaaminen edesauttavat testauksen läpivientiä, sillä näin testaukseen osallistuvat henkilöt kykenevät arvioimaan ja priorisoimaan paremmin ainakin sitä, mitä tulisi testata tai mihin tuloksia verrataan. Vastaavasti osaamisen puute

voi johtaa resurssien hukkaan. Vaikka sopivat henkilöt tunnistettaisiin, ei testauksen edellyttämää aikaa osata useinkaan resursoida oikein. Esimerkiksi henkilön C mukaan testauksen vaatima aika osataan resursoida oikein vain harvoin. Hän kertoi haastattelussa, että ”*porukka luulee, että kyllä se tosta tulee ja toi pari viikkoa riittää, niin mä ehdin sen siinä ajassa testata. Sitten on aina muut hommat siinä samaan aikaan.*” Oleellista resursoinnissa on kuitenkin ottaa huomioon testaukseen osallistuvien henkilöiden yhteinen aika, jolloin mahdollisiin ongelmiin voidaan tarttua yhteisvoimin ilman turhia viestiketjuja tai ylimääräisiä palavereja, joihin osallistutaan aina eri kokoonpanolla. Tätä korostettiin henkilöiden C, G ja H osalta. Toimittajan läsnäolo esimerkiksi hyväksyntätestauksen aikana on tärkeää, jotta havaittuihin ongelmiin voidaan puuttua välittömästi.

Henkilön A mukaan vaikeuksiin voidaan joutua myös tilanteissa, joissa testaajalla tai testaajilla ei ole rohkeutta ottaa vastuuta tekemisistään ja luottaa saatuihin tuloksiin. Testaajalla tulisi olla motivaatio saada järjestelmä tuotantoon ilman viivytyksiä ja kyky lopettaa testaus ajoissa. Pelko tuotannossa havaituista virheistä voi johtaa toimintaan, jossa testausta vältellään tai sitä jatketaan, vaikka todellisuudessa syytä ei olisi. Henkilö H kiteytti yksiselitteisesti sen, mitä testaajalta todellisuudessa vaaditaan:

> *”Testaajan ehdottomasti tärkein ominaisuus on sietää sitä, että toteutukseen jää aina virheitä. Testauksessa tärkeintä on edetä johdonmukaisesti ja keskittyä kullakin hetkellä siihen, mitä ollaan tekemässä.”*

Henkilön A mukaan ylemmältä tasolta tuleva paine järjestelmän käyttöönottamiseksi voi sujuvoittaa testausta ja eliminoida turhaa viivyttelyä. Toisaalta henkilön H mukaan kiire voi johtaa helposti epäonnistumisiin, sillä kiireessä unohtuu helposti kokonaisuus. Hänen mukaan asioita tulisi aina tarkastella laajemmin sen varalta, jos jokin on muuttunut, ja tarkastella samalla myös sitä, mihin kaikkeen tietyllä muutoksella on vaikutuksia. Kiireessä tämä voi olla mahdotonta.

Testaussuunnitelma

Toimittajan näkökulmasta testauksen onnistuminen edellyttää selkeää vastuiden jakoa sekä koordinointia. Jotta tällainen olisi ylipäätään mahdollista, esimerkiksi henkilön C mukaan ”*täytyy olla olemassa testaussuunnitelma, jossa on tietoa siitä, milloin, mitä, ja kuka testaa*.” Kaiken kaikkiaan, lähes kaikkien haastateltujen henkilöiden mukaan testaussuunnitelmassa tulisi miettiä etukäteen sitä, mitä testataan ja millä tavoin, milloin testataan ja kuka testaa. Testauksesta ei ole haastateltavien mukaan hyötyä, jos siihen ei kiinnitetä riittävää huomiota, mikä tarkoittaa tyypillisesti yksiselitteisen suunnitelman puuttumista. Jos testaukseen osallistuvat henkilöt eivät tunne vastuitaan, aikataulua tai sitä, mitä heidän tulisi testata, ajaudutaan helposti hajanaisesti organisoituun, sattumanvaraiseen ja intuitioon perustuvaan testaukseen. Henkilö G kiteytti testaussuunnitelmaan liittyvän problematiikan seuraavasti:

> *”Testauksessa ajaudutaan nopeasti väärille urille, jos testaussuunnitelmaa ei ole laadittu, eikä testausprosessi ole hallinnassa.”*

Liiallisiin yksityiskohtaisuuksiin ei kannata mennä, vaan usein on järkevintä, että asioista sovitaan karkealla tasolla. Tällöin voidaan jo ennalta ottaa kantaa muun muassa testitapauksiin ja niiden sovellettavuuteen. Järjestelmällisyyden puute voi johtaa tilanteeseen, jossa testataan sellaisiakin tapauksia, jotka eivät todellisuudessa mittaa järjestelmän luotettavuutta. Mikään ei estä sitä, etteikö virhe voisi ilmaantua myöhemmin tuotannossa. Jos esimerkiksi perusjärjestelmässä ei ole sellaista tapahtumaa, joka aiheuttaisi virhetilanteen testitapausta suoritettaessa, ei kyseinen testi ota juurikaan kantaa järjestelmän luotettavuuteen. Jos muutosten tekeminen perusjärjestelmiin on mahdotonta, tulisi tarvittava testidata tällöin generoida muulla keinoin.

Osa haastatelluista koki, että asiakkaan mielenkiinto testaussuunnitelmaa kohtaan laantuu usein siinä vaiheessa, kun tämä huomaa, että testaussuunnitelman toteuttamiseksi on tehtävä panostuksia ja uhrattava resursseja.

Testidata ja testiympäristöt

Haastateltujen näkökulmasta riippumatta haastatteluaineistosta kävi ilmi, että erot testi- ja tuotantoympäristöjen välillä saattavat johtaa siihen, ettei testaushetkellä välttämättä osata varautua tilanteisiin, joihin tuotannossa voidaan törmätä. Tällainen voi johtaa muun muassa kapeaan otokseen testitapauksia tai harhaanjohtaviin testituloksiin. Esimerkiksi henkilön C mukaan ”*on hyvä ottaa huomioon kaikki faktat, miten testiympäristö eroaa aidosta käytöstä.*” Erillisiä ympäristöjä käytettäessä testi- ja tuotantoympäristöjen ei tarvitse olla identtisiä, mutta niiden tulisi olla vertailukelpoisia. Myös testauksessa käytetyn datan ja tuotantodatan vertailukelpoisuuteen on kiinnitettävä huomiota, mikä ei välttämättä tarkoita tuotantodatan käyttöä testidatana. Tuotantodataa voidaan rajata esimerkiksi ajan suhteen, mikä tosin voi heikentää testikattavuutta. Henkilön D mukaan kokonaiskuvaa pystytään arvioimaan luotettavasti vasta, kun käsitellään koko dataa. Vastaavasti asiakkaan näkökulmasta tuotantodatasta on mahdotonta löytää tiettyjä tapauksia. Testiympäristöissä näitä tapauksia pystytään tekemään itsenäisesti, mutta uuden päivittäisen tiedon luominen on hankalaa. Oikean tasapainon löytäminen simuloidun testidatan ja tuotantodatan käytön välillä on tärkeää. Henkilön H mukaan *”testidatalla on suuri merkitys testauksen onnistumisen kannalta”*, mikä tukee edellä mainittua seikkaa.

Erityisesti henkilön A mukaan testiraporttien puuttuminen voi viivästyttää testausta, ellei raportin sisältämiä tietoja haeta suoraan tietovarastosta. Jos kehitysprosessi etenee siten, että tietovarasto rakennetaan valmiiksi ennen raportointiratkaisua, voidaan tietovarastoa tai ETL-prosessia muuttaa raporttien ehdoilla myöhemmin. Myös muiden haastateltujen näkemykset tukivat edellä mainittuja seikkoja. Prototyyppien tai testiraportti-

en hyödyntäminen ketteröittää kehitystä ja testausta, sillä virheet tunnistetaan ja niihin voidaan puuttua aiemmin. Yleisesti ottaen tämä vähentää myös kustannuksia.

Esitestaus

Toimittajan näkökulmasta kävi ilmi, että tietovarasto- ja raportointijärjestelmiä lähdetään nykyään vain harvoin kehittämään tyhjästä, sillä valtaosalla asiakkaista on jo olemassa jonkinlainen raportointiratkaisu, joka on tarkoitus korvata tai jonka rinnalle on tarkoitus rakentaa uusi ratkaisu. Korvaavan tai uuden rinnakkaisen raportointiratkaisun käyttöönotto aiheuttaa usein muutosvastarintaa loppukäyttäjissä, sillä toimintatapoja halutaan harvoin muuttaa. Uuden järjestelmän lanseerausvaihe onkin kriittinen, sillä järjestelmän täytyy vakuuttaa sen käyttäjät ja hankkia riittävä tuki. Tämä tarkoittaa sitä, että järjestelmä on testattu vikojen ja puutteiden varalta siten, että tuotannossa järjestelmä olisi luotettava ja sen käyttäminen olisi sujuvaa. Henkilö B nosti esiin esitestauksen, jossa ”*pitää olla mukana sellaiset henkilöt, jotka ymmärtää sekä ne lähdejärjestelmät että sen mihin ne loppukäyttäjät tulee sitä raporttia käyttämään.*” Haastattelun perusteella esitestaus toimii eräänlaisena suodattimena, joka pyrkii minimoimaan hyväksyntätestaukseen pääsevät virheet. Tämä parantaa loppukäyttäjien ensivaikutelmaa järjestelmästä.

Henkilön F haastattelusta nousi esiin myös esitestausta muistuttavat smoke-testit (engl. smoke tests), joiden tarkoitus on tarkistaa testattavan osan tai kokonaisuuden perustoiminta. Jos smoke-testissä törmätään ongelmiin, on testausta tyypillisesti turha jatkaa, sillä testattava osa tai kokonaisuus ei ole riittävän vakaa. Smoke-testauksen tulisi kuitenkin kattaa joukko suhteellisen kevyitä testejä, jotka eivät ole liian yksityiskohtaisia. Ennen smoke-testejä tulisi järjestelmän perusrakenteiden, kuten tietomallin, integraatioiden ja raportointikerroksen, olla valmiina. Smoke-testit tulisi tehdä ennen integraatiotestausta ja niissä keskitytään enemmän tekniseen luotettavuuteen. Henkilön B määrittämä esitestaus taas on enemmän liiketoiminnan testausta ja asioiden validointia karkealla tasolla.

Kattava yksikkötestaus

Eräs henkilön G keskeisimmistä huolen aiheista oli yksikkötestaus. Kun yksikkötestaukseen on kiinnitetty riittävästi huomiota ja se on suoritettu huolellisesti, voidaan projektissa onnistua kokonaisvaltaisesti. Jos yksikkötestauksessa lipsutaan, eikä asioita tehdä kunnolla, lähtökohdat niin testauksen kannalta kuin koko projektinkin kannalta ovat huonot. Tällöin aikataulut saattavat venyä ja ongelmat kertaantuvat myöhemmissä elinkaaren vaiheissa, mikä aiheuttaa myös suurempia kustannuksia. Huolellinen yksikkötestaus tarkoittaa suurempaa työmäärää ja resurssitarvetta, mutta kyse onkin siitä, mitä todellisuudessa halutaan. Pohdittavaksi jääkin, onko laadukas lopputulos tärkein vai arvostetaanko pienempiä kustannuksia ja näennäisesti nopeampaa aikataulua enem-

män. Aiheeseen liittyvää pohdintaa esiintyi myös useiden toimittajanäkökulmaa edustaneiden henkilöiden haastatteluissa.

Projektiryhmän sitoutuminen

Asiakkaan näkökulmasta projektiryhmän sitoutuminen oli toinen merkittävä menestystekijä. Jos projektin johto ja henkilöstö eivät kykene sitoutumaan testaukseen tai ymmärtämään sen merkitystä, lähtökohdat ovat huonot. Tämä asia tuli vahvasti esiin lähes kaikkien haastateltujen kohdalla. Esimerkiksi pelkästään huomion saaminen voi joskus olla hankalaa, jos asiaa ei nähdä riittävän merkittävänä. Käytännössä tämä voi tarkoittaa testaajien pyyntöjen tai toiveiden sivuuttamista, mikä heikentää testauksen onnistumisen mahdollisuuksia.

Määrittelyiden täydellisyys

Erityisesti asiakkaan näkökulmassa korostunut menestystekijä oli määrittelyiden täydellisyys. Jos määrittelyt puuttuvat tai ovat epätäydellisiä, testaajalla ei välttämättä ole käytössään tarvittavaa materiaalia esimerkiksi testitapausten laatimiseksi tai tulosten analysoimiseksi ja vertaamiseksi. Jos määrittelyitä ei ole käytettävissä, testaaja ei pysty tyypillisesti määrittämään, onko jokin oikein vai onko kyseessä selvitystä vaativa poikkeama. Tämä on merkittävä ongelma, sillä se voi jopa estää testaamisen. Henkilön H mukaan koko testausprosessin tulisi lähteä liikkeelle vasta, kun määrittelyt ovat kunnossa tai ainakin tarkkaan harkittuja, ja etukäteen olisi jo mietitty, mitä järjestelmältä halutaan. Tällöin testaajien ei tarvitsisi erikseen selvitellä, onko jokin oikein vai ei. Myös jatkuva uudelleen määritteleminen ja testaaminen hankaloittaa testauksen ja koko projektin läpivientiä. Henkilö H kiteytti asian ja kertoi, että *”suurimmat ongelmat syntyvät vääristä määrittelyistä.”*

6.4 Yllättävät tai usein toistuvat ongelmat

Tutkimuksessa pyrittiin myös tunnistamaan toistuvia ongelmia ja selvittämään, mistä yllättävät ongelmatilanteet yleensä johtuvat. Tällaisten toistuvasti havaittavien ongelmien tunnistaminen tuntui haastateltavista aluksi hankalalta, eivätkä he osanneet mainita mitään yksittäistä, merkittävää tuotannossa toistuvaa virhetapausta. Kun asiaa tarkasteltiin kokonaisvaltaisemmin, pohtimalla erityisesti tekijöitä, jotka olivat aiheuttaneet paljon yllätyksellisiä ongelmia, pystyttiin tunnistamaan eräs toistuva ongelma: vaihtelu lähdejärjestelmien tiedon laadussa.

Toimittajan näkökulma antoi vahvoja viitteitä siitä, että tietovarasto- ja raportointijärjestelmäprojekteissa törmätään toistuvasti yllättäviin ongelmiin lähdejärjestelmien tiedon laadun tai sen heikon tuntemuksen vuoksi. Tämä oli hyvin yleinen näkemys haastateltavien keskuudessa. Haastatteluaineistosta kävi ilmi, että tyypillinen syy esimerkiksi

ETL-latausten virhe- tai häiriötilanteisiin on usein siinä, ettei olla varauduttu poikkeaviin arvoihin lähdedatassa. Usein oletetaan, että lähdejärjestelmissä oleva data on automaattisesti laadukasta ja tietokantataulujen sarakearvot tunnetaan hyvin. Tällainen ajattelutapa johtaa valitettavan usein ongelmiin. Esimerkiksi lähdejärjestelmän tietokantataulun tietyn sarakkeen voidaan olettaa sisältävän ainoastaan kokonaislukuja, vaikka todellisuudessa osa sarakkeen arvoista sisältääkin desimaalilukuja. Tällainen tilanne voi olla esimerkiksi tuotteen ominaisuuksia kuvaavien attribuuttien kohdalla, kun tuotetta ei voidakaan esimerkiksi mitata kappaleissa. Teoreettisessa lähdeaineistossa esitetään, että ajan myötä lähdejärjestelmiin, liiketoimintasääntöihin tai yleisesti toimintatapohin liittyvät muutokset korruptoivat tietoa siten, että harvalla on kyky tietää, mitä tietokantataulut todellisuudessa sisältävät. Toisaalta ilman asianmukaisia profilointityökaluja, on haastatteluiden perusteella vaikea muodostaa kokonaiskuvaa lähdejärjestelmien tietosisällöstä. Testauksen kannalta tällaiset lähdedataan liittyvät ongelmat aiheuttavat käytännössä turhaa työtä (esimerkiksi historiadatan konvertoiminen) ja saattavat viivästyttää testausta ja muita projektin eri vaiheita. Tästä henkilöllä C oli käytännön esimerkki. Poikkeamat edellyttävät muutoksia eri kerroksilla, mikä periaatteessa johtaa eräajojen uudelleen testaukseen ja mahdollisesti myös laajempaan järjestelmätason testaukseen, kun joudutaan varmistamaan muun muassa integraatioiden toimivuus.

Henkilöiden G ja H mukaan ongelmat lähdejärjestelmien tiedon laadussa olivat merkille pantava tekijä testauksen kannalta, sillä havaittujen virheiden juurisyiden etsiminen ja validoiminen saattoi tiedon laadun vaihtelun vuoksi olla hankalaa ja aikaa vievää. Tämä johtuu heidän mukaansa usein siitä, että on vaikea tunnistaa, mistä jokin ongelma todellisuudessa johtuu. Toisaalta, monien toimittajan näkökulmaa edustavien haastateltujen mukaan huomiota kiinnitetään usein toteutukseen mutta harvoin itse lähdejärjestelmiin ja niiden tietosisältöön. Tämä voi johtua uskomuksesta, jonka mukaan lähdejärjestelmien sisältämä tieto on juuri sellaista kuin sen ajatellaankin olevan. Yksittäisiä poikkeuksia lähdejärjestelmissä on käytännössä hankala löytää ilman tiedon profilointia, ja siksi sillä on suuri merkitys myös testauksen ja siihen liittyvien toimintojen kannalta. Tätä väittämää tukivat myös henkilön G näkemykset. Lähdejärjestelmät saattavat siis sisältää poikkeuksia, joita ei normaalisti määritysten mukaan tulisi syntyä. Henkilö H kiinnitti huomiota myös päivittäislatausten kaatumisiin, vaikka toteutus olisikin kertaalleen testattu. Tällaiset ongelmat liitettiin tiedon laadun ongelmiin lähdejärjestelmätasolla.

Toisena yleisenä ongelmana haastateltavien keskuudessa pidettiin näkökulmasta riippumatta monimutkaisia transformaatioita eli muunnosprosesseja, joihin sisältyy jokin ennalta määritetty logiikka esimerkiksi lähdejärjestelmien tietokantataulujen sarakearvoihin perustuen. Testauksen kannalta ongelmallista tässä on se, ettei kyseisiä arvoja kohdetaulussa pystytä välttämättä validoimaan, koska riittävän luotettavaa vertailuaineistoa ei yksinkertaisesti ole saatavilla. Toisaalta tiedon läpinäkyvyys voi olla huono, jolloin virheiden alkuperän tunnistaminen voi olla hyvin hankalaa. Siksi kehitysvaiheessa tehty yksikkötestaus tulisi tehdä huolellisesti, mitä korostettiin erityisesti henkilöiden

G ja H kohdalla. Transformaatiot voivat perustua esimerkiksi johonkin tiettyyn liiketoimintasääntöön, jonka validoiminen voi olla mahdotonta testauksen näkökulmasta. Vaikka liiketoimintasääntöjä harvemmin kyseenalaistetaan, testauksessa tulisi kuitenkin kiinnittää huomiota viiveisiin liiketoimintasääntöjen muutosten ja käsittelyiden toteutuksen välillä. Pienet erot liiketoimintasääntöjen ja toteutuksen välillä voivat aiheuttaa havaittavia virheitä testauksessa. Henkilön D mukaan virhemarginaali voi kuitenkin olla niin pieni, ettei eroilla ole juuri merkitystä.

Järjestelmän suorituskykyyn liittyvät ongelmat eivät välttämättä ole yleisiä, mutta huoli järjestelmän suorituskyvyn riittävyydestä tietomassojen tai käyttäjämäärien kasvaessa nousi tutkimuksessa esiin toimittajan näkökulmasta joidenkin haastateltujen henkilöiden kohdalla. Usein on mahdotonta ennustaa tarkasti sitä, miten tietomassojen volyymi tai käyttäjämäärät muuttuvat tulevaisuudessa. Tietovarasto- ja raportointijärjestelmiä testataan harvoin esimerkiksi skaalautuvuuden näkökulmasta ja suorituskykytestaus on usein luonteeltaan toteavaa. Näin ollen suorituskykytestaus on käytännössä mittaamista. Henkilön D mukaan hyväksymiskriteerit suorituskyvyn osalta eivät ole yksiselitteisesti määritettävissä, sillä ”*voi olla, että 30 sekunnin vasteaika on riittävä jollekin, ja toisaalta kolmen sekunnin vasteaika voi olla liian pitkä aika jos asiaa kysytään joltain muulta.*” Joidenkin haastateltujen henkilöiden mukaan järjestelmän skaalautuvuutta tulisi miettiä jo järjestelmäarkkitehtuuria suunniteltaessa, joten suorituskykytestauksen kohdalla sitä ei useissa tapauksissa tarvitse huomioida.

6.5 Testausprosessin hyvät käytännöt

Eräs haastattelun teemoista käsitteli testausprosessin eri vaiheiden keskeisimpiä hyviksi havaittuja käytäntöjä. Vastausten vertailukelpoisuuden selkeyttämiseksi testausprosessia tarkasteltiin haastatteluiden yhteydessä teoriassa esitetyn geneerisen prosessimallin vaiheiden kautta. Tämä prosessimalli on kuvattu luvussa 4.

