LAUREA
AMMATTIKORKEAKOULU
Yhdessä enemmän

Sähköisen laskutuksen testitapausten suunnittelu

Mattola, Marjo

Laurea-ammattikorkeakoulu

Sähköisen laskutuksen testitapausten suunnittelu

Mattola Marjo
Tietojenkäsittelyn koulutusohjelma
Opinnäytetyö
Marraskuu, 2017

Laurea-ammattikorkeakoulu **Tiivistelmä**
Tietojenkäsittely
Tradenomi (AMK)

Mattola Marjo

Sähköisen laskutuksen testitapausten suunnittelu

Vuosi 2017 Sivumäärä 59

Tämä opinnäytetyö oli toimeksiantajalle tehtävä tutkimuksellinen kehittämistyö, jonka tarkoituksena oli luoda kattava listaus testitapauksia kohdeyrityksen sähköisen laskutuksen integraatio- ja järjestelmätestausta varten. Tavoitteena oli kuvata testitapaukset tarkasti käytössä olevaan testauksen hallintajärjestelmään, josta niitä voidaan poimia jatkossa myös muihin testaustilanteisiin. Työn lähtökohtana oli kohdeyrityksessä tapahtuva IT-arkkitehtuurin muutos, joka aiheuttaa muutoksia myyntilaskutukseen. Muutoksilla oli merkittävä vaikutus laskusanomaan, joka välitetään sähköisen laskutuksen palvelusta vastaavalle operaattorille. Testitapausten suunnittelulla haluttiin varmistua siitä, että testaus on käytettävissä oleviin resursseihin nähden mahdollisimman tehokasta ja kattavaa. Huonosti toteutettu testaus voi aiheuttaa yritykselle taloudellista vahinkoa tai haittaa maineelle, jos laskuliikenteessä ilmenee häiriöitä muutosten käyttöönoton jälkeen.

Opinnäytetyön teoriaosuudessa kuvataan ensin yleisellä tasolla sähköisen laskutuksen prosessi. Sen jälkeen läpikäydään testauksen merkitys ohjelmistokehitysprojektissa, testauksen menetelmät, testauksen suunnittelu sekä testitapausten laatiminen, näkökulmana integraatio- ja järjestelmätestausvaihe. Tutkimusmenetelmänä hyödynnettiin myös dokumenttianalyysiä sekä osallistuvaa havainnointia. Aineistoa testitapauksien suunnitteluun kerättiin useista eri järjestelmädokumenteista, projektikokouksista ja keskusteluista organisaation ja operaattorin asiantuntijoiden kanssa. Työn tuloksena testausjärjestelmään muodostettiin loogisiin kokonaisuuksiin jaoteltu kattava listaus testitapauksia kuvauksineen. Testitapaukset perustuivat määrittelyihin, tunnettuihin riskeihin sekä kokonaisprosessiin liittyviin käyttötapauksiin. Testitapausten viimeisessä katselmoinnissa testitapaukset hyväksyttiin käytettäväksi kohdeyrityksen integraatio- ja järjestelmätestauksessa. Lisäksi todettiin, että testitapauksia voidaan hyödyntää myös tulevissa testaustarpeissa.

Asiasanat: sähköinen laskutus, testauksen suunnittelu, testitapaus

Laurea University of Applied Sciences **Abstract**
Degree Programme in Business Information Technology
Bachelor's Thesis

Mattola Marjo

Designing Test Cases for Electronic Invoice Sending

Year 2017 Pages 59

This Bachelor's thesis was a functional study, which purpose was to develop the company's testing process by creating comprehensive set of test cases for electronic invoicing integration and system test phase. The objective was to describe the test cases to the test management tool so that the testers have all steps described completely for performing the tests. The aim was that the test cases could also be used in later testing needs. The reason for the testing was that the company is renewing its IT architecture. The renewals also affect to the invoicing process and have an impact on the content of the invoice message from the system, which is forwarded to the service provider responsible for electronic billing. Test planning and test design are the ways to make sure that the testing covers all the requirements. Test cases can be used to test the various functions consistently and efficiently so that different perspectives are taken into account and the same things are not tested in several times and resources are lost. The lack of testing can have a significant impact on the reputation of the company and may also have financial impacts.

The theoretical overview of the thesis describes the basic process of electronic invoicing and software test planning based on publications. The purpose of the overview was to have understanding of using different testing methods and the importance of test case design, so that the result of this thesis would be valid and useful in future testing needs. As a research method, both documentary analysis and participant observation were also used. The materials for designing test cases were collected from several documents, as well as from project meetings and discussions with the organization and service provider's specialists. As a result of this thesis, a comprehensive set of test cases and descriptions of test case steps were formed into logical entities to the test management tool. The test cases were based on specifications, known risks and use cases related to the overall process. After the final review of test cases they were accepted as part of company's integration and system testing and will be used later in another testing project as well.

Keywords: Electronic invoicing; Test planning, Test design, Test case

Sisällys

1 Johdanto

Liiketoiminnan prosessien kehittämisessä pyritään yleensä kustannusten vähentämiseen ja prosessien nopeuttamiseen ja toimenpiteisiin liittyy olennaisena osana liiketoimintaa tukevien ohjelmistojen kehittäminen. Ohjelmistokehityksen aloitusvaiheessa kartoitetaan liiketoiminnan tarpeet ja laaditaan vaatimusmäärittely, jonka pohjalta lähdetään laatimaan ohjelmiston toiminnallisia ja teknisiä määrittelyjä. Samanaikaisesti on tärkeää aloittaa testauksen suunnittelu, jotta mahdolliset virheet ja puutteet havaittaisiin ajoissa ja niiden korjaaminen olisi vielä helpompaa. Testauksen tavoitteena on selvittää ohjelmiston toiminnan laatu ja luotettavuus ja poistaa ohjelmiston toimintaan liittyviä riskejä. Jotta testaus olisi tehokasta käytettäviin resursseihin nähden, tulee sen suunnitteluun kiinnittää erityistä huomiota. Hyvällä suunnittelulla voidaan välttää se, ettei samoja asioita testata turhaan useita kertoja ja testaus kattaisi mahdollisimman hyvin ohjelmiston toiminnallisuudet. Huonosti toteutettu testaus voi aiheuttaa yritykselle taloudellista vahinkoa tai haittaa yrityksen maineelle, jos käyttöönotettu järjestelmä ei toimi kuten on ollut tarkoitus.

Ohjelmistokehityksen eri vaiheissa voidaan tehdä testausta eri menetelmiä hyödyntäen. Erilaisten testaustekniikoiden ymmärtäminen auttaa kattavan ja tehokkaan testauksen suunnittelussa. Valittava testaustekniikka riippuu testauksen kohteesta, sen määrittelyistä sekä testaustasosta. Tekniikat eivät sulje toisiaan pois, useampaa tekniikkaa voidaan käyttää täydentämään testauksen kattavuutta. Tekniikoiden avulla luodaan testitapauksia, jotka kuvaavat toimenpiteitä, jolla varmistetaan ohjelman toiminnallisuus. Käytännössä tulee eteen kuitenkin todellisuus siitä, että ohjelmistoa on mahdoton testata täydellisesti, esimerkiksi mahdollisten syötteiden määrä voi olla usein erittäin suuri tai ohjelmistossa on valittavana paljon eri polkuja, jolloin testitapausten määrä kasvaa mahdottomaksi. Hyvällä testitapausten suunnittelulla pyritäänkin siihen, että testaus voidaan suorittaa käytettävissä olevilla resursseilla tehokkaasti, johdonmukaisesti ja mahdollisimman kattavasti niin, että eri näkökulmat tulevat huomioiduiksi ja samoja asioita ei testata turhaan useaan kertaan.

Opinnäytetyön tarkoituksena oli luoda kattava listaus testitapauksia kohdeyrityksen sähköisen laskutuksen integraatio- ja järjestelmätestausta varten. Tavoitteena oli kuvata testitapaukset tarkasti käytössä olevaan testauksen hallintajärjestelmään, josta niitä voidaan poimia jatkossa muihinkin testaustilanteisiin. Testauksen kohteena olevalla sähköisellä laskutuksella tarkoitetaan laskun tietojen välittämistä vastaanottajalle laskusanoman avulla. Tähän liittyy olennaisesti termi verkkolasku. Sähköiseen laskutukseen voidaan lukea myös paperimuotoisten laskujen ulkoistettu tulostus-, kuoritus- ja postituspalvelu, joka toteutetaan laskusanoman perusteella. Tästä käytetään puhekielessä yleisesti termiä e-kirje. Laskuttajan näkökulmasta kaikki laskutus vois siis olla sähköistä laskutusta, koska vasta laskuaineiston vastaanottavan

operaattorin tehtävä on tulkita laskusanoman sisällöstä toimitetaanko lasku eteenpäin verkkolaskuna vai paperimuodossa. Näin ollen laskun lähettäjän ei tarvitse tehdä laskutuskanavaan perustuvia valintoja. Sähköisen laskutuksen prosessissa merkittävässä roolissa on siis operaattorin palvelu. Opinnäytetyössä suunniteltavissa testitapauksissa keskitytään tähän laskutusprosessin osaan.