6.5.1 Lähtökohtien asettaminen

Testausprosessin alku on kriittinen, sillä se määrittää lähtökohdat ja asettaa näin suunnan testauksen läpiviennin onnistumisen osalta. Tutkimusaineistosta nousi erityisesti toimittajan näkökulmasta esiin kolme keskeistä tekijää, jotka tulisi ottaa alussa huomioon. Ensinnäkin vision tulisi olla kirkas ja testauksesta vastaavien henkilöiden tulisi ymmärtää, mitä testauksen läpivienti edellyttää ja missä vaiheessa varsinainen testausprosessi käynnistetään. Liiketoiminnan ymmärrys tulisi hankkia jo alkuvaiheessa. Kuten joidenkin kohdeyrityksestä haastateltujen kohdalla kävi ilmi, toimittajan on käytännössä mahdotonta luoda yksittäisen projektin puitteissa vaadittavaa ymmärrystä liiketoiminnasta. Avainhenkilöiden tunnistaminen eri liiketoiminnan osa-alueilta onkin tärkeää jo alussa, jotta kyseiset henkilöt saadaan sitoutettua sekä projektiin että testaukseen. Lisäksi henkilö H oli sitä mieltä, että liiketoiminnan substanssiosaaminen on oltava testausorganisaation hallinnassa heti alusta alkaen. Tällaiset substanssiosaajat tietävät, mistä

raportilla olevat arvot saadaan ja miten ne tulisi esittää raportilla, ja osaavat tilanteesta riippuen arvioida parhaiten sitä, miten paljon ja millaisia testitapauksia erityisesti hyväksyntätestaukseen liittyy. Testaus on yhteistyötä toimittajan, asiakkaan ja mahdollisen kolmannen osapuolen (integraattorin) välillä ja edellyttää kaikkien osapuolten sitoutumista. Näin ollen testauksen pelisäännöt tulisi sopia yhdessä ennen kuin varsinaista testaussuunnitelmaa lähdetään laatimaan. Myös henkilön G näkemykset tukivat edellä mainittuja osaamiseen liittyviä tekijöitä.

Testauksen suunnittelu tulisi aloittaa riittävän ajoissa ja se tulisi ottaa mukaan jo määrittelyvaiheessa, jossa hahmotellaan testsaussuunnitelmaa ja johdetaan testitapaukset määrittelyiden pohjalta. Henkilön G mukaan on tärkeää tarkastella ennakoivasti, mikä on suunnitelma, ja saada kokonaisvaltainen näkemys siitä, mitkä ovat testauksen kohteet, mitä vaiheita käydään läpi ja miten kokonaisuutta hallitaan. Lisäksi hänen mukaansa on muodostettava ymmärrys siitä, mitä dokumentaatiota tai määrittelyitä tulisi olla saatavilla, mitä todellisuudessa pystytään tekemään ja mitkä ovat tavat, joilla nämä asiat toteutetaan. Henkilö G pitää tällaista lähestymistapaa hyvin oleellisena, ”*että ylipäätään pystytään arvioimaan työmääriä ja riskejä sekä saada sitä viestiä myös muulle projektille ja sinne operatiivisten järjestelmien suuntaan.*”

Iteratiivisen kehityksen luonteenomaiset määrittelyt eivät välttämättä ole koskaan täysin valmiita. Sekä toimittajan että asiakkaan näkökulmasta määrittelyiden puutteellisuutta pidettiin ongelmallisena. Tästä voidaan päätellä, että määrittelyihin tulisi panostaa enemmän projektin alkuvaiheessa, jotta testaajilla olisi riittävästi luotettavaa informaatiota käytettävissään. Esimerkiksi henkilö H totesi, että hyvin tehty määrittelytyö voi suoraviivaistaa testausta ja tehdä siitä vaivattomampaa järjestelmän kehityselinkaaren eri vaiheissa:

> ”*Koko homma lähtee liikkeelle siitä, että määrittelyt on kunnossa.*”

6.5.2 Tiedon profilointi

Tiedon profilointia ei teorian mukaan perinteisesti mielletä osaksi tietovarasto- ja raportointijärjestelmien testauksen prosessimallia, mutta sen noustua vahvasti esiin tutkimusaineistossa näkökulmasta riippumatta, on liittäminen osaksi testausprosessin hyviä käytäntöjä perusteltua. Haastateltujen yleinen näkemys oli, että tiedon profilointi voi sujuvoittaa testausprosessia ja toisaalta toimia eräänlaisena lisävakuutena, jolla pyritään minimoimaan riskejä ja välttämätään yllätyksiä järjestelmän elinkaaren myöhemmissä vaiheissa. Profiloinnin avulla voidaan tutkia esimerkiksi, millaisia arvoja jokin tietokantataulun kenttä voi sisältää, kuinka pitkiä kyseiset arvot voivat olla, millainen arvojakauma on ja mikä on kyseisen kentän minimi- ja maksimiarvo lähdejärjestelmän tietyssä tietokantataulussa. Yksinkertaisimmillaan tiedon profiloinnilla voidaan tarkoittaa esimerkiksi yksinkertaisten SQL-kyselyiden tekemistä ja analysoimista ja sitä, millaista dataa kyseisen projektin kannalta kriittiset tietokantataulut sisältävät. Toisaalta, tiedon

profilointi voi olla systemaattista ja kokonaisvaltaista tietomassojen analysointia laajassa mittakaavassa. Lähestymistapa ja laajuus on kuitenkin arvioitava aina tilannekohtaisesti. Henkilö B toteaa, että laaja-alainen ja systemaattinen tiedon profilointi kuvaa lähinnä ideaalitapauksia, ja että ”*reaalimaailmassa tällainen on muutaman kyselyn kirjoittamista ja katsomista, millaista dataa lähdejärjestelmästä palautuu.*”

Sekä toimittajan että asiakkaan näkökulmasta lähdejärjestelmien tiedon laatu aiheuttaa testauksessa usein ongelmia. Lähdejärjestelmäasiantuntijat eivät välttämättä tunne järjestelmien sisältämää dataa, vaikka heillä olisikin tietämys siitä, millaista dataa järjestelmän pitäisi sisältää eri tietokantatauluissa. Ongelmiin testauksessa törmätään helposti tilanteissa, joissa lähdejärjestelmäasiantuntijat antavat karkealla tuntumalla sääntöjä ja määrittelyitä koskien tietovarastoon integroitavia lähdejärjestelmiä. Kun lähdedata poikkeaakin joltain osin näistä säännöistä tai määrittelyistä, ajaudutaan ongelmiin. Kuten haastattelut antoivat ymmärtää, testitulosten arvioiminen voi olla hankalaa ja havaittujen virheiden juurisyitä saatetaan helposti etsiä vääristä paikoista, jos lähdejärjestelmien tiedon laatuun ja määrittelyiden oikeellisuuteen luotetaan sokeasti. Henkilön G mukaan tiedon laadun vaihtelu ”*aiheuttaa ylimääräisiä selvityksiä, eikä koskaan oikein voida tietää, onko vika omassa toteutuksessa vai lähdeaineistossa.*” Selvitykset vievät aikaa ja lisäävät työkuormaa, mikä vastaavasti voi aiheuttaa lisäkustannuksia tai viivästyksiä aikataulussa. Jos testausta ei pystytä suorittamaan sovitussa ajassa, voi testaus jäädä joiltain osin jopa vaillinaiseksi, mikä johtaa lopulta testauksen epäonnistumiseen.

Kohdeyrityksen henkilöiden haastatteluista kävi ilmi, että tiedon profilointi tulisi suorittaa ennen kehitystyön sekä testauksen aloitusta viimeistään määrittelyvaiheessa. Eräs havainto oli se, että tästä voi olla apua myös vaatimusmäärittelyiden laatimisessa ja havaintoja voidaan käyttää apuna kehitystyössä ja jo yksikkötestausvaiheessa, kun transformaatioita tai ETL-prosesseja voidaan testata tunnistettujen poikkeusten varalta. Tiedon profilointi edellyttää toki integraation rakentamisen lähdejärjestelmään, mikä edellyttää jo osin kehitystyötä. Tämän vuoksi sitä on hankala soveltaa vesiputousmallin periaatteita mukailevissa kehitysprojekteissa.

Kaiken kaikkiaan haastatteluiden välittämä yleinen näkemys oli se, että tiedon profilointi on osa onnistunutta testausprosessia ja sen kautta pystytään kokonaisvaltaisemmin luomaan näkemys siitä, millaisia testitapauksia tulisi suorittaa, ja miten testit tulisi rakentaa. Henkilö G oli sitä mieltä, että tiedon profiloiminen ”*on kyllä semmoinen mihin pitäisi panostaa.*” Henkilön C mukaan ”*profilointi on osa testausta ja pohjana sille, että nähdään minkä tyyppisiä testitapauksia voi tulla.*” Henkilö D vastaavasti kertoi, että ”*usein projektin jälkeen on korostettu sitä, että kun olisi tehty.*” Henkilön A kokemuksista kävi ilmi, että asiakkaat ovat olleet hänen kohdallaan kiitollisia, kun he ovat pystyneet korjaamaan lähdejärjestelmien dataa, sillä ennen tiedon profilointia he ovat kyenneet näkemään vain yksittäisten objektien tai tietueiden dataa kerrallaan. Hyödyt voivat siis joskus johtaa jopa operatiiviselle puolelle. Tätä näkemystä tukevat myös henkilön G

kommentit. Systemaattiset tai satunnaiset poikkeamat voivat olla seurausta esimerkiksi organisaation prosesseista, toimintatavoista tai yksinkertaisesti huonosta ja epäjärjestelmällisestä tietojen hallinnoinnista. Kokonaisuuden kannalta profilointi on kannattava työvaihe. Toisaalta, henkilö E piti tiedon profilointia ajankohtaisena vain kriittisten projektien kohdalla.

6.5.3 Testaussuunnitelman laatiminen

Testausstrategian muodostaminen ja testaussuunnitelman laatiminen on testausprosessin merkittävä vaihe. Toimittajan näkökulmasta testausstrategia määrittää sen, miten testaussuunnitelma laaditaan ja miten orjallisesti testauksen eri vaiheita tulisi käydä läpi testauksen aikana. Tällaisen strategian muodostaminen on kuitenkin harvinaista, eikä asia noussut esiin esimerkiksi asiakkaita haastateltaessa.

Tutkimusaineistosta oli hyvin yleisesti havaittavissa, että testaussuunnitelma tulisi ottaa huomioon jo järjestelmän määrittelyvaiheessa. Esimerkiksi henkilön G mukaan testaussuunnitelma on käytännössä se, mistä tulisi lähteä liikkelle. Tosin kaikkea ei voida suunnitella valmiiksi, sillä tietovarasto- ja raportointijärjestelmien tehokas kehitys on käytännössä aina luonteeltaan iteratiivista. Osa haastateltavista oli myös sitä mieltä, että määrittelyvaiheen aikana voidaan punnita liiketoiminnan asettamia vaatimuksia ja etsiä kysymyksiä, joihin järjestelmän tulisi vastata. Näin voidaan tunnistaa ainakin karkealla tasolla sellaisia asioita tai kokonaisuuksia, joita tulisi erityisesti testata. Vastaavasti henkilön G mukaan on tärkeää tarkastella ennakoivasti testauskohteita, testausvaiheita ja sitä, miten kokonaisuus hallitaan ja miten sitä pilkotaan helpommin hallittaviin osiin, mitä edellytyksiä tietyt testausvaiheet sisältävät ja millä tavoin eri testausvaiheisiin liittyvät testausaktiviteetit voidaan toteuttaa kussakin tilanteessa. Myös työmäärien ja riskien arvioiminen voi olla helpompaa sekä kommunikointi eri osapuolten välillä tehokkaampaa, jos testauksesta on olemassa jokin riittävän selkeästi ja yksiselitteisesti laadittu suunnitelma. Henkilö G korosti vahvasti testaussuunnitelman merkitystä erityisesti, jos testaamiseen liittyy vaatimuksia operatiivisten järjestelmien osalta. Näin koko projektiryhmällä on mahdollisuus hahmottaa kokonaiskuva siitä, mitä tehdään ja missä vaiheessa. Myös toimittajan näkökulma korosti kokonaiskuvan hahmottamisen tärkeyttä.

Testaussuunnitelman sisältö voi vaihdella tilanteesta riippuen, mutta tutkimusaineiston perusteella voidaan tunnistaa tiettyjä hyvien käytäntöjen mukaisia, keskeisiä asiakokonaisuuksia, joihin tulisi käytännössä aina kiinnittää huomiota. Haastatteluiden perusteella kohdeyrityksen haastatelluilla henkilöillä oli selkeä käsitys testaussuunnitelman keskeisestä sisällöstä. Toisaalta, henkilöiden G ja H kohdalta ei saatu tämän tutkimuksen puitteissa riittävästi tietoa analyysin tekemiseksi. Tutkimuksessa listattiin useita erilaisia asiakokonaisuuksia, joista osa nousi haastatteluissa toistuvasti esiin. Näin ollen testaussuunnitelmassa tulisi kuvata avainhenkilöt, avainhenkilöiden roolitus, mitä vaiheita suoritetaan ja edellytykset ennen seuraavaan vaiheeseen siirtymistä. Lisäksi testaussuunni-

telmaan tulisi sisällyttää karkealla tasolla testitapauksia eri testausvaiheisiin liittyen yhdessä määrittelyiden tai vertailuaineiston kera. Vertailuaineiston määrittäminen on haastateltavien näkemysten mukaan oleellista, sillä ilman sitä testituloksia on vaikea analysoida. Jos testitapauksia ei kerätä testaussuunnitelmaan, tulisi testaussuunnitelmasta ainakin käydä ilmi, kenen vastuulla niiden laatiminen on ja miten toiminta organisoidaan. Joidenkin haastateltavien mielestä testaajien tulisi tunnistaa testausprosessin lainalaisuudet, jotka voidaan testauksen hallinnan helpottamiseksi koota yhteen testaussuunnitelmaksi. Kiteytettynä, testaussuunnitelman sisällöstä tulisi selvitä, kuka testaa, mitä testaa, miten testaa, milloin testaa ja mistä oikea vertailuaineisto saadaan eli mihin testituloksia voidaan verrata yksiselitteisesti ja luotettavasti. Näin ollen myös aikataulutus on hyvä jäsentää osaksi testaussuunnitelmaa, sillä hajanainen ja harkitsematon testien suoritus voi viivästyttää kokonaisaikataulua ja johtaa jopa testauksen epäonnistumiseen siten, että osa vaiheista jää suorittamatta. Kun testaukseen osallistuu useampia osapuolia, on tärkeää pystyä sopimaan testausajankohdista jo etukäteen, jotta osapuolet pystyvät varaamaan tarvittavan ajan. Testausvaiheiden ajankohtien määrittäminen ja ajan varaaminen yhdessä toimimiselle korostui henkilöä H haastateltaessa. Kommunikointi eri osapuolten välillä on tehokkaampaa, kun projektiryhmä on tietoinen siitä, missä mennään ja mitä seuraavaksi tulisi tehdä. Myös käyttöoikeuksien ja eri ympäristöjen välisten siirtojen kuvaaminen voi olla järkevää, sillä siirrot ympäristöstä toiseen voivat edellyttää tiettyjä asioita myös testaukselta.

Testaussuunnitelman noudattaminen voi osoittautua haastavaksi muun työn ohessa, joten testaussuunnitelmaa laadittaessa on syytä pitää mielessä projektikohtaiset realiteetit ja kuvata asioita riittävän konkreettisella tasolla. Keskeistä on, ettei testaussuunnitelmaan kirjata liian yksityiskohtaisia tai tarkkoja ohjeistuksia tai määrityksiä. Lisäksi testausprosessin alussa voi olla hankala tunnistaa, löytyykö tarvittavaa vertailuaineistoa kutakin testitapausta varten. Näin ollen testaussuunnitelmaa tulisi päivittää kehityselinkaaren aikana iteratiivisesti aina tarpeen mukaan, jotta vältyttäisiin ennen kaikkea turhalta työtä. Esimerkiksi henkilö G totesi haastattelussa seuraavasti:

> ”*Testaussuunnitelma on elävä dokumentti, joka tulisi laatia jo määrittelyiden yhteydessä ja jota tulisi päivittää aina tarvittaessa.”*

Eräiden haastateltavien mukaan testaussuunnitelman tekeminen pelkästään muodollisuuden nimissä ei tyypillisesti johda hyvään lopputulokseen. Keskeiset tekijät onnistumisen kannalta ovat yhteistyö ja vastuun ottaminen. Toimittajan vastuulla on tekninen asiantuntemus ja asiakkaan vastuulla liiketoiminnan ymmärrys. Lisäksi yhdessä toiminen auttaa sitouttamaan projektiryhmän jäseniä osaksi testausprosessia.

Eräs poikkeuksellisen havainto toimittajan näkökulmasta liittyi siihen, että testaussuunnitelmaa aloitettaessa saatetaan törmätä ongelmiin, kun asiakkaalle valkenee, että tämän on osallistuttava ja annettava oma panoksensa suunnittelutyöhön. Haastatteluista välit-

tyikin sellainen kuva, että valmis aihio tai prosessi testaussuunnitelman tekemiseksi olisi hyödyllinen, sillä tällöin alkuun pääseminen ei asiakkaan kannalta olisi yhtä haastavaa.

6.5.4 Testitapausten laatiminen

Testitapausten laatiminen on yksi testausprosessin vaiheista, mutta toisin kuin malli antaa ymmärtää, se voi tapahtua rinnakkain yhdessä testaussuunnitelman laatimisen yhteydessä. Haastatteluiden avulla pyrittiin tutkimaan, mitä periaatteita testitapausten laadinnassa tulisi ottaa huomioon ja millaisia hyviksi havaittuja käytäntöjä siihen liittyy. Toimittajan näkökulmasta huomio kiinnittyi ennen kaikkea vertailuaineistoon sekä tasoon, jolla testitapaukset tulisi kuvata. Vastaavasti asiakkaan näkökulmasta korostui testitapausten yhdenmukaisuus ja joustavuus.

Testitapausten määrittelyä ohjaa lähdejärjestelmistä saatavilla oleva data ja mahdollisuus rakentaa testitapauksia lähdejärjestelmissä. Jos testaus on tehtävä esimerkiksi suoraan toiminnanohjausjärjestelmän tuotantoympäristöstä, testitapaukset määräytyvät sen tarjoaman datan pohjalta. Jos toiminnanohjausjärjestelmästä on kuitenkin olemassa jokin testiympäristö, jossa voidaan generoida tapahtumia tai luoda rivejä, testitapausten määrittelyä ei ohjaa enää saatavilla oleva data ja testikattavuutta voidaan kasvattaa. Näin pystytään testaamaan erilaisia poikkeustilanteita paremmin. Henkilö B totesi tähän liittyen, että ”*jos meillä on joku testiympäristö, johon me päästään naputtamaan rivejä, joita halutaan testata, niin meillä on parempi mahdollisuus testata niitä poikkeuksien poikkeuksia ja kuvata nää testitapaukset testaussuunnitelmassa.*” Tässä kohden on oltava huolellinen, ettei määritetä ennalta sellaisia testitapauksia, joiden osalta ei ole lainkaan tapahtumia lähdejärjestelmissä.

Toimittajan näkökulmasta yleinen ongelma saattaa olla se, ettei asiakkaalla ole riittävää osaamista eli asiakkaalta puuttuu asiantuntemus siitä, millaisia testitapausten tulisi olla ja millä tasolla ne tulisi kuvata. Testitapausten johtaminen määrittelyinformaation pohjalta tulisi aloittaa jo määrittelyvaiheessa, sillä vertailuaineiston koostaminen voi olla työlästä ja viedä suhteellisen paljon aikaa. Esimerkiksi henkilön D mukaan testitapauksia ”*pitää miettiä ennalta, koska se ottaa paljon kalenteriaikaa niitä vertailukelpoisia lukuja kaivettaessa.*” Testattavia kokonaisuuksia tulisi siis ensisijaisesti johtaa suoraan määrittelyistä, mitä tukee valtaosa haastateltavien näkemyksistä. Näiden kokonaisuuksien mukaan voidaan luokitella yksityiskohtaisemmin kuvatut testitapaukset, jolloin testitapauksia voidaan tarkastella luokkakohtaisesti. Esimerkiksi tietty raportti voi olla yksi kokonaisuus. Tällainen lähestymistapa auttaa hallitsemaan kokonaisuutta tehokkaammin ja valitsemaan tarkoituksenmukaisia testitapauksia laaja-alaisesti. Toisaalta, henkilö H kertoi testitapausten laadinnan olevan täysin kokemusperäistä toimintaa. Tällainen kokemuksiin perustuva lähestymistapa edellyttää vankan kokemuspohjan ja ymmärryksen, kuinka prosessissa tulisi edetä. Henkilön H mukaan testitapausten laadinnassa tulisi lähteä liikkeelle kaikkein yksinkertaisimmista tapauksista ja edetä kohti monimutkai-

sempia. Myös henkilö C oli asiasta samaa mieltä. Yleisesti ottaen, jokaisen testitapauksen kohdalla tulisi arvioida, millaista panosta vertailuaineiston määrittäminen edellyttää, tai onko vertailuaineiston määrittäminen ylipäätään mahdollista. Vertailuaineistoa kuvatessa riittää usein, että havainnollistetaan yksiselitteisesti mistä ja millä tavoin vertailuaineisto muodostetaan.

Testitapausten yhteydessä joudutaan usein pohtimaan, mikä olisi sopiva tarkkuus testitapausten kuvaamiseksi kussakin tilanteessa. Erityisesti henkilö G pohti, mitä testitapauksen yhteydessä tulisi kuvata ja millä tasolla, kun testitapauksia voi olla huomattava määrä, ja kuhunkin testitapaukseen voi liittyä tietoa muun muassa vertailuaineistosta ja sen muodostamisesta sekä hyväksymiskriteereistä. Jo pelkästään hyväksymiskriteereitä voi olla suuri määrä. Tämä johtaa henkilön G mielestä siihen, että testitapauksista tehdään hyvin geneerisiä, mikä ei välttämättä aina tyydytä testaajaa. Haasteena onkin määrittää kussakin tapauksessa järkevä tarkkuus testitapausten ja niihin liittyvän informaation kuvaamiseksi. Toimittajan näkökulmasta vastaavasti selvää on, että liian yksityiskohtainen testitapausten määrittely ja kuvaaminen johtaa todennäköisesti orjalliseen ohjeiden noudattamiseen testauksessa ja vie testaajalta vapauden hyödyntää osaamistaan parhaalla mahdollisella tavalla. Testitapausten kuvaaminen atomiselle tasolle asti ei siis tyypillisesti ole perusteltua. Tällainen lähestymistapa voi olla hyvin tehokas, kunhan testitapausten laajuus on ennalta määritetty. Esimerkiksi henkilön C mukaan ”*ei ole tarkoituksenmukaista tehdä apinatestausta, vaan asiat tulisi suunnitelmissa esittää karkealla tasolla.*” Yksityiskohtainen kuvailu voi johtaa siihen, että testaajalta jää huomaamatta sellaisia ongelmia, jotka ilmaantuvat myöhemmin tuotannossa.