Opinnäytetyön teoriaosuudessa kuvataan ensin yleisellä tasolla sähköisen laskutuksen prosessi. Sen jälkeen läpikäydään testauksen merkitys ohjelmistokehitysprojektissa, testauksen menetelmät, testauksen suunnittelu sekä testitapausten laatiminen, näkökulmana integraatio- ja järjestelmätestausvaihe. Tässä opinnäytetyössä ei käsitellä myyntitilausten pohjalta muodostettavan laskun ja laskusanoman muodostamista varsinaisessa laskutusjärjestelmässä. Opinnäytetyöstä rajataan lisäksi pois automaatiotestaus. Opinnäytetyönä kehitetyt testitapaukset arvioitiin kohdeyrityksen toimesta. Testitapaukset hyväksyttiin käytettäväksi integraatio- ja järjestelmätestauksessa ja todettiin, että testitapauksia voidaan hyödyntää myös tulevissa testaustarpeissa.

2 Työn lähtökohdat

Opinnäytetyön toimeksiantajana oli vähittäiskaupan alalla toimiva yritys, jonka tehtävänä on tuottaa mm. hankinta-, asiantuntija- ja tukipalveluita muille ryhmään kuuluville yrityksille. Kohdeyritys vastaa mm. yhteisesti käytössä olevien talouden järjestelmien kehittämisestä ja ylläpidosta. Testauksen kohteena olevaa kohdeyrityksen ylläpitovastuulla olevaa sähköistä laskutuksen palvelua käyttää 205 yritystä ja laskuliikenne käsittää kokonaisuudessaan noin 760.000 myyntilaskua vuodessa. Laskutusasiakkaita on noin 270.000 kpl, joista 60 prosenttia on kuluttaja-asiakkaita ja 40 prosenttia yritysasiakkaita. Laskuista noin 57 prosenttia lähtee asiakkaille verkkolaskuna ja 43 prosenttia laskuista toimitetaan tulostuspalvelun kautta postitukseen ja e-kirjeinä asiakkaille.

Kohdeyrityksessä uudistetaan talouden tietojärjestelmiä osana suurempaa IT-arkkitehtuurin muutosta. Tämä talouden järjestelmin uudistaminen alkoi vuonna 2016 ja käyttöönotto tapahtuu vuonna 2018. Uudistukset koskevat mm. SAP-järjestelmän myyntilaskutusta, jossa parannetaan olemassa olevia toiminnallisuuksia ja käyttöönotetaan uusia ominaisuuksia. Koska laskutusjärjestelmästä muodostuvaan laskusanoman sisältöön tulee muutoksia, vaikuttavat ne laskutusliittymään operaattorille sekä operaattorin toteutukseen, missä laskusta muodostetaan kuva ja välitetään eteenpäin asiakkaille joko verkkolaskuna tai paperilaskuna. Tästä johtuen operaattorin tulee tehdä muutoksia palvelunsa määrittelyihin ja toteutukseen. Operaattorin tehtävä on vastaanottaa SAP-järjestelmän muodostama laskuaineisto liittymällä, muuntaa aineisto verkkolaskuiksi, muodostaa laskusta kuva annettujen määrittelyjen mukaisesti, liittää mahdolliset liitteet laskuihin ja lopuksi reitittää lasku asiakkaalle. Reitti voi olla pape-

rilasku, yrityksen verkkolasku, kuluttajan e-lasku tai suoramaksu sen mukaan mikä reititysmääritys laskuaineistossa on annettu. Mikäli verkkolaskun reititys ei ole mahdollista annettujen tietojen perusteella, operaattori huolehtii, että lasku ohjautuu tulostuspalveluun ja lasku lähtee asiakkaalle vähintään paperimuotoisena.
Sähköisen laskutuksen palveluntarjoajan palveluun tehtävät muutokset tulee testata huolella, jotta laskutusjärjestelmässä otetut uudet ominaisuudet voidaan ottaa käyttöön ja asiakkaiden laskutus toimii jatkossakin sujuvasti ja virheettä. Testauksella varmistutaan siitä, että muutoksella tavoiteltava hyöty toteutuu. Koska talouden järjestelmien uudistamisen aiheuttamat muutokset olivat suuria, päätettiin samassa yhteydessä päivittää operaattorin sähköisen laskutuksen palvelun versio, koska käytössä ollut versio oli jo useamman vuoden vanha. Tästä syystä testauksen merkitys kasvoi entisestään, sillä oli varmistuttava, että uusien ominaisuuksien lisäksi myös vanhat toiminnallisuudet toimivat odotetulla tavalla.

Tässä opinnäytetyössä keskitytään käytettävän operaattorin palvelun integraatio- ja järjestelmätestaukseen. Testauksen tavoite on saada käsitys siitä, pystyykö palvelu käsittelemään laskutusjärjestelmästä tulevan aineiston ja miten palvelu toimii, jos aineistossa on jotain sellaista, mikä aiheuttaa prosessissa häiriön. Lähitulevaisuudessa on odotettavissa, että verkkolaskutus yleistyy myös ulkomaisille asiakkaille, ja se asettaa vaatimuksia laskusanoman sisällölle. SAP-järjestelmän muutosten myötä verkkolaskujen lähetys ulkomaisille asiakkaille tulee mahdolliseksi, kun laskuaineisto sisältää riittävästi eriteltynä arvonlisäveroon liittyvät tiedot. Koska pääosa laskutuksesta tapahtuu kirjanpidollisesti kriittisenä aikana kuukauden vaihteessa, aiheuttaa se operaattorin liittymään ja palveluun merkittävän kuormapiikin. Testauksessa pyritään selvittämään, miten palvelu pystyy käsittelemään suuria laskumääriä tai suuria tiedostokokoja, ja aiheuttavatko ne prosessissa viivettä.

Testauksen tarkoitus on laadunvarmistus, palvelun tulee toimia siten että laskujen vastaanottajille ei aiheudu haittaa. Laskun vastaanottajille näkyviä haittoja voivat olla esimerkiksi tilanteet, jossa laskulla ei ole riittävästi tietoa kirjanpitoa tai maksusuoritusta varten, laskulle ei jää tarpeeksi maksuaikaa tai lasku ei ole mennyt lainkaan perille. Jos palvelu ei toimi odotetulla tavalla, on siitä kohdeyritykselle mainehaitan lisäksi taloudellista haittaa. Laskujen maksusuoritusten viivästyminen heikentää yrityksen kassavirtaa ja tilanteiden selvittely asiakkaiden kanssa aiheuttaa ylimääräistä työtä.

3 Sähköinen laskutus

Sähköisessä laskutuksessa välitetään laskun tietoja järjestelmästä toiseen laskusanomalla. Laskusanoma on yleensä xml-pohjaista. Laskun lähettäjän ja vastaanottajan välissä laskusanoman välittäjinä toimivat operaattorit. Suomessa operaattoreina toimivat esimerkiksi Basware, Tieto ja OpusCapita, myös pankit voivat toimia operaattoreina.

Sähköiseen laskutukseen liittyy olennaisesti termi verkkolasku, joka on laskusanomasta muodostettu laskun kuva ja tietosisältö. Yksinkertaistettuna sen voidaan ajatella olevan paperimuotoisen laskun sähköinen muoto. Kuluttajalle verkkopankkiin lähetettävästä verkkolaskusta käytetään termiä e-lasku. E-lasku sisältää laskun tiedon ja laskun kuvan ja kuluttaja voi automatisoida laskun hyväksynnän verkkopankissaan, tai käydä hyväksymässä laskun maksettavaksi aina kun lasku saapuu. Kuluttaja-asiakkaalle voidaan välittää laskuaineistoa myös suoramaksuna. Se eroaa e-laskusta siinä, että laskusta lähetetään asiakkaalle ennakkotiedote paperilla ja laskun maksusuoritus tehdään asiakkaan tililtä automaattisesti ilman erillistä hyväksyntää. Suoramaksu eroaa e-laskusta lisäksi siinä, että se ei edellytä verkkopankin käyttöä. E-lasku ja suoramaksu eroaa yritysten verkkolaskusta siinä, että kuluttajan tulee tilata e-lasku tai suoramaksu pankista tai verkkopankista, kun taas yrityksille voidaan lähettää verkkolasku ilman erillistä tilausta, kunhan vain vastaanottajan verkkolaskuosoite on tiedossa. (Danske Bank 2017; Tietoyhteiskunnan Kehittämiskeskus ry 2017.)

EDIFACT-laskutus on vanha suuryritysten tarpeisiin kehitetty sähköisen laskusanoman välitystekniikka, mutta se ei ole verkkolaskutusta, koska toimintaperiaate on erilainen. Myös sähköpostiin lähetettävää laskua kutsutaan usein sähköiseksi laskuksi, mutta se ei ole aitoa verkkolaskutusta. (Tietoyhteiskunnan Kehittämiskeskus ry 2017.)

3.1 Laskusanoman välitys

Operaattoreiden tehtävä on reitittää laskusanoma oikealle vastaanottajalle sanoman tiedoissa olevan verkkolaskuosoitteen ja välittäjän tunnuksen perusteella. Välittäjätunnus kertoo vastaanottavan operaattorin, minne lasku tulee välittää. Verkkolaskuosoite kertoo laskun lopullisen vastaanottajan. Verkkolaskuosoite on käytännössä olla OVT-tunnus tai tilinumerosta muodostettu IBAN-tunnus. OVT-tunnus muodostuu Suomen verohallinnon maatunnuksesta (0037) ja y-tunnuksesta ilman väliviivaa. Osoitteen perään voidaan lisätä vielä tunnisteita, jos vastaanottava organisaatio haluaa erotella tarkemmin vastaanotettavat laskut. (Tietoyhteiskunnan Kehittämiskeskus ry 2017.)

Verkkolaskuja lähettävien ja vastaanottavien yritysten verkkolaskuosoitteita ylläpidetään Tietoyhteiskunnan Kehittämiskeskus ry:n (TIEKE) Verkkolaskuosoitteistossa. Tietojen päivittämisestä rekisteriin vastaavat operaattorit. Osoitteiston heikkoudeksi on kuitenkin osoittautunut tietojen ajantasaisuus, mikä aiheuttaa verkkolaskujen välittämistä osoitteiseen, jotka eivät ole enää voimassa.

Operaattorin laskusanoman käsittelyprosessi sisältää useita vaiheita, jotka on kuvattu kuviossa 1. Kun operaattori vastaanottaa laskusanoman, se validoi eli analysoi sanoman sisällön kelvollisuuden. Sanomasta tarkastetaan, että aineisto sisältää pakolliset tiedot, ja että ne ovat oikeassa muodossa. Operaattori tutkii laskusanomasta vastaanottajan tiedot ja selvittää,

missä muodossa lasku pitää välittää eteenpäin. Operaattorin tehtävä on muuntaa laskusanoma vastaanottajan käyttämään sanomamuotoon sekä muodostaa sanomasta lisäksi laskun kuva. Verkkolaskusanoma ja laskun kuva välitetään vastaanottajan tarvitsemassa muodossa suoraan vastaanottavaan taloushallinnon järjestelmään tai verkkopankkiin. Suomessa yleisimmät verkkolaskusanoman muodot eli formaatit ovat Finanssialan keskusliiton määrittelemä standardi Finvoice versio 2.01 ja Tieto Oyj:n kehittämä Teapps XML versio 2.7.2. Ulkomaisista verkkolaskuformaateista yleisiä ovat mm. Svefaktura, UBL ja PeppolBIS.

Kuvio 1: Verkkolaskuoperaattorin laskusanoman käsittelyprosessi.

Laskun perustiedoille ja laskutusaiheelle on laskusanomilla aina selkeät kentät, mutta yksityiskohtaisten lisätietojen välittymisessä voi olla eroja formaateista riippuen. Laskut eivät siis välttämättä näytä lähettäjällä ja vastaanottajalla samanlaiselta. Finvoice- ja Teapps-formaattien kenttien vastaavuudet on kuvattu Tiedon ylläpitämässä vastaavuustaulukossa, ja siihen on merkitty Suomen Taloushallintoliiton määrittämät laskusanoman vähimmäisvaatimukset, jotka tulee täyttää, jotta lasku voidaan välittää eteenpäin (Suomen Taloushallintoliitto ry 2015). Vähimmäisvaatimuksena laskun tiedoilla on arvonlisäverolain vaatimat laskun sisältövaatimukset, mutta taloushallinnon prosessien automatisointia varten erityisesti tilausnumero ja sopimusnumero ovat vastaanottajan kannalta tärkeitä kenttiä. Laskusanoman mahdollisimman tarkalla sisällöllä on merkitystä laskun vastaanottajan kannalta, sillä mitä enemmän sanomalla on tietoa, sitä paremmin siitä on mahdollista poimia tietoja hyödyntämään taloushallinnon prosesseja. Laskutietojen avulla voidaan esimerkiksi automatisoida laskun käsittelyä aina tiliöinnistä laskun hyväksyntään asti.

Laskun kuvan ja liitetiedostojen välittymisessä voi olla eroja eri laskusanomien formaattien välillä, esimerkiksi Finvoice-laskusanomaan ei sisälly erillistä laskun kuvaa, kuten muihin formaatteihin. Erot on huomioitava verkkolaskutusta käyttöönotettaessa. Muihin verkkolaskusanomiin nähden Finvoice-sanomassa on myös rajoituksia liitetiedostoille. Niiden maksimimääräksi on rajoitettu 10 kappaletta per lasku ja niiden koko saa olla yhteensä maksimissaan 1 Mt. Lisäksi liitteiden tiedostotyypeissä ja tiedostojen nimen pituudessa voi olla rajoituksia. Kaikki pankkioperaattorit eivät ota lainkaan vastaan liitetiedostoja. (Tieto Finland Oy 2016). Nämä rajoitukset on huomioitava silloin, kun laskun vastaanottajissa on pankkioperaattoreiden asiakkaita, koska pankit vastaanottavat vain Finvoice-muotoista laskusanomaa. TEAPPSXML-sanoma sen sijaan mahdollistaa sekä laskun kuvan että liitteiden välittämisen vastaanottajalle asti.

3.2 Verkkolaskutuksen tulevaisuus

Sähköisen asiakirjojen käsittelyn vauhdittamiseksi Euroopan unioni on laatinut direktiivin 2014/55/EU, jonka mukaan julkisen sektorin tulee pystyä vastaanottamaan ja käsittelemään sähköisiä laskuja, jotka noudattavat sähköisen laskutuksen Eurooppalaista standardia. Direktiivissä on määritelty, että jäsenvaltioiden keskushallinnoilla valmius tulee olla viimeistään 27.11.2018 ja paikallishallinnolla 27.11.2019. (Tieto Finland Oy 2017.) Euroopan standardointijärjestö CEN (the European Committee for Standardization) on määrittänyt EU:n laskustandardin (EN 16931), Suomea tässä standardoimistyössä on edustanut Suomen Standardisoimisliitto. Euroopan tasoiset standardit asettavat vaatimuksia myös suomalaisille Finvoice- ja TEAPPSXML-sanomamuodoille, joten niille ollaan julkaisemassa vuoden 2017 lopulla päivitetyt versiot, jotka ovat sisällöltään yhteensopivia EU:n standardin kanssa. (Verkkolaskun kehitysnäkymät ja eOsoite 2017.)

Euroopassa tapahtuvaa sähköistä laskusanoman välitystä voidaan tehdä PEPPOL-verkoston kautta. PEPPOL tulee sanoista Pan European Public eProcurement On-Line. Se on Euroopan komission vuonna 2008 alunperin alulle laittama hanke, jonka tarkoitus on ollut edistää julkishallinnon ja toimittajien sähköistä kaupankäyntiä. PEPPOL-verkostossa on mahdollista välittää mm. hankintoihin liittyviä sähköisiä asiakirjoja ja laskuja. Vuodesta 2012 verkostoa on kehittänyt OpenPEPPOL-yhteisö, joka koostuu noin sadasta julkisen ja yksityisen sektorin jäsenestä Euroopassa. Yhteisön tehtävänä on kehittää sähköisen asiakirjan sanomavälitykseen liittyviä standardeja ja määrityksiä, kuten verkostoon kuuluvien osapuolien osoitteistoa, viestien välitystä ja dokumenttien välityksessä käytettäviä formaatteja. Verkostosta laskujen välityksestä vastaavat Certified Access Points -tahot, joita on 16 Euroopan maassa ja lisäksi Kanadassa ja USA:ssa. Suomalaiset verkkolaskuoperaattorit toimivat PEPPOL-verkoston Access Pointeina. (OpusCapita Oyj 2017b.)

EU:n direktiivin astuessa voimaan vuonna 2018, on odotettavissa, että verkkolaskut yleistyvät myös ulkomailla. Tämä tarkoittaa sitä, että yrityksissä on varauduttava tarpeellisten tietojen tuottamiseen laskusanomalle ja huolehtia että käytettävä verkkolaskuformaatti sisältää tarvittavat kentät. EN16931-standardin vaatimat sisällölliset täydennykset tulee siis tehdä myös Finvoice- ja TEAPPSXML-laskusanomille. Tieto Finland Oy listaa 9.6.2017 julkaisemassaan ennakkotiedotteessa TEAPPSXML:n täydennettäviksi kentiksi:

- Veropäivä
- Laskun kohde
- Tyyppimerkintä mm. verkkolaskuosoitteelle tai organisaatiotunnisteelle
- Maksukorttitiedot
- SEPA-suoraveloitustiedot
- Alennuksiin ja laskutuslisiin veroprosentti ja veron tyyppikoodit

- Arvonlisävero yhteensä toisessa (kotimaan) valuutassa
- Tietosuojaluokka–tiedon välittäminen laskulla ja liitteellä
- Mahdollisuus merkitä liite luottamukselliseksi.