Yleistäminen on hankalaa, sillä lopulta taso, jolla testitapaukset määritetään, on tyypillisesti tilanne- ja projektikohtainen. Projektin alkuvaiheessa kaikkia mahdollisia testitapauksia on mahdotonta käydä läpi, mikä edellyttää periaatteessa iteratiivisen kehitysmallin soveltamista. Testitapauksia voidaan kirjata esimerkiksi testaussuunnitelmaan pitkin testausprosessia, kun määrittelyt tarkentuvat. Henkilö C korosti kuitenkin tapahtumalajien olemassa olon merkitystä lähdejärjestelmissä, mitä myös osa muiden haastateltavien näkemyksistä tuki. Jos lähdejärjestelmään ei ole tehty jotain tapahtumalajia, joka on kuitenkin otettu huomioon määrittelyissä ja testitapauksissa, ongelmat tulevat esiin tyypillisesti vasta tuotannossa. Henkilö C kuvaili vastaavanlaista tapausta seuraavasti:

> ”*Vaikka kuinka oltaisiin testattu niin nämä ei olisi koskaan tulleet esiin.*”

6.5.5 Testiympäristön pystyttäminen ja testidatan hankinta

Testiympäristön pystyttäminen sekä testidatan hankinta on eräs keskeisistä testausprosessin vaiheista. Tutkimuksessa pyrittiin selvittämään, millaisia periaatteita tässä vaiheessa tulisi noudattaa ja toisaalta mitä haasteita tämä vaihe pitää sisällään niin toimittajan kuin asiakkaan näkökulmasta katsottuna. Toimittajan asiantuntijat kiinnittivät huo-

miota testi- ja tuotantoympäristöjen väliseen vertailukelpoisuuteen, ja asiakkaan näkökulmasta välittyi, että testien luotettavuus riippuu pitkälti testidatasta. Näin ollen asiakkaan näkökulmasta ei noussut esiin mitään tiettyä edellytystä, joka tulisi ottaa huomioon testiympäristöä konfiguroitaessa.

Testiympäristö rakennetaan tyypillisesti siksi, että testaus voitaisiin suorittaa erillään kehitys- ja tuotantoympäristöistä. Näin se ei aiheuta häiriöitä tai ongelmia tuotannossa ja kehitystyötä voidaan tehdä rauhassa, kun kehitysympäristöä ei tarvitse jäädyttää testauksen ajaksi. Tietovarasto- ja raportointijärjestelmien tapauksessa voidaan käyttää joko erillisiä ympäristöjä tai yhdistetyä testi-tuotantoympäristöä. Usein kysymys on siitä, kuinka liiketoimintakriittinen tai laaja järjestelmä on kyseessä. Henkilön D mukaan testi-tuotantoympäristön hyödyntäminen pienten järjestelmien tapauksessa on yksinkertaisempaa ja edullisempaa. Tällöin ei tarvitse ylläpitää useita erillisiä järjestelmäympäristöjä tai replikoida dataa ympäristöstä toiseen. Henkilön E mukaan varsinkin suuria tietomassoja on hankala ylläpitää ja synkronoida keskenään. Toisaalta, erillisten ympäristöjen hyödyntäminen voi olla testauksen kannalta parempi vaihtoehto, jos kustannuksia tai läpimenoaikoja ei tarkastella. Henkilö B totesikin, että ”*täydellisessä maailmassa meillä on lähdejärjestelmissä testiympäristö, jossa voidaan kirjoittaa testitapaukset ja ladata data sieltä tietovaraston testiympäristöön häiritsemättä tuotantoympäristöä.*” Muun muassa henkilön C näkemys tuki tällaista erillisten ympäristöjen mallia, sillä järjestelmää pystytään testaamaan tuolloin kolmessa paikassa. Lisäksi erillinen testiympäristö mahdollistaa esimerkiksi testitapausten luomisen testiympäristössä, mikä korostui henkilön H haastattelussa. Hänen mukaansa ”*testidatalla on hirveä merkity*s.” Tuotannosta tiettyjen tapausten löytäminen on lähes mahdotonta, mutta testiympäristössä vastaavasti pystytään itse luomaan tapauksia. Tällaiset arkkitehtuuriratkaisut ovat lopulta projekti- ja organisaatiokohtaisia ja yleistyksiä on hankala tehdä. Eräs hyvä lähestymistapa voikin olla yhdistetyn testi- ja tuotantoympäristön soveltaminen, jolloin testauksessa käytetään lähdejärjestelmien tuotantodataa, mutta raportointijärjestelmää, johon data viedään, ei ole siirretty tuotantoon. Kun hyväksyntätestaus ja sitä edeltävät testausvaiheet on suoritettu hyväksytysti tuotantodatan pohjalta, voidaan raportointiratkaisu siirtää tuotantoon.

Ongelmana ei siis ole se, käytetäänkö kahden vai kolmen ympäristön ratkaisua. Kuten valtaosa kohdeyrityksen haastateltavista antoi selvästi ymmärtää, haasteena on usein erillisen testiympäristön ja tuotantoympäristön välisen vertailukelpoisuuden arvioiminen ja tunnistaminen. Jos testiympäristö ei ole vertailukelpoinen tuotantoympäristön suhteen, testitulokset eivät välttämättä ole luotettavia. Tämän vuoksi olisi hyvä ottaa huomioon kaikki ne tekijät, jotka erottavat testiympäristön tuotantoympäristöstä. Tätä näkemystä tukevat useimmat haastatelluista. Henkilön G mukaan poikkeavuudet ympäristöjen välillä voivat olla mahdollisia, mutta testauksessa harvoin kiinnitetään siihen huomiota. Usein kohdatut ongelmat liitetään testidataan ympäristöjen sijaan.

Tutkimuksessa pyrittiin myös selvittämään, millaista testidatan tulisi olla, ja mikä olisi paras tapa tuoda sitä testiympäristöön. Keskeisin näkemys oli se, että mitä enemmän testidata muistuttaa tuotantodataa sitä parempiin tuloksiin päästään. Esimerkiksi henkilö E kertoi haastattelussa yksiselitteisesti, että testiympäristöön ”*pitää tuoda saman tien se oikea data.*” Jos tuotantodataa ei syystä tai toisesta siirretä testiympäristöön, saatetaan helposti käyttää resursseja väärien löydösten selvittämiseen. Tietovarasto- ja raportointijärjestelmien testausta on lähestyttävä eri näkökulmasta kuin perinteistä ohjelmistotestausta, jossa tyypillisesti testidatalla ei ole suurta merkitystä. Toisaalta datan monistaminen voi olla hankalaa ja useamman ympäristön tietosisällön ylläpitäminen työlästä varsinkin, jos käsitellään suuria tietomassoja. Tämä korostui muun muassa henkilön G näkemyksessä. Käsittelyn helpottamiseksi tuotantodataa voidaan leikata jonkin parametrin suhteen, mutta tällainen vaikuttaa lopulta negatiivisesti testikattavuuteen ja testitapausten laatuun. Esimerkiksi henkilön D mukaan ei aina riitä, että ”*vaikka olisi testattu jonkin viikon tai päivän aineistolla jotain testiympäristössä.*” Hän jatkaa seuraavasti: ”*kun ne ladataan sinne tuotantoympäristöön, niin siellä pitää sit kuitenkin kattoo, että miltäs se näyttää se iso kuva*.” Näin ollen, jos tietovarastoon integroitavissa perusjärjestelmissä on huomattava määrä tapahtumia, ei välttämättä riitä, että testataan vain jonkin tietyn aikavälin aineistolla, vaan kokonaisuutta on tarkasteltava vielä tuotantoympäristössä muun muassa puutteiden varalta. Henkilö G koki haasteelliseksi aineistojen yhdenmukaisuuden ylläpitämisen.

Muun muassa henkilöiden A ja E mukaan tietovirtojen hallinta on tärkeää, sillä kun testiympäristöön tuodaan tuotantodataa on virta pysäytettävä muun muassa korjausten ajaksi. Henkilön E mukaan tässä asiassa usein epäonnistutaan. Testidatan ei tulisi muuttua testien suorituksen aikana. Tietovirtojen hallinnassa on otettava huomioon sekä alkulataukset että päivittäislataukset. Henkilön E mukaan tietovirrat ”*pitää saada keskeytettyä, että saadaan lataus uudestaan, kun on korjattu virheitä ja kattoo taas sen jälkeen et meneeks päivittäislataukset oikein.*”

6.5.6 Testien suorittaminen ja löydöksiin reagoiminen

Eräs testausprosessin osa-alue, johon haluttiin tutkimuksessa kiinnittää huomiota, oli testitapausten suorittaminen sekä löydöksiin reagoiminen testien suorittamisen aikana. Löydös tai havainto on tyypillisesti potentiaalinen virhe, joka voi edellyttää lisäselvityksiä. Tutkimuksessa pyrittiin selvittämään, miten testitapauksia tulisi käydä läpi, miten potentiaalisia virheitä tulisi käsitellä ja miten virheiksi luokiteltujen löydöksien korjaaminen tulisi organisoida. Tulisiko potentiaaliset virheet esimerkiksi analysoida ja korjata heti testihetkellä vai tulisiko testausta jatkaa, kunnes jokin määritelty kokonaisuus on käyty kokonaan läpi, ja ryhtyä toimenpiteisiin vasta kyseisen testikierroksen päätyttyä.

Henkilön H mukaan testitapauksia suoritettaessa on mentävä alimmalle tasolle saakka, jos halutaan, että luvut saadaan lopulta täsmäämään. Toimittajan näkökulma tukee tätä seikkaa, sillä haastatteluaineistosta viestittyi, että testitapauksia suoritettaessa tulisi

mennä atomiselle tiedon tasolle saakka. Tämä tarkoittaa yksityiskohtaista, summaamatonta dataa. Toisaalta, henkilö D oli sitä mieltä, ettei isojen järjestelmien tapauksessa esimerkiksi myyntiraporttia läpikäytäessä validoida kunkin yksittäisen tuotteen lukuja. Hänen mukaansa summatason tarkastelusta sen sijaan on hyvä aloittaa. Jos esimerkiksi summatasolla havaitaan jokin poikkeama, on etsittävä syytä alemmilta tasoilta, kunnes juurisyy poikkeamaan löytyy. Tällainen lähestymistapa vie aikaa mutta on henkilön H ja useiden kohdeyrityksestä haastateltujen henkilöiden mukaan ainoa tapa, jolla syyt poikkeamiin (virheet) saadaan paikallistettua ja korjattua asianmukaisesti.

Henkilön G mukaan testitapausten yhdenmukaisuus on tärkeää, mikä ei noussut esiin kohdeyrityksen henkilöitä haastateltaessa. Karkealla tasolla kuvatut testitapaukset ovat ongelmallisia, jos testien suorituksen yhdenmukaisuus halutaan säilyttää testaajasta tai ajankohdasta riippumatta. Tällaista näkemystä puoltaa myös osa muista haastatelluista. Suorituksen tasoon vaikuttaa juuri testaajalle annettu vapaus tehdä asioita itsenäisesti ja se, kuinka paljon lähdeaineistoja on saatavilla. Pyrkimyksenä olisi löytää sellaisia kokonaisuuksia, joilla voitaisiin varmistaa, että jokin tietty osakokonaisuus toimii luotettavasti. Kriteeristön määrittäminen tietylle tarkkuustasolle tai suoritusten laadulle voi olla hankalaa, mutta se voi olla tarpeellista ainakin siinä tapauksessa, kun halutaan välttää turhaa työtä ja liiallista ajankäyttöä.

Yhdenmukaiseen tapaan suorittaa testitapauksia kannattaa joka tapauksessa kiinnittää huomiota ja mahdollisen kriteeristön rakentamista harkita testitapausten tason ja projektin laajuuden mukaisesti. Yhdenmukaistaminen auttaa löytämään laajamittaisesti järkevän suorituksen tason, ja näin maksimoidaan testien hyötysuhteen. Tämä tarkoittaa sitä, että tehdään riittävät toimenpiteet ainakin vakavimpien virheiden havaitsemiseksi ja korjaamiseksi. Jos testaaja lähtee joka kerta pohtimaan testauksen tasoa, niin lopputuloksen laatu vaihtelee. Siksi toimintatapojen yhdenmukaistaminen on perusteltua.

Koostettaessa ja analysoitaessa tutkimusaineistoa, pystyttiin muodostamaan selkeä kuva siitä, mitkä ovat keskeiset vaiheet potentiaalisten virheiden hallinnassa. Tässä hyödynnetttiin erityisesti asiakasnäkökulmasta kerättyä aineistoa. Ensimmäiseksi henkilö H korosti sitä, että testaajien tulisi priorisoida testit siten, että testit etenevät loogisesti ja testaajilla on mahdollisuus tunnistaa yhteneväisiä ongelmia tai löydöksiä eri testitapausten välillä. Lähtökohtana on se, että yksi virhe järjestelmässä voi ilmentyä useampana poikkeamana testattaessa. Näin ollen testaajan tulisi osata tunnistaa ja luokitella ne löydökset, jotka liittyvät johonkin tiettyyn juurisyyhyn. Useiden haastateltujen mukaan löydöksiin reagoiminen onkin juuri osaamiseen liittyvä kysymys. Jos esimerkiksi raportteja testattaessa useammasta raportista havaitaan tiettyyn kenttään liittyvä poikkeama ja tiedetään, että kyseiset raportit muodostetaan samasta tietovaraston tietokantataulusta, on tärkeää ymmärtää ongelmien identtisyys. Tällaisessa tilanteessa todellisia virheitä on vain yksi, mikä tarkoittaa sitä, että testaajan ei tarvitse käytännössä kiinnittää huomiota kyseiseen ongelmaan eri raporteilla, kun juurisyyn tiedetään olevan sama jo-

kaisella raportilla. Tämä voi edellyttää sekä teknistä että liiketoiminnallista asiantuntemusta, joten yhteistyö eri osapuolten kesken on tärkeää. Esimerkiksi henkilön D mukaan ”*ensimmäiseksi tulisi selvittää, missä virhe on eli onko se toteutuksessa, ratkaisussa, vai onko se tieto juuri tolla tavalla lähdejärjestelmässä.*” Näin ollen testaajan tai testaajien osaaminen ja ymmärrys määrittää viime kädessä, kuinka sujuvasti testaus suoritetaan ja miten eri löydöksiin suhtaudutaan. Jos osaaminen on riittävää, eikä testaaja koe löydöksen pilaavan testattavaa kokonaisuutta, kannattaa testauskierros tai -sessio suorittaa loppuun. Kuten henkilö D mainitsikin, ”*olennaista on se, että kun testausresurssit on käytössä niin jokaiseen pieneen nyanssiin ei keskeytetä.*” Kun löydöksiä voidaan tarkastella laajemmasta näkökulmasta, niitä pystytään arvioimaan paremmin. Voi siis olla, että yhden virheen korjaaminen oikaisee useampia löydöksiä.

Kun yksi testikierros on viety loppuun ja löydökset on dokumentoitu, tulisi testaajien aloittaa löydösten juurisyiden etsiminen ja paikallistaminen. Virhe voi esiintyä käytännössä missä tahansa, eli se voi olla peräisin joko lähdejärjestelmästä, toteutuksesta tai itse ratkaisusta. Sen sijainti määrittää pitkälti virheen kriittisyyden sekä työmäärän virheen korjaamiseksi. Kun virhe on paikallistettu, tulisi arvioida jatkotoimenpiteiden tarve. Tämä edellyttää tietoa ainakin virheen kriittisyydestä sekä vaaditusta työmäärästä virheen korjaamiseksi. Liiketoiminta käytännössä määrää sen, voidaanko kyseisen virheen kanssa elää vai tulisiko se korjata. Toimittajan teknistä asiantuntemusta tarvitaan vastaavasti työmäärien ja toimenpiteiden arvioinnissa. Ainakin henkilöiden G ja H mukaan tekemistä tulisi priorisoida jatkotoimenpiteiden osalta. Jatkotoimenpiteet tulisi myös dokumentoida sen varalta, että kyseiseen ongelmaan törmätään uudestaan seuraavilla testikierroksilla. Vaiheet voivat olla osittain rinnakkaisia, eli löydöksiä voidaan arvioida samalla kun niitä kerätään. Tämä mahdollistaa työmäärältään pienten, toteutukseen liittyvien virheiden korjaamisen kehittäjien toimesta jo saman testikierroksen aikana, jolloin jatkotoimenpiteitä ei tarvitse siirtää seuraavalle kierrokselle, ja testi voidaan suorittaa välittömästi uudelleen. Lisäksi toimittajan näkökulmasta kävi vahvasti ilmi, ettei testausta kannata keskeyttää jokaisen löydöksen yhteydessä. Poikkeuksena voidaan pitää jotain perustavanlaatuista ongelmaa, joka vaikuttaa laajalti järjestelmän toimintaan. Tällaiset ongelmat ovat kuitenkin melko harvinaisia tietovarasto- ja raportointijärjestelmien kohdalla varsinkin, jos yksikkötestaus on suoritettu huolellisesti.

Lähes kaikkien haastateltujen näkemyksissä miellettiin ideaalitapaukseksi testisessio, johon osallistuvat samanaikaisesti kehittäjät, testaajat ja mahdollisesti myös liiketoiminnan edustajat. Tällaisen testisession organisointi voi kuitenkin olla hankalaa, sillä testikierroksen eri vaiheiden läpikäynti voi viedä paljon aikaa, eikä kaikkien osapuolten panosta tarvita jatkuvasti. Resursointi varsinkin asiakkaan päässä on haastavaa, kun substanssiosaajien pitäisi keskittyä lähinnä omien päivittäisten tehtäviensä suorittamiseen. Haasteena onkin, miten löytää tarkoituksenmukainen tekemisen taso, kehittää yhteistyötä ja saada osapuolet lopulta sitoutumaan testien läpivientiin, löydösten arviointiin ja virheiden korjailuun. Henkilö G esitti erääksi lähestymistavaksi panostamista

osapuolten väliseen kommunikointiin ja esimerkiksi päivittäisen, yhteisen hetken organisoimista osapuolten väliselle tiedonvaihdolle yhden testikierroksen aikana.

6.5.7 Dokumentointi

Dokumentointi ei ole yksittäinen vaihe, vaan kokonaisuus, joka on liitoksissa jokaiseen testausprosessin vaiheeseen. Tässä tutkimuksessa pyrittiin selvittämään, mitä testausprosessin aikana tulisi dokumentoida. Kuten haastateltavien asiantuntijoiden lausunnot antoivat ymmärtää, tietovarasto- ja raportointijärjestelmien kohdalla ei ole formaalisti määriteltyjä käytäntöjä, eikä dokumentointia yleisesti nähdä kovin tärkeäksi tekijäksi. Joissain tapauksissa vastuu dokumentoinnista saattaa kaatua lähes kokonaan toimittajan vastuulle, mikä on usein ongelmallinen tilanne. Tutkimuksessa korostuivat toimittajan näkökulmasta testaussuunnitelman sekä testitapausten dokumentoiminen ja vastaavasti asiakkaan näkökulmasta näiden lisäksi erityisesti testiaineiston dokumentoiminen.

Asiakasnäkökulmasta tarkasteltuna, henkilö H piti tärkeänä kaikkien testauksen kannalta merkityksellisten määritysten dokumentointia, koska asioiden läpikäyminen (mihin ratkaisuihin on päädytty ja miksi) joka kerta uudelleen on resurssien tuhlaamista. Yksiselitteisesti kuvatut määritykset tekevät työstä tehokkaampaa, kun jokaisen ongelman tai kysymyksen kohdalla ei tarvitse etsiä asiantuntevaa henkilöä, jolta voi saada selityksen kyseiseen asiaan. Esimerkiksi onko kyseessä todellinen virhe vai ainoastaan virheeltä ulkoisesti näyttävä tapaus. Määrittelyt toimivat myös pohjana testitapausten laadinnalle, joten kyseessä on tärkeä kokonaisuus. Henkilön G mukaan testauksen alkuvaiheessa tulisi jo määritellä selvästi, mitä määrittelydokumentaatiota täytyy laatia. Näiden määrittelyiden pohjalta johdetaan usein esimerkiksi projektin kannalta olennaisia testitapauksia. Toisaalta, toimittajan näkökulmasta välittyi selvästi kuva siitä, että usein jopa perusasioiden dokumentoinnissa on puutteita. Henkilön G mukaan perusasioita ovat esimerkiksi se, mistä data tulee, millaista se on, mihin se viedään ja miten sitä käsitellään lähdejärjestelmien ja raportointikerroksen välillä.