Kansainvälisissä verkkolaskusanomissa, esimerkiksi Svefakturassa ja UBL:ssä on suomalaisia formaatteja tarkemmat vaatimukset sanoman sisällölle. Ulkomaisissa verkkolaskusanomissa tarkempaa on esimerkiksi hintojen laskenta rivitasolla. Rivitasolla tietojen tulee olla täsmälleen oikein suhteessa yhteissummaan, huomioiden mahdolliset alennukset, laskutuslisät ja arvonlisäverot. Rivitasolla olevat summat tulee täsmätä laskun arvonlisäveron erittelyyn, joka on verkkolaskusanomalla omana kohtanaan. (Tieto Finland Oy 2016.) Eroavaisuudet eri laskusanomien vaatimuksissa aiheuttavat sen, että välitettävät laskut eivät välttämättä välity vastaanottajalle vaan hylkääntyvät puutteellisten tietojen vuoksi. Yhteiset verkkolaskutuksessa käytettävät standardit edesauttavat tietojen yhteensopivuutta ja verkkolaskutuksen yleistymistä yli kansallisten rajojen. Yleistymisen hidasteena on enää kansainvälinen verkkolaskuosoitteiston puute, josta olisi mahdollista saada tietoa siitä, kuka lähettää tai vastaanottaa verkkolaskuja.

3.3 Laskun välitys e-kirjeenä

Sähköiseen laskutukseen voidaan lukea paperimuotoisten laskujen ulkoistettu tulostus-, kuoritus- ja postituspalvelu. Tästä käytetään puhekielessä yleisesti termiä e-kirje, joka on alun perin Postin lanseeraama termi. Postin e-kirje-palvelun nykyinen nimi on iPost ja palvelua tuottaa Postin tytäryhtiö OpusCapita Oyj. Laskun lähettäjä välittää laskun tiedot laskutuskanavasta riippumatta samanlaisessa sanomamuodossa operaattorille, joka tulkitsee sanoman sisällöstä toimitetaanko lasku eteenpäin verkkolaskuna vai paperimuodossa. Näin ollen laskun lähettäjän ei tarvitse tehdä laskutuskanavaan perustuvia valintoja, vaan riittää, että laskusanomalla välitetään verkkolaskuosoite. Sen puuttuessa, lasku postitetaan vastaanottajan postiosoitteeseen. Siinä tapauksessa operaattori välittää laskun tiedot tulostuspalveluun valmiina laskun kuvana pdf-muodossa ja lisäksi laskuun liittyvän xml-tiedoston, jossa kerrotaan laskun vastaanottajan osoitetiedot ja toimitetaanko laskut Priority- tai Economy-luokan kirjeinä (OpusCapita 2017a).

Kun tulostuspalvelu vastaanottaa laskusanoman, tutkitaan onko laskun vastaanottajalla käytössä Netposti-palvelua. Jos asiakas on rekisteröitynyt Netpostiin vastaanottajaksi, voidaan laskusta lähettää kopio Netpostiin. Asiakas on voinut määritellä palvelussa, että hän haluaa vastaanottaa laskun vain Netpostin välityksellä, jolloin paperilaskua ei erikseen lähetetä.

4 Testaus osana ohjelmistokehitystä

Liiketoiminnan prosessien kehittämiseen liittyy olennaisena osana liiketoimintaa tukevien ohjelmistojen kehittäminen. Kehittämistoimenpiteillä pyritään yleensä kustannusten vähentämiseen ja prosessien nopeuttamiseen. Kehittämistoimenpiteiden suorittamiseksi perustetaan yleensä projekti, jonka tehtävänä on viedä läpi kehittämistarpeet käytäntöön. Ohjelmistokehityksen aloitusvaiheessa tehdään vaatimusmäärittely, jonka tarkoituksena on kartoittaa ja selkeyttää liiketoiminnan ja käyttäjien tarpeet. Vaatimusmäärittelyllä tarkoitetaan siis ominaisuuksia, joita ohjelmistolta halutaan ja millä tavoin ohjelmiston tulisi toimia. Vaatimusmäärittelyjen jälkeen aloitetaan ohjelmiston varsinainen suunnittelu toiminnallisten ja teknisten määrittelyjen luomisella.

Testaustoimenpiteillä pyritään todistamaan virheitä (error), jotka voivat johtua käyttäjän tekemästä virheellisestä toiminnasta tai vikoja (fail, bug), jotka johtuvat ohjelman sisältämistä virheistä. Virheen esiintyminen ohjelmakoodissa voi taas aiheuttaa häiriön (failure) ohjelman toiminnassa. Testauksella pyritään siis selvittämään ohjelmiston toiminnan laatu ja luotettavuus ja poistamaan sen toimintaan liittyviä riskejä. Ohjelmistotestauksella varmistetaan, että toteutettu ohjelmisto toimii odotetulla tavalla. Testauksen yhteydessä käytetään usein termejä verifiointi ja validointi. Verifioinnilla varmistetaan, että ohjelmisto vastaa vaatimusmäärittelyjä. Testitulosten ja vaatimusmäärittelyn yhteyttä kutsutaan jäljitettävyydeksi. Sen avulla voidaan tehdä päätelmiä, kuinka hyvin ohjelmisto vastaa asiakkaan alun perin määrittämiä vaatimuksia. Validoinnissa arvioidaan kuinka hyvin ohjelmisto vastaa käyttötarkoitusta (Haikala & Märijärvi 2004, 97-98). Näillä toimenpiteillä pyritään siihen, että toteutus vastaa asiakkaan tarpeita ja toimii oikein. Testaus voidaan lopettaa, kun ohjelmisto ei sisällä enää merkittäviä virheitä, se on toimintakuntoinen ja se toteuttaa kaikki siltä odotetut toiminnot (Kasurinen 2013, 13).

Testauksen työvaiheita ovat sen testaussuunnitelman ja testitapausten laadinta, testiympäristön luonti, testien suorittaminen ja testauksen tulosten tarkastelu. Testaukseen liittyy olennaisesti myös virheiden korjaus. Testauksen määrää rajoittavat käytettävissä oleva aika, raha ja välineet. (Haikala & Märijärvi 2004, 283.) Jotta testaus olisi tehokasta käytettäviin resursseihin nähden, tulee sen suunnitteluun kiinnittää erityistä huomiota. Suunnittelemalla voidaan välttää se, että samoja asioita ei testata turhaan useita kertoja ja testaus kattaisi mahdollisimman hyvin ohjelmiston toiminnallisuudet. Ohjelmistokehityksessä on tärkeää tehdä testausta jo aikaisessa vaiheessa, jotta mahdolliset virheet havaitaan ajoissa ja niiden korjaaminen on vielä helpompaa, jolloin korjauksesta aiheutuva työmäärä ja kustannus jäävät vielä mahdollisimman pieniksi. Huomioitavaa on, että testaukseen käytetty työmäärä ja testitapausten suuri lukumäärä ei välttämättä kerro vielä testauksen tehokkuudesta tai laadusta.

4.1 Vesiputous- ja V-malli

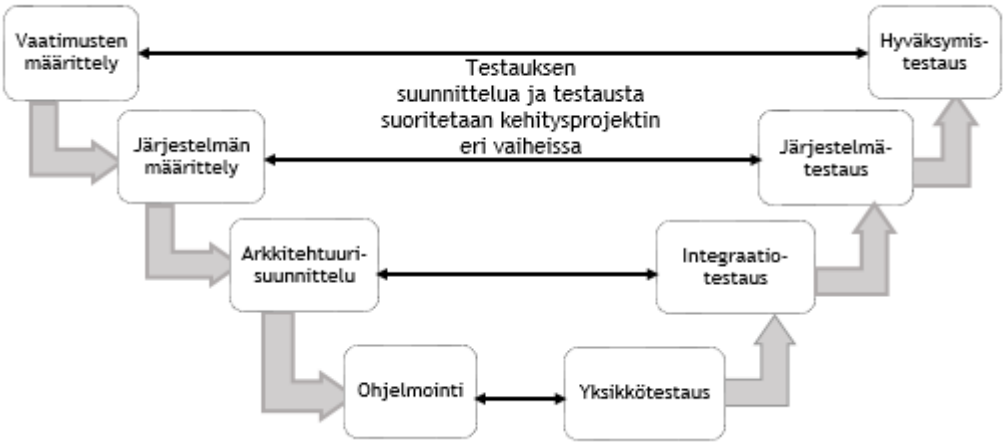
Ohjelmistokehityksessä perinteinen toteutusmalli on vesiputousmalli. Siinä ohjelmiston kehittämisessä edetään vaihe vaiheelta vaatimusmäärittelystä suunnitteluun ja toteutukseen. Toteutuksen jälkeen suoritetaan testaus, sen jälkeen siirrytään tuotantoon ja ylläpitovaiheeseen. Vesiputousmalli vaatii ohjelmiston huolellista alkuvaiheen suunnittelua. Vesiputousmallin heikkous on siinä, että testausta tehdään vasta toteutuksen jälkeen, jolloin kehitysprosessissa ollaan jo pitkällä ja havaitut virheet tai tarvittavat muutokset määrittelyissä voivat aiheuttaa suuria työmääriä vaativia korjaustoimenpiteitä.

Kehittämistä voidaan tehdä myös V-mallin mukaan, sekä suunnittelu että toteutusvaiheessa. V-mallin mukaisesti testauksen suunnittelu tapahtuu vastaavalla ohjelmiston suunnittelutasolla, esimerkiksi järjestelmätestausta suunnitellaan samalla kun ohjelmiston määrittelyjä tehdään. (Haikala & Märijärvi 2004, 288.) Kun testitapauksia aloitetaan tekemään jo ensimmäisistä määrittelydokumenttien versioista lähtien, voidaan huomata hyvin aikaisessa vaiheessa olennaisia aukkoja tai määrittelyvirheitä, jolloin korjauksia ehditään vielä tehdä ennen toteutuksen aloittamista (Black 2009, 503).

V-mallissa ohjelmiston toteutuksen testausta tehdään vaiheittain, siten että testaus aloitetaan jo toteutusvaiheen aikana jatkuen aina käyttöönottoon asti. Näin varmistetaan, että mahdolliset toteutuksen virheet saadaan korjattua mahdollisimman aikaisessa vaiheessa. Eri testausvaiheille on oma testausmenetelmänsä, jolla varmennetaan testauksen laatu. (Kasurinen 2013, 14.) V-mallissa haasteelliseksi tulee se, että testausta pitää yleensä suorittaa eri tasoilla useamman kerran, ennen kuin virheet on saatu korjattua. Koska projekteissa on usein kiire siirtyä vaiheista toiseen, on testauksen näkökulmasta tärkeää miettiä mitkä ovat testauksen lopetuskriteerit, milloin ollaan valmiita siirtymään eteenpäin ja milloin aikataulun viivästymiselle on olennaisia syitä (Black 2009, 504).

4.2 Testaustasot

Testaustasolla tarkoitetaan testauksen tietyn tyyppistä tavoitetta ja testauskohdetta. V-mallin mukaisesti toteutettavassa ohjelmistokehityksessä testaustasoja ovat yksikkötestaus, integraatiotestaus sekä järjestelmätestaus. Järjestelmätestauksen jälkeen voidaan suorittaa vielä erillinen hyväksymistestaus ennen käyttöönottoa. Testaustasojen avulla testaus pilkotaan pienempiin osiin ja jokaisella osalla on oma tavoite ja näkökulma. Eri testaustasoilla käytetään usein erilaisia testausmenetelmiä. (Haikala & Märijärvi 2006, 288.) Ohjelmistokehityksen vaiheiden ja eri testaustasojen yhteydet on kuvattu kuviossa 2.



Kuvio 2: Testauksen V-malli ja testaustasot mukaillen eri lähteistä.

Yksikkötestauksessa testataan ohjelmiston yksittäisiä komponentteja, moduuleja tai funktioita. Testauksen suorittaa yleensä ohjelmiston kehittäjä tai koodaaja ja testaus suoritetaan kehitysympäristössä yleensä siinä vaiheessa kun ohjelman osan viimeinen koodi on saatu kirjoitettua (Black 2009, 5). Tämän tason tarkoitus on varmistaa, että yksittäinen toteutettu toiminto toimii ja ohjelmiston kehityksessä voidaan edetä.

Yksikkötestauksen valmistuttua voidaan siirtyä integraatiotestaukseen. Siinä testataan kehitysympäristössä aiemmin toteutettujen yksittäisten komponenttien tai komponenttiryhmien toiminta, jotka jakavat tietoa keskenään tai kutsuvat toista. Integraatiotestauksia voi olla ohjelmistokehityksen aikana siis useita riippuen siitä kuinka monta eri komponenttia on tarpeen yhdistää. (Black 2009, 6.) Testausvaiheen tärkein tavoite on todeta, että järjestelmän eri osat toimivat yhdessä (Kasurinen 2013, 54). Tätä testausta suorittavat usein ohjelmiston kehityksestä vastaavat, mutta testausta tulee koordinoida projektitasolla, jotta integraatioon liittyvät eri näkökulmat tulevat huomioiduksi (Black 2009, 6).

Järjestelmätestauksessa kohteena on koko järjestelmä. Tämä testausvaihe on ohjelmistokehityksessä kriittinen, koska tässä vaiheessa ohjelmistoa käytetään ensimmäistä kertaa siten kuinka se on tarkoitettu käytettävän. Järjestelmätestaus suoritetaan testiympäristössä, joka vastaa jo hyvin pitkälle tulevaa tuotantoympäristöä. Testausta voidaan tehdä sekä teknisestä näkökulmasta että käyttäjän näkökulmasta, sillä testausta tehdään vaatimusmäärittelyihin perustuen, jotta nähdään vastaako toteutus odotuksia. Siitä syystä testaajana on hyvä toimia ohjelmiston kehitystyöstä mahdollisimman riippumaton taho. Järjestelmätestaus sisältää useita erilaisia testauksia. Siinä testataan toiminnallisuuksien lisäksi ei-toiminnalliset ominaisuudet, jotka kuitenkin vaikuttavat loppukäyttäjään, kuten kuormitustestaus, luotettavuustestaus ja käytettävyystestit. (Haikala & Märijärvi 2004, 290.)

Kun ohjelmisto on läpikäynyt eri testaustasot yksikkötestauksesta järjestelmätestien hyväksyntään, voidaan suorittaa vielä hyväksymistestaus. Testauksen suorittaa asiakas tuotannon kaltaisessa ympäristössä. Tämä on V-mallissa viimeinen vaihe ennen tuotanto- ja ylläpitovaiheeseen siirtymistä. Hyväksymistestauksen tavoitteena on, että ohjelmisto vastaa vaatimuksia ja testauksen tulee läpäistä asiakkaan hyväksymiskriteerit, jotka määritellään testauksen suunnitteluvaiheessa. Ohjelmiston tulee olla tässä vaiheessa riittävän korkealaatuinen ja virheetön, sen tulee täyttää vaatimusmäärittelyissä asetetut asiakkaan vaatimukset. (Kasurinen 2013, 57.)

4.3 Testauksen suunnittelu

Ohjelmiston testauksessa on neljä vaihetta, jotka ovat vaatimusten analysointi, testauksen suunnittelu ja valmistelu, testauksen toteutus ja viimeisenä testauksen päättäminen. Ensimmäisessä vaiheessa tutustutaan saatavilla oleviin dokumentteihin, joiden perusteella luodaan käsitys siitä, mitä ohjelmiston tulisi tehdä ja miten. Läpikäytäviä dokumentteja ovat mm. vaatimusmäärittelyt, toimintokuvaukset ja tekniset kuvaukset. Tämän vaiheen tarkoituksena on selvittää odotettujen testitulosten tunnistaminen, mutta myös havaita aukot ja virheet määrittelyissä. (Hooda & Chhillar 2015.) Tämän jälkeen on mahdollista lähteä suunnittelemaan testausta. Testauksen suunnittelun tärkeä osa on testausstrategia. Siinä määritellään mikä on testauksen kohde ja mitä testauksella tavoitellaan. Testausstrategia ohjaa testausta oikeaan suuntaan ja resursseja voidaan kohdistaa oikeisiin kohtiin. Strategia auttaa myös valitsemaan tilanteeseen sopivat testaustekniikat. (Farrell-Vinay 2008, 9.)

Testausdokumentaatio on määritelty IEEE:n (the Institute of Electrical and Electronics Engineers, inc.) standardissa 829. Organisaation ja testauksen laajuudesta riippuen voidaan kuitenkin arvioida, mitkä dokumentit ovat tarpeellisia. Jokaiseen testaukseen on kuitenkin hyvä sisällyttää vähintään testaussuunnitelma, testitapaukset, käytetyt testausmenetelmät ja suoritetun testauksen raportit.

Testaussuunnitelman laatiminen alkaa samalla kun ohjelmistolle määritellään vaatimuksia, koska testaussuunnitelman tehtävä on varmistaa, että ohjelmisto vastaa asiakkaan tarpeita. Yleensä koko ohjelmiston kehitysprojektille on ylätason testaussuunnitelma ja V-mallin jokaiselle eri tasolle voidaan tehdä erillinen yksityiskohtaisempi testaussuunnitelma.

Testaussuunnitelmassa määritellään mikä on testauksen tarkoitus, missä laajuudessa testausta tehdään, mitkä ovat ne toiminnallisuudet, jotka testataan ja mitä rajataan testauksen ulkopuolelle. Lisäksi testaussuunnitelmassa arvioidaan mitä mahdollisia riskejä testaukseen liittyy. Suunnitelmaan kirjataan aikataulu sekä käytettävissä olevat resurssit, vastuut, työkalut ja testausympäristö. Suunnitelman liitteenä on yleensä tarkempi kuvaus testitapauksista.