Lähes kaikki haastateltavat pitivät hyvin tärkeänä testitapausten dokumentoimista. Erityisesti henkilöllä H oli vahva näkemys testitapausten dokumentoinnin hyödyistä, ja hän halusi korostaakin sen merkitystä. Testitapauksia ei tulisi dokumentoida liian yksityiskohtaisella tasolla, vaan usein päästään parempaan lopputulokseen, jos kuvaukset tehdään karkeammalla tasolla. Tällainen näkemys oli myös henkilöllä G. Joidenkin henkilöiden mukaan tällainen lähestymistapa antaa vapauksia testaajalle, jolloin asiantuntevasta testaajasta on enemmän hyötyä. Henkilön H mukaan suurin hyöty syntyy siitä, kun näkyvyys tehtyyn työhön säilyy eli testaaja pystyy tarkistamaan missä tahansa vaiheessa, onko jokin tietty kokonaisuus testattu, millä tavoin ja millaisin tuloksin. Hyödyllistä on myös, jos testaaja pystyy arvioimaan luotettavalta pohjalta, onko jokin poikkeava tapaus ollut jo testausvaiheessa olemassa vai syntynyt vasta tuotannossa. Lisäksi testien toistettavuus paranee, ja testaajien on helpompi arvioida ongelman luonnetta. Näin ollen testitapausten yhteyteen tulisi kirjata muun muassa, mitä on testattu, mistä

vertailuaineisto saadaan, millaisia havaintoja on tehty, miten havaintoihin on reagoitu ja kuka on tehnyt havainnot. Nämä asiat tulivat lähes yksiselitteisesti esiin eri henkilöitä haastateltaessa. Lisäksi, jos testi on suoritettu uudelleen edelliseen havaintoon reagoimisen jälkeen, tulisi myös tästä tehdä tarvittavat kirjaukset. Tällainen aktiivinen raportointi auttaa hallitsemaan testausta kokonaisvaltaisemmin ja esimerkiksi henkilön A mukaan se toimii parhaimmassa tapauksessa myös eräänlaisena kannustimena, kun jokainen voi seurata testauksen edistymistä. Havaintojen pohjalta tehtävät työlistat ovat oma erillinen kokonaisuutensa, eikä niitä tarvitse liittää osaksi testaussuunnitelmaa tai testitapauksia. Henkilön H näkökulmasta tällaista testiaineistoa tulisi säilyttää muutamia viikkoja, jotta testeihin voidaan palata ongelmatapauksissa. Tällöin on nopeasti selvitettävissä esimerkiksi se, onko kyseessä uusi havainto.

6.6 Asiakkaan tukeminen

Tutkimuksessa haluttiin kartoittaa sekä toimittajan että asiakasorganisaation näkökulmasta sitä, miten toimittajan tulisi osata tukea asiakasta testauksen läpiviennissä. Tarkoituksena oli etsiä yhtymäkohtia eri näkökulmien välillä sekä painottaa erityisesti asiakkaan esiin nostamia ajatuksia. Toisaalta myös toimittajan näkökulma oli hyödyllinen, sillä asiantuntijoiden kokemukset asiakkaiden kanssa toimimisesta ja palautteesta antavat hyvän lähtökohdan nostaa esiin perusasioita, joita on havaittu projektista toiseen.

Henkilön G näkemyksistä esiin nousi yleisen konsultaatioavun merkitys eli ohjeistaminen ja suunnan näyttäminen sen osalta, miten testaus kannattaa toteuttaa projektikohtaisella tasolla ja toisaalta ymmärryksen rakentaminen asiakasorganisaatiossa siitä, millainen merkitys testauksella on lopputuotoksen laatuun. Kohdeyrityksestä haastateltujen vallitseva näkemys oli hyvin saman suuntainen: jo alkuvaiheessa toimittajan tulisi tehdä selväksi testauksen perusperiaatteet, joita noudattamalla voidaan saavuttaa hyvä lopputulos. Useilla haastatelluilla oli näkemys, jonka mukaan asiakas odottaa ennen kaikkea selkeiden toimintatapojen tuomista osaksi projektia, testauksen mahdollisimman sujuvaa läpivientiä varten. Tämä tarkoittaa käytännössä sitä, että toimittaja kykenee kommunikoimaan ja koostamaan asiakkaalle, miten testaus voitaisiin toteuttaa kussakin tapauksessa kustannustehokkaasti sovitussa aikataulussa. Henkilön G mukaan asiakkailla vallitsee usein sellainen käsitys, että toimittajalle on kertynyt tietämystä alan hyvistä käytännöistä. Vaikka tämän tulisikin pitää paikkansa, voi hiljainen tietämys hyvistä käytännöistä olla siiloutuneena toimittajan organisaatioon, jolloin sitä on hankala hyödyntää parhaalla mahdollisella tavalla. Hankalimmiksi tapauksiksi kuitenkin luokiteltiin ne, joissa asiakkaan puolelta ei saada mukaan sellaista substanssiosaajaa, jolla olisi riittävät tiedot liiketoiminnasta. Henkilöiden A ja H haastatteluista kävi ilmi, että tehokas yhteistyö edellyttää eri osapuolilta rohkeutta tehdä päätöksiä siitä, mikä on hyväksyttävä tulos ja mikä ei. Jos keskusteluyhteys eri osapuolten välillä jumiutuu, tekeminen hidastuu ja testitulosten saaminen viivästyy.

Yhteistyön ja testaussuunnitelman lisäksi tutkimuksessa havaittiin, että toimittajan substanssiosaaminen oli tärkeää sekä toimittajan että asiakkaan näkökulmasta. Tämä tarkoittaa käytännössä sitä, että toimittaja hallitsee kehitystyöhön liittyvän yksikkötestauksen erinomaisesti ja tuntee vastuunsa toimittamistaan ohjelmistomoduuleista, kuten esimerkiksi ETL-ajoista. Tähän kuuluu myös se, että toimittaja osaa edellyttää asiakkaalta riittävää testausmenettelyä, jotta esimerkiksi takuuehdot täyttyisivät. Toisaalta myös kokonaiskuvan ymmärtäminen tietovarasto- ja raportointijärjestelmistä sekä niiden testauksesta on tärkeää, sillä asiakas ei välttämättä ymmärrä tietomallia tai tietovaraston toimintaa tai tunne taustalla olevaa dataa riittävällä tasolla eikä täten kykene hahmottamaan asioita moniulotteisesti. Näin ollen toimittaja voi auttaa asiakasta paikallistamaan ja arvioimaan virheiden juurisyyt nopeammin, kun kokonaisuus on hallinnassa. Roolituksesta riippumatta toimittajan tulisi lisäksi antaa ohjeita siitä, mitä testausvaiheita tulisi suorittaa kussakin tapauksessa, jotta voidaan varmistua järjestelmän luotettavuudesta ja laadusta eli tässä tapauksessa siitä, vastaako järjestelmä odotuksia.

Kohdeyrityksestä haastateltujen henkilöiden ja henkilön G näkemyksistä oli havaittavissa, että asiakkaan ohjaaminen oikeaan suuntaan alusta alkaen on tärkeää, sillä projektin myöhäisemmässä vaiheessa on asetettua kurssia hankala muuttaa. Tilanteet, joissa asiakas on ottanut kokonaisvastuun testauksesta, ovat harvoin johtaneet hyviin tuloksiin. Testaus on parhaimmillaan toimittajan, asiakkaan ja mahdollisen integraattorin välistä tiivistä yhteistyötä, jossa toimittajan tulisi ottaa selkeä rooli. Varsinkin henkilön H vastauksissa korostui osapuolten yhteinen aika, jolloin keskitytään ainoastaan järjestelmän testaamiseen liiketoimintatestien sekä raporttien testauksen osalta. Näin virheisiin reagoiminen on huomattavasti nopeampaa, toiminta ketterämpää ja ongelmat saadaan usein ratkaistua nopeasti, kun eri osa-alueiden asiantuntemus on välittömästi käytettävissä ilman viiveitä. Oleellista on, että toimittaja ottaa vastuuta ja on asiakkaan käytettävissä, kun kysymyksiä herää tai ongelmia nousee esiin. Tällöin toimittajan olisi oltava nopeasti selvittämässä asiaa ja kehittämässä mahdollista ratkaisua.

Testaussuunnitelman merkitys korostui molemmissa näkökulmissa siten, että toimittajan tulisi auttaa asiakasta testaussuunnitelman laatimisessa ja ottaa suurempaa roolia kyseisessä prosessissa. Ensimmäiseksi tulisikin huolehtia siitä, että testaussuunnitelma tehdään kunnolla, vaikka testaus ei olisikaan vielä ajankohtaista. Henkilöiden G ja H haastatteluiden perusteella asiakkaat tyypillisesti tarvitsevat apua testausmenetelmien valitsemiseksi sekä testausprosessin kuvaamiseksi ja implementoimiseksi osaksi projektia. Toimittajan näkökulmasta haastateltujen kohdalla toistui usein myös testauksen laajuuden määrittäminen ja aikataulutus.

7 TULOSTEN YHTEENVETO JA SUOSITUKSET

7.1 Erityispiirteet testauksen näkökulmasta

Mitä tietovarasto- ja raportointijärjestelmällä tarkoitetaan ja mitä erityispiirteitä siihen liittyy testauksen näkökulmasta? Ensimmäisen tutkimuskysymyksen tavoitteena oli määrittää tutkimuksen kannalta keskeisin käsite eli tietovarasto- ja raportointijärjestelmä, jäsentää tutkimuksen rakennetta ymmärrettävämmäksi ja tarjota lukijalle riittävät lähtötiedot tarkasteltavaan aiheeseen. Tutkimuskysymykseen vastattiin käytännössä täysin teoreettisen aineiston pohjalta, eikä haastatteluissa otettu kantaa tietovarasto- ja raportointijärjestelmän käsitteen määrittelyyn, sillä kaikilla haastateltavilla oli laajat lähtötiedot aiheesta. Toisaalta, haastattelutilanteissa pyrittiin varmistumaan siitä, että haastattelija ja haasteltava jakoivat saman suuntaisen näkemyksen eri järjestelmäkerrosten muodostamasta järjestelmän rakenteesta. Tämä oli tärkeää, sillä testausta tarkasteltiin juuri järjestelmäkerrosten näkökulmasta. Tutkimuksen tulosten pohjalta tietovarasto- ja raportointijärjestelmällä tarkoitetaan tietojärjestelmää, jolla kerätään ja integroidaan tietoa (dataa) eri tietolähteistä, jalostetaan siitä informaatiota ja tuodaan informaatio raportointivälineiden avulla kaikkien sitä tarvitsevien saataville ajasta ja paikasta riippumatta.

Erityispiirteitä oli lähdeaineiston puitteissa mahdollista tarkastella sekä teoreettisesta että empiirisestä näkökulmasta. Teoreettinen näkemys testaukseen liittyvistä erityispiirteistä on laaja-alainen, ja siinä erityispiirteet kuvataan usein yksityiskohtaisesti. Yhtymäkohtia empiiristen tulosten suhteen oli selvästi havaittavissa. Joidenkin teemojenkohdalla empiiriset tulokset muodostivat hieman suppeamman näkemyksen verrattuna teoreettista näkökulmaan, ja joidenkin teemojen kohdalla tilanne oli päinvastainen. Näin ollen ne täydensivät toisiaan. Eri näkökulmien väliltä löydettyjen yhtymäkohtien perusteella keskeisimpänä tietovarasto- ja raportointijärjestelmien erityispiirteenä voidaan pitää tietosisällön luotettavuuden merkitystä. Näin ollen testauksen keskiössä on tiedon luotettavuuden ja oikeellisuuden varmistaminen, kun taas perinteisten ohjelmistotuotteiden kohdalla kysymys on enemmän toiminnallisuuksien testauksesta ja lähdekoodin tarkastelusta. Käytännössä tämä tarkoittaa sitä, että perinteisessä ohjelmistotestauksessa tavoitteena on ohjelmointivirheiden aiheuttamien kustannusten minimoiminen, ja vastaavasti tietovarasto- ja raportointijärjestelmien testauksen kohdalla virheellisen tai harhaanjohtavan informaation vaikutusten minimoiminen.

Tietovarasto- ja raportointijärjestelmälle ominaisia piirteitä ovat usein myös integraatioiden suuri määrä, pitkät tiedon jalostusketjut sekä monimutkaiset tiedon muunnosprosessit. Näiden ominaispiirteiden vallitessa törmätään usein tiedon laadun ja läpinäkyvyyden aiheuttamiin ongelmiin, mikä asettaa huomattavia haasteita testauksen kannalta. Tällaisia haasteita koetaan harvemmin perinteisen ohjelmistotestauksen kohdalla. Eroja perinteiseen ohjelmistotestaukseen verrattuna löytyy myös muun muassa testien suorituksen luonteesta. Esimerkiksi tietovarastoa testattaessa varsinaista käyttöliittymää ei välttämättä ole olemassa, kun taas perinteisten tietojärjestelmien kohdalla testit saatetaan suorittaa suoraan käyttöliittymästä käsin. Empiirisistä tuloksista nousi esiin myös tietovarasto- ja raportointijärjestelmän eräajopohjaisuus, mitä teoria tukee hyvin. Teorian mukaan testit suoritetaan järjestelmän käynnistäminä tapahtumina eikä ohjelmistotestauksen yleisen periaatteen mukaisesti käyttäjän käynnistäminä tapahtumina, joissa käyttäjä antaa järjestelmälle syötteen. Näin ollen tietovarasto- ja raportointijärjestelmien testauksessa ei tarvitse kiinnittää huomiota käyttäjien tekemiin operaatioihin tai toiminnallisuuksiin.

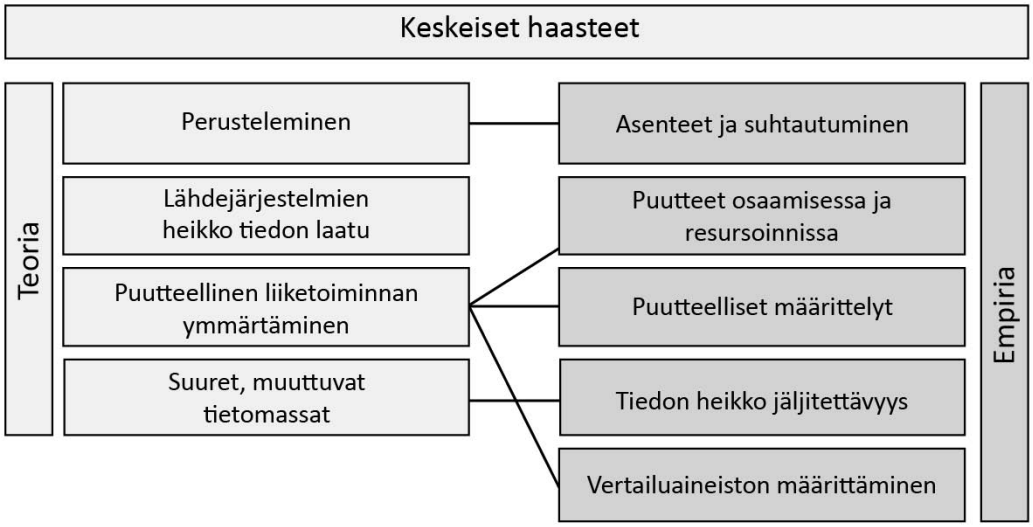
7.2 Testauksen asettuminen osaksi kehityselinkaarta

Miten testauksen tulisi asettua osaksi tietovarasto- ja raportointijärjestelmän kehityselinkaarta? Toisen tutkimuskysymyksen tavoitteena oli muodostaa laajempi näkemys siitä, miten testaus liitetään osaksi kehityselinkaarta. Testauksen asettumista osaksi kehityselinkaarta käsiteltiin teoreettisissa tuloksissa enemmän, eikä empiiristen tulosten avulla pystytä muodostamaan selkeää, yksiselitteistä näkemystä kyseisestä asiasta. Sen sijaan tuloksista pystyttiin löytämään viitteitä, jotka tukivat teoriassa esitettyjä näkemyksiä. Keskeiseksi muodostuneen näkemyksen mukaan testaus tulisi ottaa huomioon kaikissa elinkaaren vaiheissa. Näin ollen testauksen ei tulisi olla vain kertaluonteinen projektin vaihe, vaan sen tulisi toistua iteraatiosta toiseen ja jatkua myös käyttöönoton jälkeen.

Tarkasteltaessa testausta syvällisemmin, voidaan se vaiheistaa esimerkiksi testaustasojen muodostaman jaon mukaisesti. Tämä tarkoittaa käytännössä sitä, että testauksessa tulisi lähteä liikkeelle yksikkötestauksesta, joka voidaan aloittaa, kun kehitys on jo käynnistynyt ja määrittelyt sekä testaussuunnitelma on hyväksytty. Integraatio- tai järjestelmätestaus tulisi tehdä kehitystyön päätyttyä, kun kaikki testaussuunnitelman mukaiset yksikkötestit on läpäisty hyväksytysti, testiympäristö on valmiina, se on riittävän vakaa ja läpäissyt esitestit (engl. smoke tests), resurssit on varattu ja niiden käytettävyys varmistettu ja testitapaukset on laadittu, katselmoitu, priorisoitu ja hyväksytty. Tämän jälkeen tulisi siirtyä hyväksyntätestaukseen, jota voi edeltää vielä yhdeksi testauksen menestystekijäksi tässä tutkimuksessa kuvailtu liiketoiminnan esitestaus. Hyväksyntätestausvaiheeseen tulisi siirtyä, kun aiemman testausvaiheen testit on läpäisty hyväksytysti, eikä vakavia virheitä tai puutteita havaita. Lisäksi alhaisen vakavuustason virheitä tulisi olla maksimissaan testaussuunnitelman mukainen hyväksyttävissä oleva määrä.

7.3 Testauksen keskeiset haasteet

Mitä keskeisiä haasteita tietovarasto- ja raportointijärjestelmien testaukseen liittyy? Kolmannen tutkimuskysymyksen tarkoitus oli tarkastella haasteita, jotka saattavat hankaloittaa testausta ja voivat vaikuttaa negatiivisesti lopputulokseen. Teoreettiset ja empiiriset tulokset tukivat pitkälti toisiaan ja yhtenäisen näkemyksen muodostaminen oli pääosin ristiriidatonta. Tutkimuksen tulosten perusteella keskeisimmät haasteet tietovarasto- ja raportointijärjestelmien testauksessa liittyvät tyypillisesti puutteelliseen liiketoiminnan ymmärrykseen, puutteisiin osaamisessa ja resursoinnissa, asenteisiin ja ennakkokäsityksiin testauksen suhteen, heikkoon lähdejärjestelmien tiedon laatuun, lähdejärjestelmissä tapahtuviin muutoksiin, vertailuaineiston määrittämiseen, tiedon jäljitettävyyteen, suuriin, muuttuviin tietomassoihin sekä puutteellisiin määrittelyihin. Kuvassa 7.1 havainnollistetaan tutkimuksessa esiin nousseiden keskeisten haasteiden sijoittumista teoreettisen ja empiirisen näkökulman välillä. Kuvassa näkyvät viivat liittävät eri näkökulmissa esiintyneet tekijät toisiinsa ja muodostavat yhteyden näiden tekijöiden välille.



Kuva 7.1. Testauksen keskeiset haasteet

Teoreettisesta näkökulmasta puutteellinen liiketoiminnan ymmärrys on tekijä, joka vaikuttaa yleisesti testauksen onnistumiseen, ja tyypillisesti johtaa väärien asioiden testaamiseen ja sitä kautta koko testausprosessin epäonnistumiseen. Puutteellinen liiketoiminnan ymmärrys voi asiakkaan näkökulmasta ilmetä esimerkiksi kyvyttömyytenä tunnistaa testauksen kannalta olennaisia kokonaisuuksia ja näin tehdä oikeita päätöksiä siitä, mitä tulisi testata ja miten tulisi testata. Toimittajan kohdalla tilanne on tyypillisesti vaikea, sillä tämän on hankala pureutua liiketoimintaan syvällisesti käytettävissä olevien resurssien takia ja näin löytää testauksen kannalta olennaisimmat asiat. Tämä voi aiheuttaa ongelmia esimerkiksi ETL-kerroksen yksikkötestauksessa, kun liiketoimintasääntöjä

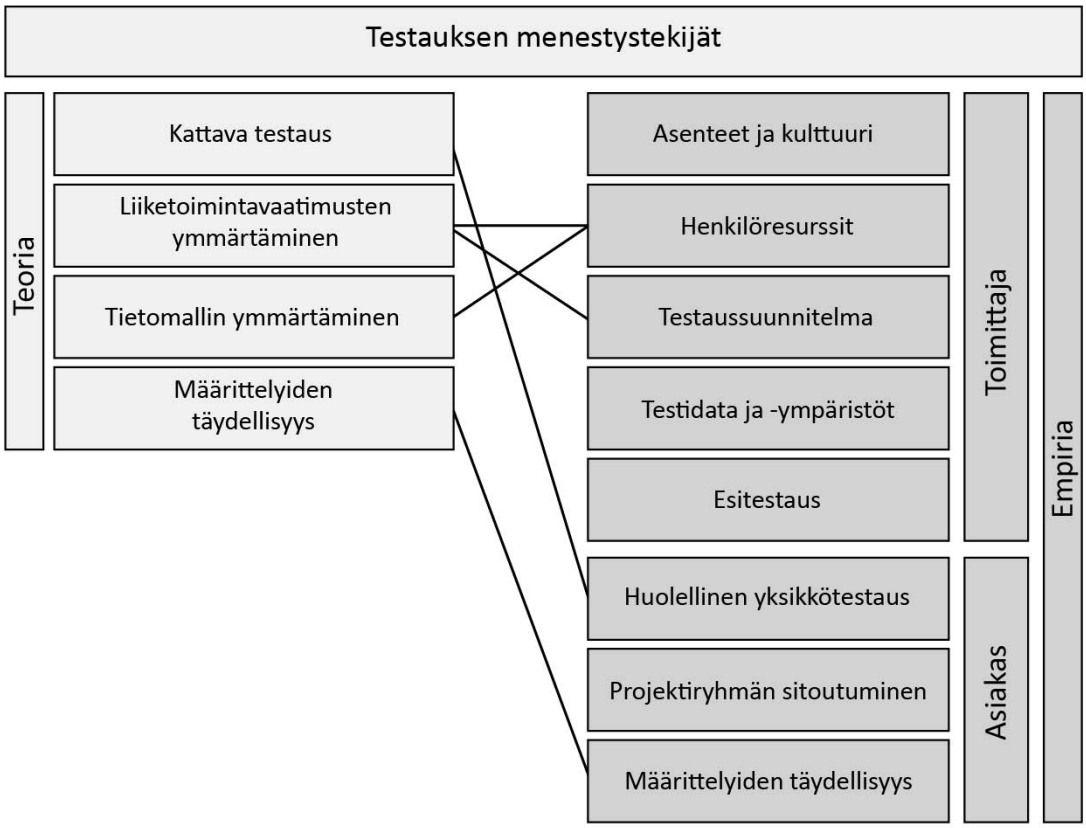
implementoidaan osaksi tiedon muunnosprosesseja. Toisaalta, liiketoiminnan puutteellinen ymmärrys liittyy suoraan eri osapuolten osaamiseen ja osin myös resursointiin. Asiantuntemuksen puute voi johtaa ongelmiin testausprosessin läpiviennissä ja aiheuttaa viiveitä esimerkiksi löydösten arvioinnissa. Resursoinnin kannalta ongelmallista on pätevien testaajien tai avainhenkilöiden löytäminen ja sitouttaminen. Avainhenkilöt, jotka ymmärtävät hyvin liiketoimintaa ja pystyvät arvioimaan löydöksiä tehokkaasti, ovat usein valmiiksi hyvin työllistettyjä. Haasteet resursoinnissa voivat johtaa tilanteeseen, jossa on vain yksi osa-aikainen testaaja. Tällainen menettely johtaa väistämättä suppeaan näkökulmaan ja kasvattaa urautumisen todennäköisyyttä. Tämä tarkoittaa käytännössä sitä, että testaaja tulee helposti sokeaksi sille, mitä hän tekee, eikä kykene näkemään asioita kokonaisvaltaisesti eri näkökulmista. Haasteiden selvittämiseksi osapuolten tulisi tehdä yhteistyötä ja tukea toisiaan, mikä edellyttää jatkuvaa kommunikointia ja ohjeistamista.