Tärkein osuus testaussuunnitelmassa on testauksen keskeytys- ja hyväksymiskriteerit. Keskeytyskriteereitä tarvitaan siihen tilanteeseen, jos ohjelmiston testauksessa alkaa ilmetä runsaasti kriittisiä virheitä eikä testaus näytä enää tuottavan hyötyä. Hyväksymiskriteerit määrittelevät sen, missä tilanteessa voidaan katsoa, että testaus on ollut riittävää ja ohjelmisto vastaa laatuvaatimuksia niin, että testaus voidaan lopettaa. (Farrell-Vinay 2008, 133-135.)

4.4 Testauksen riittävyys

Testauksen tehtävä on löytää virheitä. Mikään testaus ei voi kuitenkaan havaita kaikkia virheitä ja siksi testauksen suunnittelussa on mietittävä, missä vaiheessa voidaan todeta testauksen olevan riittävää siihen nähden, että ohjelmistoa voidaan käyttää luotettavasti. Järjestelmätestauksen kattavuutta voidaan lähestyä eri näkökulmista, mutta tärkeintä on huomioida asiakkaan ja loppukäyttäjän näkökulma, mitkä ovat heidän odotuksensa ohjelmiston toiminnasta ja mihin he ohjelmistoa käyttävät. Testauksen laadukkuutta ja kattavuutta arvioitaessa on hyvä tehdä riskiarviointi siitä, mitä tapahtuu, jos ongelmia ilmenee ja mikä on ongelmien esiintymisen todennäköisyys. (Black 2009, 14.)

Ensisijaisesti ohjelmiston päätoiminnallisuudet tulee testata mahdollisimman kattavasti. Tämän jälkeen pyritään testaamaan tyypillisimmät käyttötapaukset, tyypillisimmät syötteet ja tulosteet. Jos testausresursseja on näiden jälkeen vielä käytettävissä, testataan käyttöliittymän ominaisuudet, toimintojen eri variaatiot ja erilaisten syötteiden ja tulosteiden yhdistelmät. (Farrell-Vinay 2008, 27.)

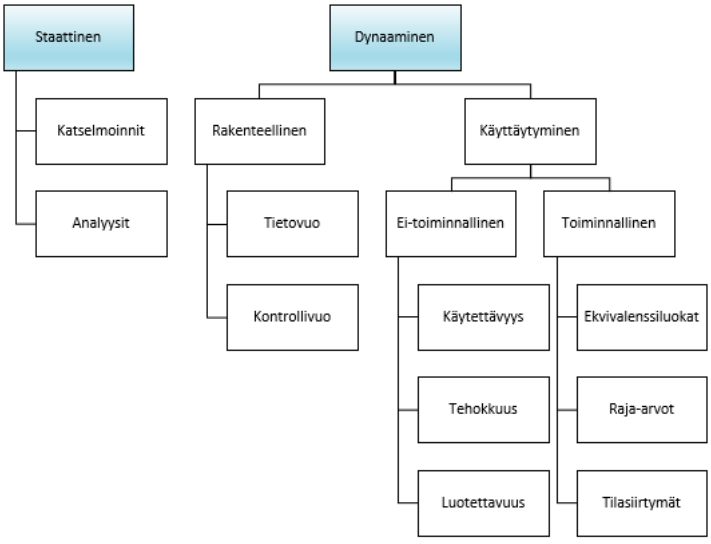
Testauksen kattavuus tarkistaa määrittelydokumenttien pohjalta, jos niitä on käytettävissä. Dokumenteista poimitaan toiminnallisuudet, jonka jälkeen läpikäydään testitapaukset ja tarkistetaan, löytyikö sellainen jokaiselle toiminnallisuudelle. Jos kaikille ei löytynyt testitapausta, tulee arvioida onko sellainen tarpeen. Lisäksi tulee huomioida myös ne seikat, mitä ohjelmisto ei tee, ei saisi tehdä ja pitäisi tehdä, jotta ohjelmiston virheellinen toiminta tulee testattua. Pelkästään toiminnot eivät ole niitä, jotka näkyvät asiakkaille vaan testauksessa tulee huomioida myös, että ohjelmisto toimii vakaasti ja suorituskyky riittää todellisen käytön tarpeeseen, käsiteltävä tieto on oikeaa ja ohjelmisto toipuu virhetilanteista. (Black 2009, 111-112.)

Testausta on mahdotonta tehdä täysin kattavasti, koska käytettävissä oleva aika ja raha ovat usein rajallisia. Siksi onkin tärkeää määritellä selkeästi ne asiat, milloin testaaminen voidaan lopettaa. Usein joudutaan tekemään arviointeja siitä, voidaanko ohjelmiston julkaisussa edetä, voidaanko havaittujen tai havaitsemattomien virheiden kanssa elää, vai ollaanko valmiita myöhästymään aikataulusta ja laittamaan lisää resursseja testaukseen ja ohjelman korjaamiseen. Lopettamiskriteerit perustuvat usein vaatimusmäärittelyihin, eli ohjelman tulee toimia niin kuin asiakas on tilannut. Lopetuskriteerit kirjataan testaussuunnitelmaan. Kriteerit

voivat olla esimerkiksi löydettyjen virheiden määrä ei ole enää kasvanut, kriittisiä virheitä ei enää löydy, kattavuusmittareiden mukaan ohjelma on kattavasti testattu. Kattavuusmittarin lisäksi testauksen kattavuuden arviointiin on olemassa mm. mutkikkuusmittari sekä lause- ja päätöskattavuusmittari. (Farrell-Vinay 2008, 293-295.) Olennaista on kuitenkin se, että suoritetuista testeistä on saatavilla riittävästi dokumentoituja tuloksia päätöksenteon tueksi. Testauksen kattavuutta voidaan niiden pohjalta arvioida, jos tiedetään mitkä ovat ne merkittävät ongelmakohdat, jotka testauksella olisi pitänyt löytyä, ja voidaan todeta, että testauksessa on nämä kohdat käyty riittävän tarkasti läpi. Dokumentaation tulee osoittaa myös, että kaikki testaukseen annetut resurssit on hyödynnetty ja testaus on suoritettu järjestelmällisesti ja asiantuntevasti, sekä havaitut virheet on korjattu tai niiden suhteen on toimenpidesuunnitelma. (Kaner, Bach & Pettichord 2002, 171.)

4.5 Testaustekniikat

Ohjelmistokehityksen aikana suoritettavilla eri testaustasoilla voidaan tehdä testausta eri menetelmiä hyödyntäen. Erilaisia testaustekniikoita on esitelty kuviossa 3. Niiden ymmärtäminen auttaa kattavan ja tehokkaan testauksen suunnittelussa. Testausmenetelmät voidaan jakaa ylätasolla joko staattisiin tai dynaamisiin menetelmiin. Staattinen testaaminen tarkoittaa sitä, että järjestelmää ei suoranaisesti käytetä, vaan järjestelmää tutkitaan esimerkiksi erilaisten analysaattorien tai arviointien näkökulmasta. Staattista testausta tehdään jo ohjelmistokehityksen varhaisessa vaiheessa ja sen tavoitteena on havaita selkeät ongelmakohdat, esimerkiksi koodissa olevat perustavaa laatua olevat virheet. (Kasurinen 2013, 65.) Staattiseksi testaukseksi voidaan lukea jo määrittelydokumenttien katselmointi, sillä niiden oikeellisuus on perusta laadukkaalle ohjelmistolle. (Hooda & Chhillar 2015.)



Kuvio 3: Esimerkkejä erilaisista testaustekniikoista

Dynaaminen testaus on vastaavasti testausta, jota tehdään ohjelmistoa käyttämällä ja sitä tehdään eri testaustasoilla. Siinä tutkitaan ohjelmiston toimivuutta esimerkiksi erilaisilla syötteillä. Dynaaminen testaus voidaan jakaa vielä toiminnalliseen ja ei-toiminnalliseen testaukseen. Toiminnallisessa testauksessa testitapaukset valitaan ohjelmiston toiminnallisuusmäärittelyyn pohjautuen perusteella ilman tietoa sen sisäisestä rakenteesta. Ei-toiminnallisessa testauksessa läpikäydään ohjelmiston sellaisia ominaisuuksia, jotka eivät ole varsinaisia toiminnallisuuksia, mutta vaikuttavat ohjelmiston käyttämiseen. Näitä ovat esimerkiksi luotettavuus, tehokkuus ja käytettävyys. (Finnish Software Testing Board 2015.)

Valittava testaustekniikka riippuu testauksen kohteesta, sen määrittelyistä sekä testaustasosta. Menetelmät eivät sulje toisiaan pois, useampaa testausmenetelmää voidaan käyttää täydentämään testauksen kattavuutta. Kaner ym. (2002, 32) listaa viisi seikkaa, jotka tulee huomioida valitessa käytettäviä testaustekniikoita:

1. Kuka suorittaa testauksen. Millainen näkökulma ja asiantuntevuus testaajilla on.
2. Mitä testataan. Testauksen kattavuus.
3. Miksi testausta pitää tehdä, mitä ovat mahdolliset riskit
4. Miten testaus on tarkoitus suorittaa.
5. Miten pystytään arvioimaan onko testi läpäisty vai epäonnistunut.

Edellä mainittuja kysymyksiä voidaan hyödyntää esimerkiksi tilanteessa, jossa testauksen tavoitteena on määrittelyihin perustuvan testaus. Tällöin testauksen suunnittelussa tulee arvioida testauksen kattavuutta siten, että kaikki määrittelyissä mainitut toiminnallisuudet tulevat testatuksi. Lisäksi tulee pohtia mahdollisia riskejä, jos ohjelmisto ei toimi kuten määrittelyissä on kuvattu ja arvioida testauksen onnistumista. Kolmanneksi arvioidaan, miten voidaan määritellä onko testi suoritettu onnistuneesti, vai havaittiinko virheitä määrittelyihin verrattuna. Hooda ja Chhillar mainitsevat tutkimuksessaan (2015), että ohjelmistotestauksen tärkeimmät osa-alueet ovat toiminnallisuus, suorituskyky ja tietoturva, jotka tulee erityisesti huomioida laadukkaan testauksen takaamiseksi. Tutkimuksen mukaan toiminnallista testausta tehdään yleensä runsaasti, mutta suorituskykyyn ja tietoturvallisuuteen olisi syytä kiinnittää enemmän huomioita.

Eri testaustekniikoita hyödynnettäessä voidaan testitapauksia suunnitella eri menetelmin. Menetelmän käyttö riippuu siitä, mikä näkökulma ja osaaminen testaajalla on. Lasilaatikkotestaus tai toiselta nimeltä valkolaatikkotestaus (Glass Box Testing / White Box Testing) on menetelmä, jossa tutkitaan ohjelman sisäistä toimintaa tarkalla tasolla. Testaus perustuu siis ohjelman rakenteeseen, jonka testaaja tuntee. Testaajat ovatkin tässä tapauksessa usein ohjelmiston kehittäjiä. Testauksessa tutkitaan mitä annettu syöte aiheuttaa ohjelman lähdekooditasolla, tällöin pystytään jäljittämään tarkasti mahdollisten virheiden syyt. Testaajan tulee

siis tuntea järjestelmä syvällisemmin kuin myöhemmin tässä työssä kuvattavassa mustalaatikkotestauksessa, mutta toisaalta taas testaajan tulee ymmärtää myös ohjelman toimintaa. (Kasurinen 2013, 67.) Lasilaatikkotestausta voidaan suorittaa testauksen joka tasolla, mutta erityisesti sitä tehdään V-mallin alkupäässä eli ohjelmistokehityksen alkupuolella, kun taas loppupuolella siirrytään enemmän mustalaatikkotestaukseen. Lasilaatikkotestaukselle tyypillisiä testauskattavuuden mittareita ovat:

- Lauseiden kattavuus (Statement coverage)
- Haarojen kattavuus (Branch / Path coverage)
- Päätösten ja ehtokombinaatioiden kattavuus (Decision coverage).

Toiminnallisen testauksen yksi menetelmistä on mustalaatikkotestaus (Black Box Testing). Siinä ohjelmaa testataan siitä näkökulmasta miten sen halutaan toimivan tuntematta tarkemmin ohjelman koodia. Testauksen pohjana voidaan käyttää määrittelydokumentteja ja käyttötapauksia, mutta testaus voi olla myös tutkivaa testausta. Testaus pyritään suorittamaan jäljittelemällä sitä, miten ohjelmaa aidosti käytettäisiin. Testauksessa annetaan ohjelmalle erilaisia syötteitä ja katsotaan miten ohjelma reagoi, syötteet voidaan määritellä testitapauksiin valmiiksi ja testitapauksen suorituksen jälkeen lopputulosta verrataan siihen, miten ohjelman oli tarkoitus toimia. Tämä on hyvä menetelmä todeta se, miten ohjelma reagoi kun sitä käytetään väärin tai annetaan vääränlaisia syötteitä. Jokaiselle vaatimukselle tulee olla positiivinen ja negatiivinen testitapaus, kelvollisella ja epäkelvollisella syötteellä. (Kasurinen 2013, 65-66.) Mustalaatikko –menetelmää voidaan hyödyntää kaikilla eri testaustasoilla. Mustalaatikkotestausta ovat mm. vaatimusmäärittelyihin ja ohjelmiston määrittelyihin perustuvat testaukset, joissa testitapaukset suunnitellaan dokumentaatioon perustuen (Kaner ym. 2002, 39).

Määrittelyihin perustuvassa testauksessa keskitytään testaamaan määrittelydokumentaatiossa kuvattuja asioita vasten, tavoitteena on todeta, että ohjelmisto toimii kuten on kuvattu. Tässä testausmenetelmässä tulee ymmärtää, mitkä ovat ohjelmiston määrittelyjä ja etsiä kaikki siihen liittyvä dokumentaatio testauksen pohjaksi. Huomioitavaa on se, että kehitettäessä uutta ohjelmistoa, määrittelyissä käydään läpi koko ohjelmiston toiminta, mutta muutosprojekteissa määrittelyt tehdään yleensä vain niistä toiminnoista, jotka muuttuvat. Testauksessa tuleekin siis huomioida myös olemassa olevat toiminnot. Määrittelyihin perustuvassa testauksessa tulee huomioida lisäksi virhehallinta, jota yleensä ei ole kuvattu dokumentaatiossa. Haasteena on se, että määrittelyt usein muuttuvat ohjelmistokehityksen aikana, jolloin testauksen suunnittelussa tulee pysyä ajantasalla muutoksista. Tällöin kehitysprojektin muutoksenhallintaprosessi on erittäin tärkeässä asemassa. Määrittelyihin perustuvan testauksen hyviä puolia on, että sen avulla voidaan testata määritysten laatua, ja parantaa ohjelmistokehityksen prosessia. (Kaner & Fiedler 2016, 112-113.)

Tilapohjainen testaus (State transition testing) perustuu mustalaatikkotekniikkaan. Siinä testitapaukset suunnitellaan tutkimalla, miten ohjelma reagoi tilanteissa, joissa syötteen arvo vaikuttaa siihen mitä ohjelma seuraavaksi suorittaa. Hyväksyttävä arvo siirtää käyttäjän ohjelmassa eteenpäin seuraavaan vaiheeseen, mutta virheellinen arvo palauttaa käyttäjän takaisin alkutilanteeseen. (Kaner ym. 2002, 37.) Testaukseen valitaan siis kaikki tilamuutosten vaihtoehdot.

Mustalaatikkotestauksessa testitapauksiin valitaan syötteet. Jotta testaaminen olisi mahdollisimman tehokasta, päällekkäistä testausta vältetään ja testaus olisi kattavaa, on hyvä tehdä arvoalueanalyysi. Yksi menetelmä on jakaa syötteet ekvivalenssiluokkiin. Siinä ajatuksena on se, että jos ohjelma toimii yhdellä ekvivalenssiluokan syötteellä, toimii se myös muilla saman luokan syötteillä. Testitapauksia suunnitellessa tulee mahdolliset syötteet siis miettiä tarkoin ja jakaa ne sopiviin luokkiin. Tätä jaottelua kutsutaan ekvivalenssiositukseksi. (Haikala & Märijärvi 2004, 292.) Ositusta tehdessä tulee miettiä epäkelvot syötteet, liian pienet tai suuret syötteet ja kelvolliset syötteet. Testauksen kattavuutta voidaan lisätä ekvivalenssiluokkien lisäksi raja-arvoanalyysillä (Boundary value analysis, BVA). Siinä testitapauksiin lisätään ekvivalenssiluokkien rajoilla olevat arvot. (Haikala & Märijärvi 2004, 292.)

Lasilaatikkotestauksen ja mustalaatikkotestauksen välimuoto on harmaalaatikkotestaus (Grey Box Testing). Menetelmässä hyödynnetään mustalaatikkotestauksen tapaan vaatimusmäärittelyihin perustuvia testitapauksia sekä ohjelman koodin tuntemiseen perustuvia testitapauksia. (Kasurinen 2013, 68.) Ohjelmistotestauksessa muita yleisesti tarvittavia testejä ovat savutestaus (Smoke testing), regressiotestaus (Regression testing), sekä suorituskykyyn liittyvät kuormitus- tai rasitustestaus (Load testing / Stress testing). Savutestaus suoritetaan yleensä testauksen aloitustoimenpiteenä. Sen avulla voidaan arvioida, onko tulevan testauksen kohde riittävän valmis varsinaiseen testaukseen. (Kaner ym. 2002, 40.) Kuormitustestauksella tutkitaan, kuinka hyvin ohjelmisto kestää, jos siihen kohdistuu paljon yhtäaikaista käyttöä tai yhtäaikaisia komentoja. Tämä on hyvin tärkeä testi kun ohjelmistolla toimii suuressa organisaatiossa, jossa käyttäjiä on paljon tai jos käsiteltävän aineiston määrä on suuri. Rasitustestauksella (stress testing) selvitetään miten hyvin ohjelmisto toimii sen suorituskyvyn ylärajalla, esimerkiksi silloin kun käytettävissä olevan keskusmuistin määrä on vähäinen. (Finnish Software Testing Board 2015.)