Eräs empiirisistä tuloksista esiin noussut haaste oli asenteet ja suhtautuminen. Asenteet ja suhtautuminen testausta kohtaan usein määrittää sen, millaisen jalansijan testaus saa projektissa. Koska testauksesta koituu kustannuksia, on se pystyttävä perustelemaan liiketoiminnalle. Kuten teoreettisista tuloksista käy hyvin ilmi, usein juuri perusteleminen on hankalaa, sillä lisäarvoa on vaikea havainnollistaa eikä testauksen merkitystä aina täysin ymmärretä. Testauksen merkityksen vähätteleminen voi johtaa tilanteeseen, jossa testausta esimerkiksi lykätään tai projektin aikataulua korjataan testauksen kustannuksella. Projektiväsymys ja asioiden lykkääminen myöhemmäksi voi käyttöönoton lähestyessä lamaannuttaa projektiorganisaation erityisesti tapauksissa, joissa kunnollista testaussuunnitelmaa ei ole tehty. Huolimattoman yksikkötestauksen ja harkitsemattomasti laadittujen testitapausten pohjalta vakavia virheitä saattaa päästä hyväksyntätestausvaiheeeseen asti, ja tällä voi olla vaikutuksia muun muassa käyttäjien muutosvastarintaan. Näin ollen ennalta vallitsevat asenteet ja yleinen suhtautuminen voivat ohjata testausprosessia alusta alkaen väärään suuntaan.

Teoreettisten tulosten perusteella epäonnistumiset aiheutuvat usein lähdejärjestelmien heikosta tiedon laadusta, mikä tarkoittaa puutteita, epäjohdonmukaisuuksia ja merkityseroja tiedossa, tiedon väärinkäyttöä tai yksinkertaisesti tietovarastojärjestelmän kyvyttömyyttä yhdenmukaistaa eri järjestelmistä poimittua tietoa häiriötilanteet huomioiden. Vaikka empiiristen tulosten perusteella tiedon laadun ongelmia ei koettu keskeiseksi haasteeksi, se tunnistettiin yksiselitteiseksi yhdeksi merkittävimmäksi yllättäviä ongelmia aiheuttavaksi tekijäksi. Näin ollen on perusteltua käsitellä sitä myös yhtenä keskeisistä haasteista. Tiedon laadun ohella myös yllättävät muutokset lähdejärjestelmissä ovat haasteellisia, sillä esimerkiksi epäjohdonmukaisuuksien syntyminen voi johtaa harhaan testituloksia arvioitaessa. Toisaalta, muutoksia on vaikea ottaa huomioon testauksessa ilman kokonaisuuden hallintaa.

Suuret, muuttuvat tietomassat sekä monimutkaiset transformaatiot ovat myös eräs testauksen huolenaihe, sillä ne vaikuttavat suoraan esimerkiksi tiedon validointiin ja jäljitettävyyteen tehden testien suorittamisesta ja löydösten arvioinnista hankalampaa. Empiiristen tulosten mukaan esimerkiksi virheiden tai poikkeamien juurisyiden selvittäminen voi olla hidasta monimutkaisista tiedon muunnosprosesseista johtuen. Näiden lisäksi empiirisissä tuloksissa korostettiin vertailuaineiston määrittämistä yhtenä keskeisistä testauksen haasteista. Tämä perusteltiin sillä, että kaikkien käsittelysääntöjen ja poikkeustapausten tunnistaminen on ensinnäkin hyvin vaikeaa varsinkin, jos liiketoimintaa tai lähdejärjestelmiä ei tunneta riittävän hyvin. Puutteelliset määrittelyt hankaloittavat tyypillisesti esimerkiksi sen arviointia, mitä tulisi testata ja mitkä testitapaukset mittaisivat parhaiten järjestelmän luotettavuutta.

7.4 Testauksen menestystekijät



Kuva 7.2. Testauksen menestystekijät

Mitkä ovat tietovarasto- ja raportointijärjestelmien testauksen menestystekijät? Neljäs tutkimuskysymys käsitteli testauksen menestystekijöitä, mihin oli vaikea löytää vastausta saatavilla olevan teoreettisen lähdeaineiston pohjalta. Sen sijaan empiiriset tulokset olivat melko yksiselitteisiä, ja niiden pohjalta oli helppo muodostaa yksiselitteinen näkemys testauksen menestystekijöistä. Teoreettisista tuloksista oli havaittavissa viitteitä siitä, mitä menestyminen edellyttää, ja nämä havinnot tukivat hyvin pitkälti

myös empiirisiä tuloksia. Koostetut tulokset esitetään kuvassa 7.2, jossa ne on jaoteltu näkökulman mukaisesti havainnollistamaan mahdollisia yhtymäkohtia näkökulmien välillä eri tekijöiden suhteen. Testauksen menestystekijät kiteytyivät asenteisiin, henkilöresursseihin, testaussuunnitelmaan, testidataan ja -ympäristöön, (liiketoiminnan) esitestaukseen, määrittelyiden täydellisyyteen, huolelliseen yksikkötestaukseen sekä projektiryhmän sitoutumiseen. Teoriasta lisättiin tähän vielä kattava testaus, liiketoimintavaatimusten ja tietomallin ymmärrys sekä määrittelyiden täydellisyys.

Eräs menestystekijöistä liittyy organisaation asenteisiin ja kulttuuriin, mistä ei ollut mainintaa teoriassa. Empiiristen tulosten osalta sitä kuitenkin pidettiin vahvasti yhtenä testauksen menestystekijöistä. Tämä tarkoittaa käytännössä sitä, että testauksen merkitys ja perusperiaatteet ymmärretään ja mielletään luonnolliseksi osaksi projektia. Luonnollisella osalla tarkoitetaan sitä, että testaus suoritetaan huolellisesti ja harkinnanvaraisesti eli toiminta on tavoitehakuista ja suunniteltua. Epäilevä suhtautuminen ja testauksen perusperiaatteiden tai edellytysten ymmärryksen puute syö lopulta motivaatiota, mikä johtaa helposti huonoihin kokemuksiin ja sitä kautta asenteiden muokkautumiseen. Vastaavasti positiivinen suhtautuminen ja edellytysten ymmärtäminen usein ohjaa toimia oikeaan suuntaan.

> **SUOSITUS 1:** Toimittajan tulisi ottaa proaktiivinen ote asiakkaan tukemiseksi testaukseen liittyvissä asioissa ja perehdyttää tämä testauksen perusperiaatteisiin.

> **SUOSITUS 2:** Testausta ei tulisi lähestyä liian teknispainotteisesti, sillä liiketoiminnan osallistaminen ja sitouttaminen testaukseen voi muutoin olla hankalaa.

Toisena keskeisenä menestystekijänä tutkimusten tulosten perusteella voidaan pitää henkilöresursseja. Olennaisinta henkilöiden valitsemisessa on, että testaukseen osallistuvat henkilöt kykenevät määrittämään asianmukaisen vertailuaineiston ja toteamaan tarvittaessa tiedon oikeellisuuden eri järjestelmäkerroksilla. Tämä edellyttää tyypillisesti laaja-alaista sekä teknistä että liiketoiminnallista asiantuntemusta, minkä johdosta kyvyiltään parhaiden ehdokkaiden löytäminen ja sitouttaminen voi olla haasteellista. Oikeanlaiset henkilöt osaavat määritellä selkeämmin sen, mitä järjestelmältä odotetaan, minkä johdosta he kykenevät tunnistamaan ja priorisoimaan arvioimaan paremmin sitä, mitä tulisi testata ja mihin tuloksia verrataan, sekä arvioimaan havaintoja tehokkaammin. Henkilöresursseihin voidaan liittää teoreettisista tuloksista myös liiketoimintavaatimuksien ja tietomallin ymmärtäminen, sillä ne ovat puhtaasti osaamiseen liittyviä tekijöitä. Resursoinnin kannalta oleellista on järjestää testaukseen osallistuville yhteistä aikaa, jolloin ongelmiin voidaan tarttua yhdessä tehokkaammin. Tärkeintä on kuitenkin se, että testaajat pystyvät keskittymään kullakin hetkellä siihen, mitä ovat tekemässä. Kiire ja keskittymisen puute johtaa usein siihen, että kokonaisuus unohtuu.

SUOSITUS 3: Testauksen osalta tulisi etsiä ja sitouttaa oikeat henkilöt, joilla on tarvittava osaaminen ja kyky sietää virheitä.

SUOSITUS 4: Testaukseen osallistuville henkilöille tulisi järjestää yhteistä aikaa, jotta ongelmiin voitaisiin tarttua tehokkaammin ja kokonaisvaltaisemmin. Yhteinen aika voidaan järjestää esimerkiksi sopimalla jokin tietty, yhteinen ajankohta asioiden tarkasteluun päivittäin tai viikottain.

Tutkimuksen tulosten perusteella testaussuunnitelma on yksi keskeisistä testauksen menestystekijöistä. Asioiden suunnitteleminen ja määrittely karkealla tasolla jo alkuvaiheessa säästää aikaa ja rahaa myöhemmissä elinkaaren vaiheissa testauksen osalta. Huolellisesti ja harkiten tehtynä se sujuvoittaa testausta ja auttaa pääsemään paremmin alkuun, kun jokainen projektiryhmän jäsen ymmärtää testauksen edellytykset ja tietää, mitä seuraavaksi tulisi tehdä. Testaussuunnitelmasta tulisi käydä ilmi ainakin seuraavat asiat: mitä testataan, miten testataan, milloin testataan, kuka testaa ja millaisia työvälineitä testaamisessa hyödynnetään. Lisäksi mahdollisimman moni testitapauksista tulisi kuvata yleisellä tasolla testaussuunnitelmaan. Liiallisiin yksityiskohtiin ei tulisi mennä millään osa-alueella, sillä yksityiskohtaisten ohjeistusten noudattaminen on hankalaa ja kuvaaminen todennäköisesti aiheuttaa ainoastaan turhaa työtä.

Testaussuunnitelma on liitoksissa liiketoimintavaatimuksiin ja määrityksiin, sillä niistä voidaan johtaa muun muassa, mitä tulisi testata tai mikä on määrittelyiden kannalta järkevää. Yhtymäkohtia löydettiin myös kattavan testauksen sekä määrittelyiden täydellisyyden osalta. Empiirisissä tuloksissa korostunut huolellinen yksikkötestaus on osa kattavaa testausta, sillä siinä edetään johdonmukaisesti ja tehdään tarvittavat testit harkiten kutakin yksikköä kohden. Näin ollen se toimii lähtökohtana kattavalle testaukselle. Määrittelyiden täydellisyys on myös huomioitu sekä teoreettisissa että empiirisissä tuloksissa.

SUOSITUS 5: Testaussuunnitelmaan tulisi kuvata yleisellä tasolla, mitä testataan, miten testataan, milloin testataan, kuka testaa, millaisia työvälineitä testaamisessa hyödynnetään, mitä testitapauksia käydään läpi ja mistä tai miten testitulosten arvioinnissa käytettävä vertailuaineisto saadaan.

SUOSITUS 6: Testaussuunitelmassa ei tulisi kuvata asioita liian yksityiskohtaisella tasolla varsinkaan testitapausten dokumentoinnin osalta.

SUOSITUS 7: Kun vastuu testaussuunnitelmasta siirtyy asiakkaalle, tämän mielenkiinto sen laatimiseksi voi usein laantua. Tämän vuoksi toimittajan tulisi auttaa asiakas nopeasti alkuun. Ennalta mietitty prosessi testauksen läpiviemiseksi ja valmis testaussuunnitelman sapluuna voivat tällöin olla hyödyllisiä.

Testidatan osalta tärkeänä pidettiin tuotantodatan hyödyntämistä testauksessa ja testiympäristön osalta vertailukelpoisuutta tuotantoympäristön suhteen tapauksissa, joissa käytetään erillisiä testi- ja tuotantoympäristöjä. Eroavaisuudet eri ympäristöjen välillä on myös ymmärrettävä, jotta ne voitaisiin ottaa huomioon testituloksia analysoitaessa.

> **SUOSITUS 8:** Erillisten testi- ja tuotantoympäristöjen vertailukelpoisuus tulisi aina arvioida.

> **SUOSITUS 9:** Testauksessa tulisi aina pyrkiä hyödyntämään tuotantodataa soveltuvilta osin siten, että luottamuksellinen data rajataan laajuuden ulkopuolelle.

> **SUOSITUS 10:** Tuotantodataa kannattaa usein leikata jonkin tietyn parametrin suhteen, mikä tekee testauksesta usein tehokkaampaa ja ympäristöjen ylläpitämisestä helpompaa.

Puhtaasti empiirisistä tuloksista nousivat esiin esitestaus sekä projektiryhmän sitoutuminen. Esitestaus ennen hyväksyntätestausta minimoi hyväksyntätestaukseen pääsevät virheet ja parantaa loppukäyttäjien ensivaikutelmaa järjestelmästä, mikä koettiin tärkeäksi menestystekijäksi muun muassa muutosvastarinnan kannalta. Jos järjestelmä kykenee vastaamaan käyttäjien odotuksiin paremmin jo hyväksyntätestausvaiheessa, on muun muassa muutosvastarinta todennäköisesti pienempi kuin tilanteessa, jossa huomattava määrä virheitä pääsee aina hyväksyntätestausvaiheeseen asti.

> **SUOSITUS 11:** Liiketoiminnan esitestaus kannattaa toteuttaa projektiryhmän voimin ennen varsinaisen hyväksyntätestauksen aloittamista. Näin hyväksyntätestaukseen siirryttäessä voidaan suurella todennäköisyydellä varmistua siitä, ettei vakavia virheitä tai puutteita enää esiinny.

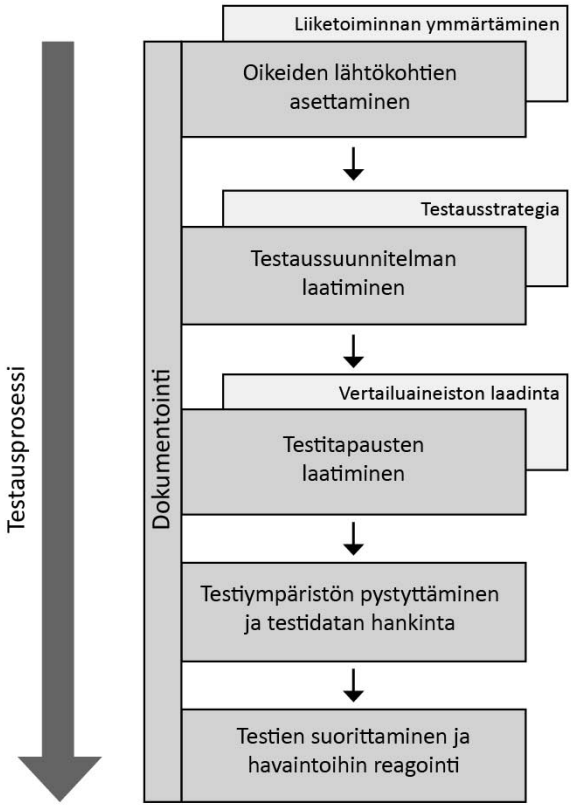
7.5 Yllättävät tai usein toistuvat ongelmat

Millaisiin yllättäviin tai usein toistuviin ongelmiin testauksessa voidaan törmätä? Viidennessä tutkimuskysymyksessä etsittiin vastausta siihen, mistä testauksen aikana syntyvät ja usein toistuvat ongelmatilanteet tyypillisesti johtuvat. Tällaisten ongelmatilanteiden ja niiden syiden tunnistaminen oli hankalaa, eivätkä teoreettiset tulokset juuri antaneet viitteitä tämän kysymyksen selvittämiseksi. Empiiristen tulosten pohjalta yllätykselliset tai usein toistuvat ongelmat liittyvät joko lähdejärjestelmien tiedon laatuun tai monimutkaisiin muunnosprosesseihin, jotka perustuvat johonkin ennalta määrättyyn logiikkaan tai korruptoituneisiin liiketoimintasääntöihin.

Lähdejärjestelmien tiedon laatu ja uskomukset tiedon laatuun liittyen ovat eräs keskeinen ongelma, johon usein törmätään. Tyypillinen seuraus on jonkin ETL-latauksen kaatuminen tai epäonnistuminen, kun lähteestä yritetään tuoda sellaisia poikkeavia arvoja, joihin ei ole alunperin varauduttu, eikä kyseistä tilannetta näin ollen ole testattu. Ongelmana on huomion kiinnittyminen pelkästään toteutukseen, eikä lähdejärjestelmiin, sillä lähdejärjestelmien tiedon laadun oletetaan tyypillisesti olevan kunnossa. Kun ajan kuluessa lähdejärjestelmiin, liiketoimintasääntöihin ja -logiikoihin tehdään muutoksia, tieto korruptoituu. Tämä johtaa siihen, että harvalla on kyky tietää, mitä eri lähdejärjestelmien tietokannat todellisuudessa sisältävät. Yksittäisiä poikkeustapauksia on hankala löytää ilman tiedon profilointia. Testauksen kannalta tällaiset ongelmat aiheuttavat helposti turhaa työtä, kun testaajat yrittävät löytää syytä toteutuksesta tai etsiä poikkeuksia lähdejärjestelmästä tunnistaakseen ja korjatakseen vian. Juurisyiden tunnistaminen saattaa olla hankalaa ja aikaa vievää. Tällöin aikataulut saattavat venyä, mikä voi johtaa esimerkiksi myöhempien elinkaaren vaiheiden viivästymiseen. Ratkaisuvaihtoehdoksi tämän tutkimuksen puitteissa on suositeltu tiedon profilointia määrittelyvaiheen aikana tai jo ennen sitä.

> **SUOSITUS 12:** Huomiota tulisi kiinnittää lähdejärjestelmien tiedon laatuun, sillä se aiheuttaa usein ongelmia myös testauksen kannalta. Tiedon profilointi auttaa löytämään poikkeustapauksia lähdejärjestelmistä ja näin toteuttamaan testauksen vaivattomammin ja tehokkaammin.

Toisena keskeisenä ongelmana voidaan pitää monimutkaisia muunnosprosesseja, joihin liittyy jokin ennalta määrätty logiikka esimerkiksi lähdejärjestelmien tietokantataulujen sarakearvoihin perustuen. Erityisen ongelmallista testauksen kannalta tämä on siksi, ettei kyseisiä kohdetaulun arvoja pystytä välttämättä validoimaan, sillä vertailuaineisto puuttuu. Monimutkaisten muunnosprosessien myötä myös tiedon läpinäkyvyys voi olla olematonta, jolloin virheiden alkuperän etsiminen ja tunnistaminen käy hankalammaksi. Ratkaisuvaihtoehdoksi tämän työn puitteissa ehdotetaan huolellista yksikkötestausta, sillä esimerkiksi liiketoimintasääntöjen validoiminen ja vaikutusten arvioiminen olisi käytännössä hyvin hankalaa.



Kuva 7.3. Tutkimuksessa korostuneet testausprosessin vaiheet

7.6 Testausprosessin hyvät käytännöt

Millainen on tietovarasto- ja raportointijärjestelmien testausprosessi ja mitä hyviä käytäntöjä siihen liittyy? Kuudes tutkimuskysymys käsitteli tietovarasto- ja raportointijärjestelmien testausprosessia, ja sen eri vaiheisiin liittyviä hyviä käytäntöjä. Tähän tutkimuskysymykseen etsittiin vastausta hyödyntämällä teoreettista prosessimallia tietovarasto- ja raportointijärjestelmien testausprosessista. Teoreettinen prosessimalli muodostuu kahdeksasta vaiheesta ja se toimi haastattelututkimuksen viitekehyksenä. Empiiristen tulosten pohjalta pystyttiin tunnistamaan testausprosessin keskeisimmät vaiheet ja muodostamaan kuvan 7.3 mukainen, yksinkertainen testausprosessin prosessimalli, johon on sisällytetty teoreettisia tuloksia soveltuvilta osin.

Testausprosessin ensimmäinen vaihe muodostaa testauksen lähtökohdat ja asettaa suunnan testauksen läpiviemiseksi. Kuten empiirisistä tuloksista käy ilmi, oikeiden lähtökohtien asettaminen on kriittinen testausprosessin alkupään vaihe, jossa pyrkimys on kerätä projektin laajuudessa tarvittava liiketoiminta- tai substanssiosaaminen yhteen ja määritellä testauksen kannalta oleelliset asiat kuten esimerkiksi hyväksymiskriteerit, vertailuaineistot ja yleisesti liiketoimintavaatimukset, joita usein koostetaan ja katselmoidaan vaiheen aikana. Teoreettisessa prosessimallissa tätä ensimmäistä vaihetta kutsutaan yleisesti liiketoiminnan ymmärtämiseksi. Teoreettisten tulosten eräs merkittävä havainto

liittyi juuri liiketoimintavaatimusten ja testauksen väliseen yhteyteen. Tutkimuksessa esiteltiin neliportainen maturiteettimalli, jonka perusteella liiketoimintavaatimukset ja testaus tulisi liittää toisiinsa siten, että testit olisivat osa vaatimuksia, ja ne suunniteltaisiin osin jo vaatimusmäärittelyn yhteydessä. Lisäksi vaatimuksista ja testeistä voitaisiin käyttää yhtä ja samaa kuvausta, jota hyödynnettäisiin päivittäisessä työssä. Näin testit tukisivat projektin päämääriä tehokkaammin. Empiiristen tulosten perusteella testauksen karkea suunnittelu tulisi aloittaa jo määrittelyvaiheen aikana, mikä tarkoittaa ensimmäisten testitapausten muodostamista. Tämä tukee näin teoriassa esitettyjä väittämiä.

Järjestelmän toteutettavuuden ja testattavuuden arvioiminen on osa alkupään määrittelytyötä, ja siinä tarkastellaan muun muassa, onko yrityksellä käytettävissään tarvittavaa dataa järjestelmän toteuttamiseksi. Tässä kohdin tulisi ottaa huomioon myös testattavuuden katselmointi, jossa tarkastellaan tiedon laatua lähdejärjestelmissä ja arvioidaan sen vaikutuksia testauksen läpivientiin. Tiedon profiloiminen projektin alkuvaiheessa voi sujuvoittaa testausta, kun tiedon laatuun liittyvien riskien vaikutukset pystytään minimoimaan ja yllätyksiltä pystytään välttymään myöhemmissä testausvaiheissa. Toisaalta, tiedon profiloinnista voi olla apua myös määrittelytyössä ja havaintoja voidaan hyödyntää myöhemmin esimerkiksi testitapauksia laadittaessa tai yksikkötestausta tehtäessä. Tiedon profiloinnin tulosten avulla tunnistetaan poikkeustapauksia, jotka voidaan ottaa jo yksikkötestausvaiheessa paremmin huomioon. Yleisesti mitä aiemmin virheet tai puutteet saadaan kiinni, sitä vähemmän kustannuksia ne aiheuttavat.

Liiketoiminnan ymmärtäminen tai lähtökohtien rakentaminen sisältää myös avainhenkilöiden tunnistamisen ja sitouttamisen, mikä tarkoittaa käytännössä riittävän substanssiosaamisen hankkimista ja sitouttamista osaksi testausprosessia. Tämä tulisi tehdä jo alkuvaiheessa, sillä avainhenkilöiden sitouttaminen voi olla hankalaa myöhemmin. Tällaiset substanssiosaajat ymmärtävät, mistä raporteille tulevat tiedot saadaan, miten ne tulisi esittää raporteilla ja mitä vasten kyseiset tiedon tulisi validoida. Testaus on yhteistyötä eri osapuolten kesken, ja pelisäännöt tulisi sopia jo ennen testaussuunnitelman laadintaa eli vision tulisi olla kirkas projektiryhmän kesken.

> **SUOSITUS 13:** Vision tulisi olla kirkas ja testausprosessin tulisi olla selkeä. Tämä tarkoittaa käytännössä sitä, että testaukseen osallistuvat henkilöt ymmärtävät, mitä testauksen läpivienti edellyttää muun muassa osaamisen ja aikataulujen kannalta.

> **SUOSITUS 14:** Liiketoimintavaatimukset ja testaus tulisi kytkeä toisiinsa siten, että testit olisivat osa liiketoimintavaatimuksia ja ne suunniteltaisiin ainakin keskeisiltä osin jo määrittelyvaiheessa.

> **SUOSITUS 15:** Riittävä liiketoiminta- tai substanssiosaaminen tulisi hankkia ja sitouttaa heti alusta alkaen.

> **SUOSITUS 16:** Esivalmisteluihin kuten määrittelyyn ja tiedon profilointiin tulisi panostaa. Huolellisesti tehdyt määrittelyt suoraviivaistavat testausta ja toisaalta tiedon profiloinnista on hyötyä monesta näkökulmasta, sillä sen avulla pystytään luomaan laaja-alaisempi näkemys siitä, millaisia testejä tulisi suorittaa ja miten testit tulisi rakentaa.

Testaussuunnitelman laatiminen on keskeinen osa testausprosessia, ja se tulisi tehdä heti testausprosessin ensimmäisen vaiheen jälkeen. Haastatteluiden perusteella testaussuunnitelma jää usein tekemättä tai se jää hyvin vaillinaiseksi dokumentiksi, jota ei lopulta noudateta. Testaussuunnitelmaan tulisi kuitenkin panostaa, vaikka sen noudattaminen onkin usein haastavaa muiden töiden ohessa. Tämän ohella tulisi kuitenkin muistaa, että testaussuunnitelman tekeminen muodollisuuden vuoksi ei ainakaan empiiristen tulosten mukaan johda sekään hyvään lopputulokseen. Tällöin resurssit menevät hukkaan, eikä tehdystä työstä saada aikaan varsinaista lisäarvoa.

Teoreettisten tulosten mukaan testaussuunnitelmassa tulisi lähteä liikkeelle määrittelemällä testausstrategia, mikä tarkoittaa karkealla tasolla seuraavien kysymysten pohdintaa: mitä tavoitteita tulisi asettaa, mikä on olennaista projektin kannalta ja millä tavoin päästään parhaaseen lopputulokseen. Testausstrategia voi kuulostaa utopistiselta ajatukselta mutta sen keskeinen on ajatus on, että testauksen tulisi olla tavoitteellista toimintaa, joka tukee projektin päämääriä. Tällöin testauksen perusteleminen voi olla helpompaa, kun tarkastelua pystytään laajentamaan ja kokonaisuus hahmottamaan paremmin. Toisaalta, myös empiiriset tulokset tukevat tätä seikkaa. Niiden mukaan testausstrategian avulla testaus voidaan kytkeä osaksi projektin päämäärää, jolloin todennäköisemmin kiinnitetään huomiota oikeisiin asioihin. Toinen olennainen tekijä on selvittää, mitä testausstrategian toteuttaminen edellyttää ja miten edistymistä voidaan mitata.

Tyypillisesti kaikkea ei voida määrittää heti alussa, vaan testaussuunnitelman tekeminen edellyttää iteratiivisuutta etenkin tietovarasto- ja raportointijärjestelmien tapauksessa. Olennaista on lähteä liikkeelle isoista linjoista ja määritellä asioita karkealla tasolla. Tärkeää on muodostaa selkeä kuva siitä, miten kokonaisuutta hallitaan ja pilkotaan helpommin hallittavissa oleviin osiin. Testaussuunnitelmaa tulisi tämän jälkeen päivittää aina tarvittaessa ja se edellyttää, että muutoksista kommunikoidaan eri osapuolille. Vastuun ottaminen ja yhteistyö eri osapuolten välillä korostuu, sillä testaussuunnitelmaa ei voida tyypillisesti suorittaa pelkästään toimittajan tai asiakkaan toimesta. Usein toimittajalla onkin päävastuu teknisestä asiantuntemuksesta sekä testaukseen liittyvistä perusperiaatteista ja asiakkaalla vastaavasti liiketoiminnan ymmärryksestä.

Testaussuunnitelman sisältö voi vaihdella, mutta keskeisen sisällön osalta voidaan löytää yhtymäkohtia teoreettisten ja empiiristen tulosten välillä. Teoreettisten tulosten mukaan testaussuunnitelmasta tulisi käydä ilmi testaajat, työkalut, riskit, aikataulutus, tes-

tausmenetelmät ja testit sekä testiympäristöt. Empiiriset tulokset myötäilevät pitkälti tätä listaa, sillä niiden mukaan testaussuunnitelmasta tulisi käydä ilmi ainakin avainhenkilöt, vastuut ja roolitus, testausvaiheet, resursointi sekä karkean tason testitapaukset vertailuaineiston määrityksineen. Näin ollen testaussuunnitelmasta tulisi selvitä, kuka testaa, mitä testaa, miten testaa, milloin testaa ja mistä tai miten oikea vertailuaineisto saadaan. Yleisesti testaussuunnitelmaan ei kannata kirjata liian yksityiskohtaista informaatiota, sillä turhaa työtä kannattaa välttää. Kuten teoriassa on esitetty, kullekin testausvaiheelle tulisi olla oma yksityiskohtaisempi testaussuunnitelma. Näin kokonaisuutta voidaan jakaa pienempiin osiin, mikä parantaa kokonaisuuden hallittavuutta. Testaussuunnitelman tulisi sisältää tai siitä tulisi käydä ilmi erilliset suunnitelmat eri testaustasojen läpikäymiseksi.

> **SUOSITUS 17:** Testaussuunnitelmassa tulisi ottaa huomioon testausstrategia, jonka avulla testaus kytketään osaksi projektin päämääriä ja toiminta tavoitteellistetaan.

> **SUOSITUS 18:** Testaussuunnitelman tekeminen edellyttää yhteistyötä eri osapuolten välillä ja sen tulisi olla iteratiivista, sillä kaikkea ei voida määrittää heti alkuvaiheessa.

> **SUOSITUS 19:** Testaussuunnitelmasta tulisi löytyä koko testausprosessin kannalta keskeinen sisältö karkealla tasolla määritettynä. Testausuunnitelmaan ei tulisi kuitenkaan kuvata asioita liian yksityiskohtaisella tasolla, sillä liian yksityiskohtaisten ohjeistusten noudattaminen on hankalaa, eikä se palvele tarkoitusta.

> **SUOSITUS 20:** Testaussuunnitelmassa tulisi tunnistaa olennaiset kokonaisuudet ja luokitella testitapaukset näiden kokonaisuuksien mukaisesti parantaen kokonaisuuksien hallittavuutta.

Teoreettisessa prosessimallissa esiintynyt testiennusteiden laadinta ja odotettavien tulosten arviointi sisällytetään tämän tutkimuksen osalta osaksi testitapausten laadintaa. Empiiristen tulosten perusteella testitapausten laatiminen tulisi aloittaa jo määrittelyvaiheessa, sillä vertailuaineiston määrittäminen voi olla työlästä ja viedä suhteellisen paljon aikaa. Testausprosessin alkupään vaiheita voidaan suorittaa usein rinnakkain, joten tätä vaihetta voidaan oikeastaan kuvata paremmin testausprosessin osana. Testitapausten laadinnassa kokonaisuuden hallinta on tärkeää. Eräs tapa on esimerkiksi muodostaa vaatimusmäärittelyiden ja testaussuunnitelman pohjalta testattavia kokonaisuuksia, joita voidaan luonnehtia eräänlaisiksi luokiksi. Tämän jälkeen voidaan luokkakohtaisesti tarkastella sitä, mitä asioita kannattaa testata ja miten. Näin voidaan hallita kokonaisuutta tehokkaammin ja estää muun muassa ekvivalenttien testitapausten syntyminen, mikä tuli esille myös teoreettisissa tuloksissa. Työmäärän ja testitapausten määrän minimoimiseksi tulisi pyrkiä testaamaan useampia asioita samanaikaisesti, mikä edellyttää te-

hokkaita testitapauksia. Oleellista on lähteä liikkeelle yksinkertaisista tapauksista ja edetä järjestelmällisesti kohti monimutkaisempia.

Empiirisissä tuloksissa huomio kiinnittyi erityisesti vertailuaineiston laadintaan ja tasoon, jolla testitapaukset tulisi kuvata. Jokaisen testitapauksen kohdalla tulisikin arvioida sitä, millaista työpanosta vertailuaineiston määrittäminen edellyttää tai onko tällainen määrittely ylipäätään mahdollista. Vertailuaineistoa kuvatessa usein riittää, että havainnollistetaan yksiselitteisesti mistä ja millä tavoin se muodostetaan. Kun testitapauksia on huomattava määrä ja niihin liittyy paljon informaatiota, joudutaan vertailuaineiston ohella usein pohtimaan sitä, millä tarkkuudella testitapauksia tulisi kuvata. Liian yksityiskohtainen määrittely on työlästä ja johtaa orjalliseen ohjeiden noudattamiseen, jolloin testaajan osaamista ei saada hyödynnettyä parhaalla mahdollisella tavalla.

Tyypillisesti testitapausten laadintaa ohjaa saatavilla oleva data ja mahdollisuus rakentaa testitapauksia manuaalisesti esimerkiksi lähdejärjestelmien testiympäristöissä. Mikäli tämä on mahdollista, voidaan testikattavuutta kasvattaa, jolloin myös poikkeustapausten testaaminen on tehokkaampaa.

> **SUOSITUS 21:** Testitapauksia johdettaessa tai määritettäessä tulisi lähteä liikkeelle yksinkertaisista ja edetä kohti monimutkaisempia tapauksia. Kokonaisuutta tulisi pilkkoa pienempiin, helpommin hallittavissa oleviin osiin luokittelemalla testitapauksia.

> **SUOSITUS 22:** Testitapauksia ei tulisi kuvata liian yksityiskohtaisesti. Tarkkuus tulisi määrittää projektikohtaisesti ja suoritusten tasoa tulisi pyrkiä yhdenmukaistamaan testaajasta riippumatta.

Testiympäristön pystyttäminen ja testidatan hankinta on seuraava prosessimallin vaiheista ja muodostaa osa-alueen, jonka kohdalla teoria tuki hyvin empiirisiä tuloksia. Toisaalta, teoreettista lähdeaineistoa tutkittaessa oli havaittavissa, että kyseistä osa-aluetta oli tutkittu melko vähän. Tutkimuksen tuloksista voidaan päätellä, että huomiota tulisi kiinnittää testi- ja tuotantoympäristöjen välisen vertailukelpoisuuden arviointiin ja eroavaisuuksien tunnistamiseen. Jos testiympäristö ei ole vertailukelpoinen tuotantoympäristön suhteen, testitulokset eivät välttämättä ole päteviä. Tämän vuoksi olisi hyvä ottaa huomioon kaikki ne tekijät, jotka erottavat testiympäristön tuotantoympäristöstä, ja ymmärtää, kuinka mahdolliset erot näiden ympäristöjen välillä vaikuttavat testituloksiin. Huomiota tulisi kuitenkin kiinnittää enemmän testidataan, sillä testidatan tulisi periaatteessa kattaa kaikki mahdolliset liiketoimintatapahtumat. Tämän vuoksi testauksessa tulisikin hyödyntää suoraan tuotantodataa, luottamuksellista tietoa lukuunottamatta. Keskeinen ajatus on se, että mitä enemmän testeissä käytetty data muistuttaa tuotantodataa, sitä parempiin tuloksiin päästään. Toisaalta suurten tietomassojen monistaminen ja ylläpitäminen voi olla työlästä, minkä vuoksi tuotantodataa kannattaa tyypillisesti leikata jonkin

ennalta määritetyn parametrin (esimerkiksi aikaväli) suhteen. Usein liiketoimintavaatimukset antavat viitteitä tällaisten parametrien valitsemiseksi.

> **SUOSITUS 23:** Erillistä testi- ja tuotantoympäristöä sovellettaessa tulisi varmistaa ympäristöjen välinen vertailukelpoisuus ja pitää huolta aineistojen yhdenmukaisuudesta. Eräs hyvä lähestymistapa on soveltaa yhdistettyä testi- ja tuotantoympäristöä, jolloin testauksessa käytetään lähdejärjestelmien tuotantodataa, mutta raportointijärjestelmä, johon data viedään ei ole varsinaisesti tuotantoympäristössä. Kun yksikkötestaus, esitestaus ja hyväksyntätestausta on suoritettu tuotantodatan pohjalta, voidaan raportointiratkaisu siirtää tuotantoon.

> **SUOSITUS 24:** Jos testausta halutaan tehostaa, tuotantodataa kannattaa leikata jonkin parametrin suhteen. Luottamuksellista tietoa ei kuitenkaan koskaan tulisi tuoda testiympäristöön.

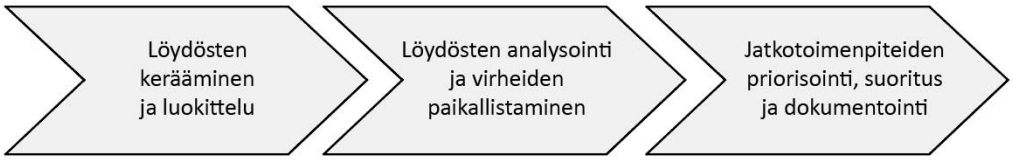
Tässä tutkimuksessa testausprosessin teoreettista prosessimallia yksinkertaistettiin ja sen viimeiseksi vaiheeksi määritettiin testien suorittaminen ja havaintoihin reagoiminen. Näin ollen teoreettisessa prosessimallissa esitetty käyttöönottovaihe jätettiin huomioimatta, sillä sen on tarkoitus kuvata lähinnä testauksen jatkuvuutta myös käyttöönoton jälkeen. Tutkimuksessa pyrittiin selvittämään, mitä huomion arvoisia seikkoja testien suorittamiseen liittyy, miten löydöksiä tulisi käsitellä ja arvioida sekä miten löydöksiin kohdistuvat korjaustoimenpiteet tulisi organisoida.

Empiriassa esitettyjen tulosten mukaisesti testaajien tulisi ensiksi priorisoida testit siten, että ne etenevät loogisessa järjestyksessä ja että testaajat kykenevät tunnistamaan mahdollisesti yhteneväisiä ongelmia ja muodostamaan näin kokonaisuuksia, joihin voi liittyä sama juurisyy. Tämä edellyttää sekä teknistä että liiketoiminnan asiantuntemusta, joten yhteistyö eri osapuolten kesken on tärkeää. Jos löydökset eivät estä testausta, kannattaa yksi testisessio tai testikierros suorittaa loppuun, ennen kuin löydöksiä arvioidaan tarkemmin korjaustoimenpiteiden osalta. Kun yksi testikierros on saatettu loppuun ja löydökset dokumentoitu, tulisi testaajien aloittaa löydösten juurisyiden etsiminen ja paikallistaminen. Tämän jälkeen tulisi arvioida jatkotoimenpiteiden tarve, mikä edellyttää ainakin tietoa löydöksen vakavuudesta, vaikutuksista ja mahdollisten korjaustoimenpiteiden työmäärästä. Myös jatkotoimenpiteet tulisi dokumentoida, jos esimerkiksi samaan ongelmaan törmätään uudelleen seuraavien testikierrosten aikana. Kun löydöksiä tarkastellaan laaja-alaisemmin, niitä pystytään luokittelemaan ja arvioimaan tehokkaammin. Voi siis olla, että useampi poikkeama järjestelmässä voi johtua samasta juurisyystä, jonka korjaaminen oikaisee samalla kertaa monta poikkeamaa tai virhettä.

Empiiristen tulosten perusteella ideaalitapausta voitaisiin kuvailla tilanteena, jossa yhteen testikierrokseen tai -sessioon osallistuvat samanaikaisesti kehittäjät, testaajat ja tarpeen vaatiessa substanssiosaajia. Koska tällaisten sessioiden järjestäminen on hanka-

laa resursointisyistä, eräs hyvä lähestymistapa voisi olla panostaminen osapuolten väliseen kommunikointiin ja organisoida päivittäinen hetki osapuolten väliselle tiedonvaihdolle testikierroksen aikana.

> **SUOSITUS 25:** Testien suorittaminen tulisi olla systemaattista toimintaa, joka voidaan organisoida esimerkiksi kuvan 7.4 esittämällä tavalla. Keskeinen idea on edetä loogisesti ja pyrkiä tunnistamaan löydöksiä, jotka voivat olla seurausta samasta juurisyystä.



Kuva 7.4. Ongelmiin reagoiminen ja yhden testikierroksen vaiheet

> **SUOSITUS 26:** Eräs hyvä lähestymistapa suorittaa testejä on järjestää testaukseen osallistuville henkilöille yhteistä aikaa katselmoida ja arvioida löydöksiä sekä päättää jatkotoimenpiteistä.

Tulosten pohjalta muodostettuun prosessimalliin on lisätty dokumentointi omana kokonaisuutenaan. Tämä kokonaisuus puuttuu teoreettisesta prosessimallista, mutta empiiristen tulosten mukaisesti se on olennainen osa testausprosessia. Sitä ei voida pitää yksittäisenä vaiheena vaan kokonaisuutena, joka on liitoksissa kaikkiin testausprosessin eri vaiheisiin.

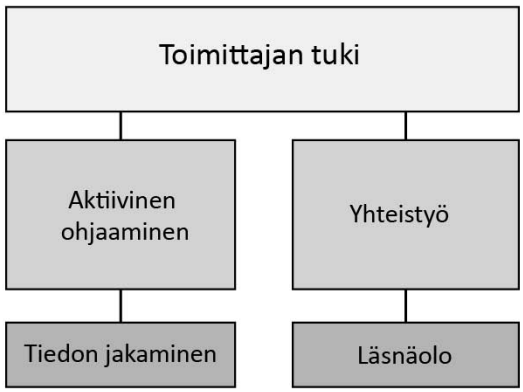
Tuloksissa korostuivat erityisesti määrittelyiden ja testitapausten dokumentoiminen. Määrittelyiden huolellinen dokumentoiminen oli tämän tutkimuksen tulosten mukaan oleellista, sillä se voi sujuvoittaa testausta merkittävästi. Hyvien määrittelyiden pohjalta ongelmien ratkominen ja kokonaiskuvan hahmottaminen on helpompaa. Näin ollen testaajan ei tarvitse käyttää resursseja turhaan erilaisiin kyselyihin ja tiedon hankintaan kerta toisensa jälkeen. Yhtenäinen malli tarvittavasta määrittelydokumentaatiosta edesauttaa dokumentoinnin hallintaa.

> **SUOSITUS 27:** Testauksen hallinnan kannalta olisi suotavaa, että määrittelydokumentaation tuottamiseksi olisi olemassa yhdenmukainen malli. Oleellinen määrittelydokumentaatio tulisi olla testaajien saatavilla ajasta ja paikasta riippumatta.

Testitapausten dokumentoinnin yhteydessä vastaavasti oleellisinta on muistaa, ettei liiallinen yksityiskohtaisuus ole hyväksi, vaan paras lopputulos saavutetaan tyypillisesti karkean tason kuvauksilla. Tällainen lähestymistapa antaa enemmän vapauksia testaajalle ja vähentää testitapausten laatimiseksi vaadittua työmäärää. Eräs dokumentoitujen testitapausten suurimmista hyödyistä on kokonaisvaltainen näkyvyys tehtyyn työhön. Tällöin testaaja osaa sanoa, onko jokin kokonaisuus jo testattu, millä tavoin ja millaisia havaintoja siihen liittyi. Havaintoja voidaan arvioida paremmin, kun testaaja pystyy esimerkiksi määrittämään, onko jokin poikkeama havaittu jo testausvaiheessa vai onko kyseessä kokonaan uusi, tuotannossa syntynyt virhe. Näin ollen näkyvyys parantaa testien toistettavuutta ja kokonaisuuden hallintaa. Tämä tarkoittaa esimerkiksi myös sitä, että uusien testaajien perehdyttäminen on helpompaa. Kaiken kaikkiaan testitapausten yhteyteen tulisi kirjata, mitä on testattu, mistä ja miten vertailuaineisto saadaan, millaisia havaintoja on tehty, kuka on tehnyt havainnot ja miten havaintoihin on reagoitu.

> **SUOSITUS 28:** Testitapausten dokumentoinnin tulisi muodostaa kokonaisvaltainen näkyvyys testeihin liittyvään informaatioon. Tämä tarkoittaa karkealla tasolla kuvattuihin testitapauksiin liittyvien, oleellisten tietojen ylläpitämistä ja säilyttämistä testauksen ajan.

7.7 Asiakkaan tukeminen



Kuva 7.5. Kuinka toimittajan tulisi tukea asiakasta testauksen osalta.

Eräs tutkimuksessa tarkastelluista kysymyksistä oli se, millä tavoin toimittajan tulisi osata tukea asiakasta testauksen aikana. Se nostettiin tutkimuksessa esiin, vaikka se ei muodostanutkaan varsinaista tutkimuskysymystä. Sen tarkastelu on kuitenkin hyödyllistä tutkimuksen kohdeyrityksen kannalta, sillä se laajentaa näkökulmaa tutkimusongelman suhteen. Tutkimuksen tuloksissa korostui kaksi tekijää: asiakkaan aktiivinen ohjaaminen ja yhteistyö (kuva 7.5).

Asiakkaan ohjaaminen oikeaan suuntaan on tärkeää alusta alkaen, sillä asetettua kurssia on projektin myöhemmissä vaiheissa hankala muuttaa. Tämä liittyy olennaisesti myös testausprosessin alkuvaiheeseen, jossa tehdään esivalmistelut sopivien lähtökohtien asettamiseksi. Oleellista on, että toimittaja kykenee proaktiivisesti kommunikoimaan asiakkaan kanssa ja tuomaan kokonaisvaltaisen näkemyksen, jonka pohjalta asiakkaan on helpompi lähestyä testaukseen liittyviä kokonaisuuksia. Tutkimuksen mukaan asiakkaat tarvitsevat usein apua esimerkiksi testausmenetelmien valitsemiseksi sekä testausprosessin kuvaamiseksi ja implementoimiseksi osaksi projektia. Näin ollen toimittajan tulisi pystyä koostamaan asiakkaalle, miten testaus voitaisiin toteuttaa kussakin tapauksessa kustannustehokkaasti ja aikataulussa. Kaiken kaikkiaan, ohjeistaminen ja oikean suunnan näyttäminen edellyttää jatkuvaa kommunikointia.

Yhteistyö on toinen merkittävä tekijä, joka sekin liittyy kommunikointiin. Tutkimuksen perusteella testaus on parhaimmillaan eri osapuolten välistä tiivistä yhteistyötä, jossa eri osapuolilla on selkeät vastuut. Asiakas ei välttämättä ymmärrä kaikkia järjestelmään liittyviä teknisiä ratkaisuja tai toimintaperiaatteita tai tunne taustalla olevaa dataa sillä tasolla, että kykenisi hahmottamaan asioita riittävän moniulotteisesti. Näin ollen toimittaja voi auttaa asiakasta esimerkiksi virheiden juurisyiden paikallistamisessa ja arvioimisessa.

> **SUOSITUS 29:** Toimittajan tulisi olla proaktiivinen ja ohjata asiakasta oikeaan suuntaan. Tämä tarkoittaa muun muassa kokonaisvaltaisen näkemyksen tuomista osaksi projektia, minkä pohjalta asiakas pääsee tehokkaammin liikkelle. Ohjaaminen on tiedon jakamista.

> **SUOSITUS 30:** Testauksen tulisi olla tiivistä yhteistyötä eri osapuolten kesken, joten kommunikoinnin tulisi toimia aktiivisesti kaikkiin suuntiin. Tämä edellyttää toimittajan puolelta ”läsnäoloa”, jotta asiakas voi tuntea tilanteen turvatuksi.

7.8 Testauksen eri osa-alueiden hyvät käytännöt

Miten eri järjestelmäkerroksia tulisi testata ja mitä eri testausvaiheissa tulisi ottaa huomioon? Seitsemännessä tutkimuskysymyksessä keskityttiin eri järjestelmäkerroksilla sovellettaviin testausmenetelmiin ja eri testausvaiheiden läpikäyntiin. Suurin osa teoreettisesta lähdeaineistosta keskittyi vastaamaan juuri tähän kysymykseen. Toisaalta empiiristen tulosten saaminen oli hankalaa, eikä haastatteluiden avulla pystytty muodostamaan kokonaisuutta, joka tukisi teoreettisia tuloksia.

Teoreettisten tulosten mukaisesti ETL-kerroksen testauksessa tulisi keskittyä transformaatioiden eli muunnosprosessien testaamiseen sekä tiedon validointiin. Myös tietohäviöiden ja järjestelmän palautettavuuden testaaminen oli olennainen osa-alue, johon tulisi kiinnittää huomiota. Tiedon varastointi- ja raportointikerroksilla testaus on suu-

rimmaksi osaksi tiedon validointia ja arvojen verifiointia eri järjestelmäkerrosten välillä. Vastaavasti raporttien testauksessa korostuvat layoutin eli ulkoasun validoiminen, toiminnallisuuksien testaaminen sekä tiedon validointi. Yksityiskohtaisemmat suositukset esitetään liitteessä 2.

8 PÄÄTELMÄT

8.1 Tutkimuksen johtopäätökset

Tämän tutkimuksen tarkoituksena oli selvittää, miten tietovarasto- ja raportointijärjestelmiä tulisi testata ja mitä testauksessa tulisi ottaa huomioon. Tietovarasto- ja raportointijärjestelmien testauksesta on vielä toistaiseksi vain vähän teoreettista lähdeaineistoa saatavilla, mistä voidaan päätellä, ettei testausta tunnisteta vielä kovin hyvin osaksi tietovarasto- ja raportointijärjestelmäprojekteja. Toimiala on saavuttanut jo kypsyysvaiheen, eikä kyseisiin järjestelmiin liittyvä teknologia sinänsä ole uutta. Testauksen saama vähäinen huomio kuitenkin kasvaa jatkuvasti, kun toimialalla pyritään löytämään keinoja järjestelmien virheistä tai puutteista aiheutuvien kustannusten ja resurssitarpeiden minimoimiseksi. Tutkimuksesta tuli selväksi se, että kyseessä on todellinen, aito ongelma, johon ei ole kehitetty yhtenäistä ratkaisua.

Tässä tutkimuksessa tarkasteltiin tietovarasto- ja raportointijärjestelmien testauksen problematiikkaa vastaamalla tutkimusongelmasta johdettuihin tutkimuskysymyksiin. Koska tutkimusongelma on laaja-alainen ja moniulotteinen, tutkimuksen tavoitteena oli muodostaa tarkistuslista, joka sisältää karkean tason toimenpidesuosituksia. Hyvien käytäntöjen tarkasteleminen edellytti useiden eri osa-alueiden läpikäyntiä ja mallintamista. Luvussa 7 esitetyt tutkimuksen tulokset ja koostavat suositukset laadittiin sekä teoreettisen kirjallisuustutkimuksen että empiirisen haastattelututkimuksen pohjalta. Nämä suositukset muodostavat laaja-alaiset ja hyvät lähtökohdat tietovarasto- ja raportointijärjestelmien testaamiseksi, ja ne tarjoavat ohjeistuksia niin projektin johdolle kuin liiketoiminnan testaajillekin. Tämä tutkimus jakaa kokonaisuuden pienempiin, helpommin hallittaviin osiin tai teemoihin, jotka tuloksineen tarjoavat selkeät lähtökohdat muun muassa mahdollisille jatkotutkimuksille.

8.2 Tutkimuksen arviointi

Tutkimuksen onnistumista arvioidessa tyypillinen lähtökohta on tarkastella tutkimukselle asetettujen tavoitteiden täyttymistä. Tutkimuksen onnistumista puoltaa se, että tutkimuksen kokonaiskuvan avulla laadituissa johtopäätöksissä vastattiin kaikkiin tutkimuskysymyksiin ja luotiin eräs vaihtoehtoinen ratkaisu käsiteltävään tutkimusongelmaan. Tutkimuksen luonteesta ja tutkimuksen aiheen laajuudesta johtuen tuloksia oli käsiteltävä hyvin yleisellä tasolla, eikä tutkimusongelmaa voitu lähestyä kovin konkreettisesta näkökulmasta tämän tutkimuksen laajuudessa. Tämän vuoksi tutkimuksen antamat suo-

situkset ovat melko yleisellä tasolla kuvattuja. Kohdeyrityksen näkökulmasta voidaan kuitenkin tunnistaa tutkimusongelmaan liittyvät kokonaisuudet ja kiinnittää niihin enemmän huomiota käytännön työssä. Tulokset voivat myös toimia lähtökohtana suunniteltaessa kohdeyrityksen sisäisiä, yhtenäisiä tietovarasto- ja raportointijärjestelmien testauskäytäntöjä. Näin ollen tutkimusta voidaan pitää onnistuneena myös kohdeyrityksen näkökulmasta, sillä siinä tunnistettiin testaukseen liittyvät keskeiset ongelmat ja luotiin kehykset eri osa-alueiden hyville käytännöille. Onnistumista tukee myös tutkimuksen tulosten suhteuttaminen tutkimuksen lähtökohtiin. Tutkittua tietoa aiheesta on tarjolla vähän, eikä kohdeyrityksellä ole yhdenmukaista ohjeistusta tai tietämystä tietovarasto- ja raportointijärjestelmien testaukseen liittyen.

Tieteellisesti tutkimuksen onnistumista voidaan arvioida tutkimuksen reliabiliteetin ja validiteetin avulla (Olkkonen 1994, s. 38-39). Reliabiliteetti kuvaa tutkimuksen luotettavuutta toistettavuuden näkökulmasta. Toistettavuudella tarkoitetaan käytännössä sitä, kuinka tarkasti kyseinen tutkimus voidaan toistaa hyödyntäen samoja tutkimusmenetelmiä. Tutkimuksen reliabiliteettia voidaan pitää korkeana, mikäli se samoin tutkimusmenetelmin toistettaessa tuottaa samat tulokset. Validiteetti vastaavasti määrittää mittaako tutkimus todella sitä, mitä sen oli tarkoitus mitata tai kuinka todenmukaisia tutkimuksen tulokset todella ovat. (Golafshani 2003; Koskinen et al. 2005, s. 254-256) Toisin sanoen, validiteetti ilmaisee, kuinka hyvin käytetyt menetelmät soveltuvat tutkimuksen kohteena olevan ilmiön tutkimiseen. Perinteisesti sekä reliabiliteetin ja validiteetin käsite liitetään kvantitatiiviseen tutkimukseen ja niiden hyödyntämistä kvalitatiivisissa tutkimuksissa usein kritisoidaan, sillä kvalitatiivisten tutkimusten tulokset ovat tyypillisesti subjektiivisia ja vaikeasti mitattavia (Golafshani 2003). Tästä johtuen perinteisiä reliabiliteetin ja validiteetin käsitteitä tulisi soveltaa kvalitatiivisen tutkimuksen osalta. Esimerkiksi Koskinen et al. (2005, s. 257) esittää tutkimuksen validiteetin arvioimiseksi luotettavuutta ja siirrettävyyttä sekä reliabiliteetin arvioimiseksi riippuvuutta.

Tämän tutkimuksen osalta korkeaa reliabiliteettia tukee erityisesti tutkimusmenetelmien sekä haastatteluiden rakenteen yksityiskohtainen kuvaaminen tieteellisiä käsitteitä noudattaen, mikä mahdollistaa tutkimuksen toistettavuuden tarkasti samoja periaatteita noudattaen. Toisaalta haastattelumenetelmäksi valittu teemahaastattelu voi vaikuttaa reliabiliteettiin negatiivisesti, sillä sen puolistrukturoitu rakenne voi johtaa erilaisiin tulkintoihin niin haastattelijan kuin haastateltavankin näkökulmasta. Toisaalta valitut teemat rajaavat keskusteluaiheet hyvin, jolloin suurella todennäköisyydellä samat asiat nousevat esiin tutkimuksen toteutuskerrasta riippumatta. Teemahaastatteluiden yhteydessä tutkimuksen reliabiliteettia pyrittiin kasvattamaan yhdenmukaistamalla haastateltavien tulkintoja tarkentamalla haastatteluissa käytettyjä käsitteitä tarvittaessa. Kaiken kaikkiaan, tämän tutkimuksen reliabiliteetin voidaan katsoa olevan riittävällä tasolla, koska tutkimusmenetelmät on kuvattu riittävän yksityiskohtaisella tasolla. Tämä mahdollistaa tutkimuksen uudelleen toteuttamisen, jolloin päästään suurella todennäköisyydellä samoihin tuloksiin. Reliabiliteettia tämän tutkimuksen osalta lisää myös aiheesta

kirjoitetun, saatavilla olevan lähdeaineiston vähäinen määrä. Toisaalta haastattelukysymysten yleismaailmallinen näkökulma, jossa ei mennä liialti yksityiskohtiin, heikentää reliabiliteettia.

Tämän tutkimuksen validiteettia vastaavasti heikentää tutkimustulosten subjektiivisuus. Haastateltavien taustat olivat erilaisia, mikä vaikutti heidän näkemyksiinsä ja lopulta vastauksiin haastattelutilanteissa. Erityisesti tutkijan omalla näkemyksellä, kokemuksella ja asiantuntemuksella aihepiiriin liittyen oli verrattain suuri merkitys vastausten tulkinnassa, tulosten analysoinnissa sekä yhteenvedossa. Kysymys onkin siitä, että se mikä jonkun henkilön mielestä on olennaista, voi toisen henkilön mielestä olla epäolennaista. Laaja-alaisempi käytännön kokemus muun muassa tietovarasto- tai raportointijärjestelmäprojektien toteutuksesta olisi todennäköisesti auttanut tutkijaa rajaamaan haastattelukysymyksiä paremmin ja tulkitsemaan sekä analysoimaan tuloksia todenmukaisemmin. Toisaalta tulokset voidaan kyseenalaistaa sen vuoksi, että haastatteluihin osallistui lopulta vain kahdeksan henkilöä, joista vain kaksi oli varsinaisen kohdeyrityksen ulkopuolelta valittuja. Tämän tutkimuksen puitteissa ei voida myöskään olla varmoja siitä, oltaisiinko muita menetelmiä hyödyntämällä saatu aikaiseksi parempia ja todenmukaisempia tuloksia. Lisäksi validiteettia on vaikea arvioida senkin vuoksi, että aikaisempia tutkimuksia kyseisestä aiheesta on kirjoitettu hyvin vähän, eikä yrityksissä vielä tunnisteta kunnolla tutkimuksen kohteeksi valittuun ilmiöön liittyviä tekijöitä. Näin ollen tutkimuksen validiteetin arvioiminen voi olla hankalaa ja arviot saattavat vaihdella henkilöstä toiseen. Tämän johdosta tuloksia ei voida suoranaisesti yleistää, mutta niiden avulla voidaan varmasti tunnistaa sellaisia kokonaisuuksia, joihin yritysten kannattaa kiinnittää huomiota kehittäessään tietovarastointi- ja raportointiratkaisuja.

Tutkimuksen reliabiliteettia ja validiteettia olisivat lisänneet tutkijan henkilökohtainen osaaminen ja käytännön kokemukset tietovarasto- ja raportointijärjestelmien kehityksestä ja testauksesta. Lisäksi tutkimuksessa olisi voitu haastatella useampia henkilöitä eri organisaatioista, jotka toimivat eri toimialoilla. Käytettävissä olleiden resurssien niukkuuden vuoksi tällaisten vaihtoehtojen toteuttaminen oli kuitenkin mahdotonta, mikä puoltaa tietyssä mielessä tutkimuksen onnistumista. Yhteenvetona voidaan todeta, että tutkimus onnistui niiden resurssien puitteissa, mitä tutkijalla oli käytettävissään. Tutkimuksessa pystyttiin siis nostamaan esiin selkeä näkemys tutkittavaan ongelmaan ja tunnistamaan yleisellä tasolla siihen liittyviä tekijöitä. Kaiken kaikkiaan tulokset antavat hyvän pohjan jatkotutkimuksille tai lisäselvityksille.

Tutkimuksen reliabiliteetin ja validiteetin arvioinnin lisäksi esiin voidaan nostaa myös teoriaosuuden vähäinen lähdemateriaalin käyttö. Tämä johtuu käytännössä saatavilla olevan lähdemateriaalin vähäisyydestä. Hyvistä käytännöistä on olemassa erittäin vähän tutkittua tietoa tai ylipäätään kirjallisuutta. Lisäksi valtaosa tällaisesta materiaalista on yritysten markkinointimateriaalia, joiden käyttöä tieteellisessä tutkimuksessa tulisi ensisijaisesti välttää. Tämän vuoksi teoriaosuudessa ei voitu pureutua tutkimusongelmaan

riittävän syvällisesti. Se nostaa kuitenkin esiin hyviä tarkistuslistoja eri testausvaiheiden suorittamiseksi ja toisaalta luo pohjan empiiristen tulosten analysoinnille.

8.3 Jatkotutkimusmahdollisuudet

Tutkimuksen aihepiiristä, tietovarasto- ja raportointijärjestelmien testauksesta, on vain vähän tutkittua tietoa tieteellisessä kirjallisuudessa, eikä esimerkiksi kirjallisuutta ole ollut saatavilla kuin vasta viime vuosina. Toisaalta tieteellisten artikkeleiden, julkaisujen, yritysten markkinointimateriaalien määrä on kuitenkin ollut selvästi kasvussa viime vuosina. Tämä kertoo todennäköisesti siitä, että aihepiiri on saavuttanut yhä suurempaa huomiota maailmalla. Tutkimuksen edetessä oli selvästi havaittavissa, että esimerkiksi yrityksissä ei välttämättä vielä tunnisteta tietovarasto- ja raportointijärjestelmien testausta osaksi kyseisiä järjestelmäprojekteja. Toki tietovarasto- ja raportointijärjestelmät poikkeavat luonteeltaan perinteisistä operatiivisista tietojärjestelmistä, mutta yhtälailla, ne ovat ohjelmistoprojekteja, joihin pätevät käytännössä samat lainalaisuudet - testaus mukaanlukien. Vaikka tietovarasto- ja raportointijärjestelmät ovatkin jo arkipäivää yritysten toiminnassa, on lisätutkimukselle tarvetta tämän aihepiirin osalta. Tämä tutkimus antaa hyvät lähtökohdat tutkia aihepiiriä syvällisemmin ja näin hankkia lisää tietoutta aihepiirin eri osa-alueista.

Vastaavanlaisten tutkimusten tekeminen on perusteltua, sillä hyvien käytänteiden käsitteleminen edellyttää lähtökohtaisesti tätä tutkimusta suurempaa otosta. Näin tuloksista saadaan todenmukaisempia ja tarkempia. Jatkotutkimusta voitaisiin tehdä käytännössä mistä tahansa tämän aihepiirin osa-alueesta keskittymällä esimerkiksi johonkin testausprosessin vaiheeseen tai tietyn testausvaiheen syvällisempään tarkasteluun. Tässä tutkimuksessa pyrittiin tarkastelemaan aihepiiriä melko laajasti, mutta jatkotutkimukset kannattaa rajata tarkemmin. Mahdollista jatkotutkimusta voidaan lähestyä joko teknisestä tai hallinnollisesta näkökulmasta aina tarpeiden mukaan. Tekninen näkökulma antaisi kehittäjille ja testaajille konkreettisempia ohjeistuksia ja työkaluja päivittäisten rutiinien tueksi ja toisaalta hallinnollinen näkökulma auttaisi muun muassa projektipäälliköitä kokonaisuuksien hallinnassa. Lisäksi eräs tutkimuksen aikana esiin noussut konkreettinen jatkotutkimusaihe on hyvien käytäntöjen mukainen yleistestaussuunnitelman pohja, joka olisi nopeasti muokattavissa eri organisaatioiden tarpeisiin. Tämä tutkimus antaa hyvät lähtökohdat myös tällaiselle jatkotutkimusvaihtoehdolle.

LÄHTEET

Ariyachandra, T., Watson, H. 2006. Which Data Warehouse Architecture is Most Successful. Business Intelligence Journal. Vol. 11, No. 1. 4-6 ss.

Ariyachandra, T., Watson, H. 2010. Key Organizational Factors in Data Warehouse Architecture Selection. Decision Support Systems. Elsevier B.V. 200-212 pp.

Bhat, S. R. 2007. Data Warehouse Testing: Practical Approach. Stickyminds.

Black, R. 2009. Managing the Testing Process: Practical Tools and Techniques for Managing Hardware and Software Testing. Third Edition. Wiley Publishing, Inc. 638 s.

Bocarsly, J. R. 2011. Complex ETL Testing: A Strategic Approach. Real-Time Technology Solutions. New York.

Brahmkshatriya, K. 2007. Data Warehouse Testing. Stickyminds.

Chaudhuri, S., Dayal, U. 1997. An Overview of Data Warehousing and OLAP Technology. ACM Sigmod Record. 517-526 ss.

Cobb, C. G. 2011. Making Sense of Agile Project Management: Balancing Control and Agility. John Wiley & Sons, Inc.

DeLone, W.H., McLean E.R. 1992. Information Systems Success: The Quest for the Dependent Variable. Information Systems Research. Vol. 3, No. 1. 60-95 ss.

El Gamal, N., ElBastawissy, A., Galal-Edeen, G. 2011. Towards a Data Warehouse Testing Framework. Ninth International Conference on ICT and Knowledge Engineering. 65-71 ss.

English, L. P. 1999. Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits. John Wiley & Sons, Inc.

Gardner, S. 1998. Building the Data Warehouse. Communications of the ACM. Vol. 41, No. 9. 52-60 ss.

Geiger, J., Inmon, W.H., Zachman, J. 1997. Data Warehousing and the Zachman Framework: Managing Enterprise Knowledge. McGraw-Hill.

Golafshani, N. 2003. Understanding Reliability and Validity in Qualitative Research. The Qualitative Report. Vol. 8, No. 4. 597-607 ss.

Golfarelli, M., Rizzi, S. 2009. A Comprehensive Approach to Data Warehouse Testing. DOLAP'09, Hong Kong, China. ACM.
Golfarelli, M., Rizzi, S. 2011. Data Warehouse Testing: A Prototype-based Methodology. Information and Software Technology. 1183-1198 ss.

Gupta, S. L., Pahwa, P., Mathur, S. 2012. Classification of Data Warehouse Testing Approaches. International Journal of Computers & Technology. Vol. 3, No. 3. 381-386 ss.

Haikala, I., Märijärvi, J. 2006. Ohjelmistotuotanto. Yhdestoista painos. Talentum Media Oy. 440 s.

Harding, J. A., Yu, B. 1999. Information-centred enterprise design supported by a factory data model and data warehousing. Computers In Industry. Elsevier Science B.V. 23-36 ss.

Hetzel, W. 1988. The Complete Guide to Software Testing. Second Edition. John Wiley & Sons, Inc. 283 s.

Hirsjärvi, S., Hurme, H. 2001. Tutkimushaastattelu - Teemahaastattelun teoria ja käytäntö. Yliopistopaino, Helsinki. 213 s.

Hirsjärvi, S., Hurme, H. 2008. Tutkimushaastattelu: Teemahaastattelun teoria ja käytäntö. Gaudeamus Helsinki University Press, Helsinki. 213 s.

Hovi, A., Koistinen, H., Ylinen, J. 2001. Tietovarastot liiketoiminnan tukena. Talentum Media Oy. Gummerus Kirjapaino Oy, Jyväskylä.

Inmon, W. H. 2002. Building the Data Warehouse, Third Edition. John Wiley & Sons, Inc.

ISTQB. 2012. Standard Glossary of Terms Used in Software Testing Version 2.2. Glossary Working Party of International Software Testing Qualification Board.

Itkonen, J., Kauppinen, M. 2011. Vaatimusten ja testauksen integrointi? Vuoropuhelu nykytilasta ja tulevaisuudesta. Aalto-yliopisto. SoberIT.

Kasanen, E., Lukka, K., Siitonen, A. 1991. Konstruktiivinen tutkimusote liiketaloustieteessä. Liiketaloudellinen aikakauskirja. 301-327 s.

Kamal, R., Nakul, M. 2013. Adventures with Testing BI/DW Application: On a Crusade to Find the Holy Grail. Microsoft. [WWW]. Viitattu 9.6.2013. Saatavilla: http://msdn.microsoft.com/en-us/library/gg248101.aspx

Kelley, C. 2004. Data Warehousing and Data Marts. The Internet Encyclopedia. John Wiley & Sons, Inc.

Khan, A. 2012. Business Intelligence & Data Warehousing Simplified: 500 Questions, Answers, and Tips. Mercury Learning and Information. 296 s.

Kimball, R., Ross, M. 2002. The Data Warehouse Toolkit, Second Edition. Wiley Publishing, Inc.

Kimball, R., Ross, M., Thornthwaite, W., Mundy, J., Becker, B. 2008. The Data Warehouse Lifecycle Toolkit, Second Edition. Wiley Publishing, Inc.

Koskinen, I., Alasuutari, P., Peltonen, T. 2005. Laadulliset menetelmät kauppatieteissä. Tampere. Osuuskunta Vastapaino. 350 s.

Leffingwell, D., Widrig, D. 2000. Managing Software Requirements: a Unified Approach. Addison-Wesley. 491 s.

Mathen, M. P. 2010. Data Warehouse Testing. DeveloperIQ Magazine. Infosys, Ltd.

Mookerjea, A., Malisetty, P. 2008. Data Warehouse / ETL Testing: Best Practices. Pure Conference Solution Ltd.

Myers, G. J., Sandler, C., Badgett, T., Thomas, T. M. 2004. The Art of Software Testing. John Wiley & Sons, Inc. 256 s.

Neilimo, K., Näsi, J. 1980. Nomoteettinen tutkimusote ja suomalainen yrityksen taloustiede. Tutkimus positivismin soveltamisesta. Tampere, Tampereen yliopisto, Yrityksen taloustieteen ja yksityisoikeuden laitoksen julkaisuja, sarja A2. 145 s.

Olkkonen, T. 1994. Johdatus teollisuustalouden tutkimustyöhön. Toinen painos. Espoo, Teknillinen korkeakoulu, Tuotantotalouden osasto, Teollisuustalouden laboratorio. Raportti 152/1993/Teta. 143 s.

Ponniah, P. 2010. Data Warehousing Fundamentals for IT Professionals, Second Edition. John Wiley & Sons, Inc.

Preece, J., Rogers, Y., Sharp, H. 2002. Interaction Design: Beyond Human-Computer Interaction. John Wiley & Sons, Inc. 544 s.

Pyhäjärvi, M., Pöyhönen, E. 2005. Testaus ohjelmistokehityksen osana. Julkaisuversio 1.0.

Rainardi, V. 2008. Building a Data Warehouse: With Examples in SQL Server. Apress. 523 s.

Rongo. 2012. Tietovarastointi. [WWW]. Viitattu 23.1.2013. Saatavilla: http:// http://rongo.fi/palvelut/tietovarastointi

Salmi, T., Järvenpää, M. 2000. Laskentatoimen case-tutkimus ja nomoteettinen tutkimusajattelu sulassa sovussa. Tiivistelmä. [WWW]. Viitattu 10.2.2012. Saatavilla: http://lipas.uwasa.fi/~ts/sopu/sopu.html

Schaefer, H. 2008. What a Tester Should Know, At Any Time, Even After Midnight. Testing Experience: The Magazine for Professional Testers. No. 1. 40-46 ss.

Srivastava, V. R. 2011. Testing Trends in Data Warehouse. ISQT. Testing Services, Mind Tree Limited.

Tanuska, P., Moravcik, O., Vazan, P., Miksa, F. 2009. The Proposal of Data Warehouse Testing Activities. Proceedings of the 20th Central European Conference on Information and Intelligent Systems. Faculty of Organization and Informatics, Varazdin, Croatia.

Theobald, J. 2007. Strategies for Testing Data Warehouse Applications. Information Management. [WWW]. Viitattu 25.1.2013. Saatavilla: http://www.information-management.com/issues/20070601/1086005-1.html

Tuomi, I. 1999. Data is More Than Knowledge: Implications of the Reversed Knowledge Hierarchy for Knowledge Management and Organizational Memory. Journal of Management Information Systems. Vol 16, No. 3. 107-121 ss.

van Veenendaal, E., Pol, M. 1997. A Test Management Approach for Structured Testing. Achieving Software Product Quality. UTN Publishers.

Velicanu, M., Matei, G. 2007. Building a Data Warehouse Step by Step. Informatica Economica, Vol. 42, No. 2.

Vucevic, D., Zhang, M. J. 2011. Testing Data Warehouse Applications. Trafford Publishing. 189 s.

Vuorela, S. 2005. Haastattelumenetelmät. Teoksessa Ovaska, S., Aula, A., Majaranta, P. 2005. Käytettävyystutkimuksen menetelmät. Tampereen yliopisto. Tietojenkäsittelytieteiden laitos.

Watson, H. J., Goodhue, D. L., Wixom, B. H. 2002. The Benefits of Data Warehousing: Why Some Organizations Realize Exceptional Payoffs. Information & Management. Elsevier Science B.V. 491-502 ss.

Yaddow, W. 2011. Data Warehouse Master Test Plan. Viitattu 13.3.2013. Saatavilla: http://www.slideshare.net/wyaddow/data-warehouse-master-test-plan

Yaddow, W. 2009. QA Quidelines for Data Warehouse Quality Verification. Viitattu 17.3.2013. http://www.slideshare.net/wyaddow/etl-and-data-test-guidelines-for-large-applications

LIITE 1: HAASTATTELUKYSYMYKSET

Alkutoimet

- Äänityslupa
- Tutkimuksen ja aiheen alustus
- Haastateltavan kokemus

DW/BI-järjestelmien erityispiirteet

- Miten tietovarasto- ja raportointijärjestelmät eroavat mielestäsi muista tietojärjestelmistä testauksen näkökulmasta eli mitä erityispiirteitä niihin liittyy tietovarasto- ja raportointijärjestelmien testaukseen?

DW/BI-järjestelmien testauksen menestystekijät

Muistele mielestäsi testauksen näkökulmasta onnistuneita ja epäonnistuneita projekteja. *Onnistumisen määrittäminen.*

- Mitkä tekijät johtavat mielestäsi tietovarasto- ja raportointijärjestelmien testauksen epäonnistumiseen? Millaisia ominaispiirteitä mielestäsi epäonnistunut testaus on pitänyt sisällään?

- Entä mitkä ovat mielestäsi testauksen keskeisiä menestystekijöitä? Millaisia ominaispiirteitä mielestäsi onnistunut testaus on pitänyt sisällään?

DW/BI-järjestelmien testausprosessi

- Kuvaile mielestäsi onnistuneen testausprosessin keskeisiä tekijöitä testausprosessin eri vaiheiden näkökulmasta. Mitä eri vaiheissa tulisi ottaa huomioon?

    - Alkuvalmistelut
    - Testaussuunnitelman laatiminen ja hyväksymiskriteerien määrittäminen
    - Testitapausten laadinta
    - Testausympäristön pystyttäminen ja testidatan hankkiminen
    - Testitapausten suorittaminen
        - Millä tavoin virheen arviointiin ja korjaamiseen tähtäävät toimenpiteet tulisi organisoida, jos testauksen aikana havaitaan potentiaalinen virhe? Kuvaile prosessia.

- Mitä testauksen aikana tulisi vähintään dokumentoida?
- Tunnistatko joitain sellaisia toistuvia ja haitallisia virheitä tai puutteita, jotka on havaittu tyypillisesti vasta tuotannossa?

DW/BI-järjestelmien testauksen haasteet

- Mitkä asiat olet kokenut omakohtaisesti tai havainnut haasteelliseksi testauksen yhteydessä?

- Millaisia haasteita resurssien niukkuus tyypillisesti asettaa testauksen näkökulmasta? Miten toimittaja voi auttaa näiden haasteiden ylittämisessä?

DW/BI-järjestelmien testauksen osa-alueet

- Mitä kokonaisuuksia ja järjestelmäkerroksia mielestäsi tulisi erityisesti testata? Miksi? *Järjestelmäkerrosten määrittäminen. Esimerkki kokonaisuudesta.*

- Mihin testauksen osa-alueisiin toimittajan tulisi ensisijaisesti keskittyä ja mitkä näet olevan asiakkaan omalla vastuulla? *Testauksen osa-alueiden määrittäminen*

- Mitkä asiat on erityisesti otettava huomioon mainitsemillasi testauksen osa-alueilla eli mitä hyviä käytäntöjä niihin liittyy? *Vertaa aikaisempia kokemuksiasi ja pohdi asioita yleisellä tasolla.*

- Tarkastellaan asiakkaan vastuulla olevia testauksen osa-alueita: mitä ongelmia näihin liittyy ja miten ongelmat voidaan ratkaista?

- Miten toimittajan tulisi (osata) tukea tilaajaa testauksen läpiviennissä?

- Tulisiko testausta automatisoida jollain testauksen osa-alueella? Millä osa-alueella ja miten?

- Tulisiko mielestäsi tiedon luottamuksellisuus ottaa huomioon testauksessa ja miten?

LIITE 2: TESTAUKSEN ERI OSA-ALUEIDEN HYVÄT KÄYTÄNNÖT

ETL-kerroksen testaus	• Tietovarastoon ladatun datan oikeellisuus tulisi validoida järjestelmällisesti (esimerkiksi näytteenottomenetelmää hyödyntäen). • Tiedon validoinnissa tulisi varmistaa myös, ettei tietovarastoon ole päässyt syntymään duplikaatteja eli identtisiä tietueita ja että datan eheys säilyy aina muunnosprosesseista ja latausten toimintaperiaatteista huolimatta. • Latauslogien ja virheviestien populoituminen tulisi verifioida latausalueella. • Surrogaattiavainten oikeellisuus tulisi tarkistaa sen varalta, että ne ovat todella yksilöiviä ja luonteeltaan tunnisteeksi soveltuvia. • Tietotyyppien ja avainmäärittelyiden oikeellisuus ja johdonmukaisuus tulisi verifioida. • Testauksessa tulisi varmistaa, että transformaatiot toimivat asianmukaisesti ja noudattavat liiketoimintasääntöjä. Myös kertaluonteiset erityistapaukset tulisi käsitellä. • Kaikilla ETL-töillä tulisi olla pääsy tarvittavaan lähdetietoon ja kohdetietokantoihin. • Lähdetieto tulisi poimia oikeasta lähteestä, mikä on huomioitava silloin, kun sama tieto esiintyy useammassa lähteessä. • Virheellisten tai hylättyjen tietueiden käsittelyt tulisi validoida. • Jokaista testiskenaariota tulisi testata epäkelvoilla syötteillä (negatiivinen testaus). • ETL-prosessit tulisi testata tietohäviöiden varalta. • Tietovuotojen testaamisen tulisi olla jatkuvaa. • Ennen palautettavuuden testausta on määritettävä tapahtumatyypit, joilta halutaan erityisesti suojautua. Jokaisen tapahtuma- ja prosessityypin välinen kombinaatio tulisi simuloida, minkä jälkeen kyseisen prosessityypin sisältävä ETL-prosessi tulisi ajaa uudelleen. (Bhat 2007; Brahmkshatriya 2007; Mookerjea & Malisetty 2008; Mathen 2010; Kamal & Nakul 2013; Rainardi 2008, s. 479;Vucevic & Zhang 2011.)
Tietovaraston testaus	• Tietovaraston noudattama tietomalli tulisi validoida (myös) loppukäyttäjien toimesta. • Tietovaraston sisältämä data tulisi validoida lähdetauluja vastaan. • Tietovaraston testauksessa tulisi verifioida, että faktataulujen vierasavaimet on indeksoitu, yhteen faktatauluun liittyy aina yksi tai useampia dimensiotauluja ja että eri hierarkiatasojen sisältämät faktat eivät sisällä NULL-arvoja. (Tanuska et al. 2009)

Raporttien testaus	• Raportin layout tulisi vastata määrityksiä ja kuvata raportilta vaadittuja asioita oikein, mikä tarkoittaa esimerkiksi sitä, että raportti sisältää odotetunlaiset visuaaliset elementit, otsikkotiedot, suodattimet, kehotteet , attribuutit ja mittaristot. • Tiedon formaatit tulisi tarkistaa raportilla (stilisointi, null-arvot, prosenttiluvut, otsikkotiedot). • Raportin toiminallisuudet (porautuminen, järjestäminen, tuonti- ja vientioperaatiot) tulisi testata ja ulkoasun oikeellisuus tarkastaa tässä yhteydessä. • Raportilla esittävä tieto tulisi validoida huomioiden myös tarkkuus ja tarkoituksenmukaisuus. • Raportilla esiintyvien lähdetiedoista johdettujen mittareiden tai summatietojen taustalla olevat logiikat tulisi verifioida. (Kamal & Nakul 2013; Srivastava 2011; Yaddow 2009.)
Integraatiotestaus	• Integraatiotestauksessa tulisi varmistaa, että laitteisto on asennettu ja konfiguroitu asianmukaisesti, tarvittavat yhteydet eri komponenttien välillä on konfiguroitu asianmukaisesti, ohjelmistot on siirretty testiympäristöön sekä laitteiston, tietoverkon ja ohjelmistojen konfiguraatiot on dokumentoitu. • Integraatiotestauksessa tulisi hyödyntää tuotantodataa. • Integraatiotestauksessa tulisi varmistaa, että työjonot suoritetaan oikeassa järjestyksessä ja oikea tieto populoituu kohdetauluihin oikeassa muodossa asetetussa aikaikkunassa. • Integraatiotestauksessa tulisi suorittaa end-to-end-tyyppinen testi, jossa koko prosessi ajetaan alusta loppuun aina lähdejärjestelmistä raporteille. • Ajastettuja ETL-töitä tulisi ajaa muutamien päivien ajan, jotta varmistutaan, että tarpeellinen tieto populoituu tietovarastoon asianmukaisesti ja tiedon eheys säilyy. (Bhat 2007; Brahmkshatriya 2007; Mathen 2010; Rainardi 2008, s. 487; Tanuska et al. 2009; Theobald 2007; Yaddow 2011, s. 3.)
Hyväksyntätestaus	• Hyväksyntätestauksessa tulisi käyttää ehdottomasti suoraan tuotantodataa. • Hyväksyntätestauksen aikana loppukäyttäjien tulisi kirjata ylös kommentteja, mielipiteitä ja huomioita järjestelmän toiminnallisuudesta, käytettävyydestä ja ulkoasusta sekä koota nämä yhtenäiseksi loppuraportiksi. • Testaajien tulisi yhdessä liiketoiminnan ja käyttäjien kanssa pohtia, millaista dataa hyväksyntätestauksessa tarvitaan ja kuinka usein tämä data tulisi hyväksyntätestauksen aikana päivittää. • Ennen hyväksyntätestauksen aloittamista käyttäjiä tulisi ohjeistaa järjestelmän toiminnallisuuksista ja arkkitehtuurista, jotta he ymmärtäisivät millaisessa ympäristössä he operoivat. Testauksen aikana toimittajan tulisi olla testaajien käytettävissä. (Rainardi 2008, s. 486; Theobald 2007; Vucevic & Zhang 2011, s. 99; Yaddow 2011, s. 4.)
Suorituskykytestaus	• Suorituskykyä testattaessa tulisi ottaa huomioon tiedon sekä proses-

	soinnin ja latausten tehokkuus että hakuoperaatioiden tehokkuus kolmessa pisteessä: lähdejärjestelmien ja latausalueen välillä, latausalueen ja tietovaraston välillä sekä tietovaraston ja esityskerroksen välillä. • Tietovaraston tietokannat tulisi ladata tuotantoa vastaavalla massalla, latausaika tarkistaa ja vertata niitä määritettyihin aikaikkunoihin. • Järjestelmää kuormittavia operaatioita tulisi suorittaa eri datavolyymein ja prosessointiaikoja arvioida skaalautuvuuden kannalta. • Liitos-operaatioita tulisi suorittaa suurilla datavolyymeillä kyselyiden suorituskyvyn mittaamiseksi. • Suorituskykytestauksessa tulisi toimia yhdessä liiketoiminnan kanssa kehittäen tuotantoa vastaavia esimerkkihakuja ja määrittäen suorituskykykriteerejä. (Golfarelli & Rizzi 2009; Mathen 2010; Rainardi 2008, s. 483; Theobald 2007; Vucevic & Zhang 2011; s. 97-98.)
Tietoturvatestaus	• Tietoturvatestauksessa pääsy järjestelmään eri pisteistä tietoverkkoa tulisi perusteellisesti. Tämä tulisi toteuttaa tietoturvapolitiikan ja liiketoiminnan vaatimusten mukaisesti siten, että eri osista tietoverkkoa otetaan yhteys järjestelmään. • Pääsyoikeudet tulisi testata yksinkertaisilla, esivalmistelluilla raporteilla, mikäli valmiita raportteja ei ole saatavilla. Näin voidaan varmistaa, että käyttäjät näkevät raporteilla vain ne tiedot, joihin heillä on oikeus. • Tiedon varmuuskopioinnin ja palautettavuuden testaamista varten tulisi perustaa erillinen strategia, jossa määritellään, miten varastointikerroksen tietojen varmuuskopiointi toteutetaan ja kuinka palautettavuuden tulisi toimia. Olennaista on käyttää tuotantoa vastaavia tietovolyymeja, jotta kyetään näkemään tiedon palautukseen liittyvät vaikutukset todellisuutta vastaavassa tilanteessa. • Tietoturvatestauksessa tulisi johtaa testitapauksia suoraan organisaation tietoturvapolitiikasta. (Rainardi 2008, s. 485-486; Tanuska et al. 2009.)