Regressiotestausta suoritetaan silloin kun ohjelmiston toteutuksessa on tehty muutoksia, esimerkiksi korjattu virhe, jonka jälkeen halutaan varmistua, että ohjelmisto toimii oikein, virhe on korjaantunut ja mikään muu ohjelmiston toiminnallisuus ei ole rikkoontunut. Järjestelmätestauksessa regressiotestaus on merkittävässä asemassa, koska siinä testataan jo kokonaisuuksia ja muutoksella voi olla vaikutuksia laajemmalti. Regressiotestausta voidaan tehdä mo-

nella eri tapaa, mutta sen tehokkuuden kannalta on tärkeää miettiä miten se suoritetaan, valitaanko testaukseen kaikki aiemmat testit tai suoritetaanko tietty ennalta valittu testitapausten joukko, joka on kattavuudeltaan riittävä tilanteen todentamiseksi. (Abbas, Bhatti, Shah & Sultan 2017.)

4.6 Testitapaukset

Testitapaus kuvaa tapahtumaa, jolla varmistetaan ohjelman toiminnallisuus. Niiden tavoitteena on tehdä testauksesta johdonmukaista ja tehokasta siten, että eri näkökulmat tulevat huomioiduiksi ja samoja asioita ei testata turhaan useaan kertaan. Yksi testitapaus määrittelee kaikki ne askeleet, mitä vaaditaan, että toiminnallisuus tulee tarkastettua ja mitä toimenpiteitä testin toteutus vaatii. Testitapauksella voidaan testata ohjelmiston toimintaa, mutta myös tilanteita miten se reagoi kun sitä käytetään eri tavalla kuin on tarkoitettu. (Kasurinen 2013, 118.)

Testauksen suunnittelussa tulee huomioida erilaiset ohjelmiston käyttötapaukset. Niiden avulla voidaan varmistua siitä, että jokin kokonaisuus toimii alusta loppuun ja testaus voidaan katsoa olevan riittävän kattava. Käyttötapauksia voidaan luoda vaatimusmäärittelyjen perusteella, joista käy ilmi millaisia toimenpiteitä käyttäjän tulee pystyä tekemään ja mitä eri vaihtoehtoja toimenpiteille voi olla.(Kasurinen 2013, 31.) Käyttötapaus voi sisältää useita eri toimintoja, jotka tulee huomioida testitapauksissa.

Testitapauksia laatimiseksi pitää olla tarkka ymmärrys siitä, mitä ohjelmistolla tehdään. Käyttötapaukset kertovat sen jo hyvin ja tarkentavia tietoja voidaan selvittää mm. liiketoiminnan vaatimusmäärittelyjen, toiminnallisten määrittelyjen ja käyttöohjeiden perusteella. Jos määrittelyjä ei ole, viimeistään testitapausten suunnittelun yhteydessä on ne hyvä dokumentoida, koska testitapauksilla tulee olla jokin perusta, mihin ne pohjautuvat. Testitapauksia suunnitellessa on hyvä läpikäydä lisäksi muita näkökulmia, mitä dokumentaatiot kertovat. On tärkeää huomioida käyttäjän ajattelu ja toimintatavat, sillä ohjelmiston kehittäjällä ja loppukäyttäjällä voi olla erilainen ajatus siitä, miten ohjelmiston tulisi toimia. Ohjelmiston kehittäjien kuten myös loppukäyttäjien kanssa käydyistä keskusteluista voi olla hyödyllistä poimia testattavia kohtia, koska usein näillä osapuolilla voi olla hyvää tietämystä siitä, mitkä ovat ohjelmiston virhealttiimmat kohdat. Aineistoa voi kerätä myös vanhan olemassa olevan ohjelmiston toiminnasta. (Farrell-Vinay 2008, 21-22, 244, 250-251.) Tyypillisesti testauksen edetessä testitapauksia tulee lisää, esimerkiksi silloin kun testaaja havaitsee ongelmakohdan (Kasurinen 2013, 118).

Käytännössä ohjelmistoa on mahdoton testata täydellisesti, esimerkiksi mahdollisten syötteiden määrä on usein erittäin suuri tai ohjelmistossa on valittavana paljon eri polkuja, jolloin

testitapausten määrä kasvaa mahdottomaksi. Ohjelmiston toiminnoista tulee tunnistaa ne seikat, joita asiakas ohjelmistolta odottaa. Ne asettavat testaukselle vaatimukset. Vaatimuksia voivat olla laatuvaatimukset sekä toiminnalliset ja ei-toiminnalliset vaatimukset. (Kasurinen 2013, 121.)

Riskianalyysillä voidaan kartoittaa ohjelmiston toimivuuden riskejä ja niiden toteutumisen todennäköisyyttä. Analyysiä voidaan hyödyntää siten, että sen perusteella valitaan testattavaksi toimintoja, jotka katsotaan olevan riskialttiimpia. Testitapauksissa tärkeysjärjestyksessä kärkeen nostetaan ne tapaukset, joiden virheellinen toiminta on järjestelmän kannalta eniten haitallista. Riskejä analysoidessa käydään läpi uhat, joita ohjelmiston tai sen aineiston käsittelyssä voi olla, mitkä ovat todennäköisimmät virhekohdat ja vikatilanteet sekä näiden tilanteiden mahdollisesti aiheutumat vaikutukset. Riskejä voidaan kartoittaa mm. tutkimalla vaatimus- ja ohjelmistomäärittelyjä ja toteutuneita vikatilastoja, haastatella ohjelmistosuunnittelijoita ja käyttäjiä. (Kaner ym. 2002, 250.)

Toinen menetelmä testitapausten valintaan on suunnitelmalähtöinen valinta. Kun riskianalyysissä pyritään kartoittamaan vikakohdat, jotka todennäköisesti aiheuttavat korjaustöitä ja siten lisäkustannuksia, suunnitelmalähtöinen testitapausten valinta perustuu ajatukseen, että ohjelma ja sen toiminnot vastaavat sille annettuja laatuvaatimuksia. Näiden kahden menetelmän ero on siinä, että riskiperusteinen valinta sopii tilanteeseen, jossa testausresurssit ovat rajalliset. Silloin keskitytään vain korjaamaan kriittiset kohdat ja sallitaan se, että ohjelman toiminnallisuutta voidaan vielä myöhemmin parantaa. Suunnitelmalähtöisessä valinnassa testitapausten suorittamiseen on käytettävissä enemmän resursseja ja testausta voidaan tehdään kunnes laatuvaatimukset on saatu täytettyä. Tällöin käyttöönoton jälkeisiä korjauksia halutaan välttää tai korjauksien tekeminen jälkikäteen on hankalaa toteuttaa. (Kasurinen 2013, 121-122.)

Käytännössä testitapausten valinta tehdään usein hyödyntämällä molempia menetelmiä, mutta kuitenkin niin, että laatuvaatimukset ovat ne, jotka ohjaavat toimintaa. Ohjelman tärkeimmät ominaisuudet valitaan ensimmäisenä testaukseen ja resurssien riittäessä jatketaan yksityiskohtaisemmalle, vähemmän kriittiselle tasolle. (Kasurinen 2013, 123-124.) Testitapausten priorisoinnilla voidaan säästää aikaa sekä kustannuksia, kun testaus suoritetaan sen mukaan mikä on kriittistä ja siinä järjestyksessä miten testauksen odotetaan tuottavan havaintoja. Abbas ym. toteavat tutkimuksessaan (2017), että priorisoinnin kautta valitut testitapaukset tuottavat paremmin virhehavaintoja kuin satunnaisesti testaukseen valitut tapaukset. Tutkimuksen mukaan priorisointia voidaan tehdä eri menetelmillä, joissa huomioitavia seikkoja ovat aina asiakkaan vaatimukset, testauksen kattavuus, kustannusvaikutukset sekä testitapausten historia, miten niillä on mahdollisesti aiemmin tehty havaintoja ohjelmiston toiminnasta.

Testaussuunnitelmassa otetaan kantaa siihen, milloin testaus voidaan lopettaa, vaikka kaikkia testitapauksia ei olisikaan ehditty suorittamaan. Näissä testauksen lopetusehdoissa määritellään esimerkiksi, että kriittiseksi katsottuja virheitä ei enää löydy, järjestelmä vastaa asetettuja laatuvaatimuksia tai testausresurssit on käytetty. (Kasurinen 2013, 123-124.)

Testitapaus kuvaa siis tapahtumaa, jolla varmistetaan ohjelman toiminnallisuus. Testitapauksen on tarkoitus määritellä kaikki ne askeleet, mitä vaaditaan, että toiminnallisuus tulee tarkastettua ja mitä toimenpiteitä testin toteutus vaatii. Testitapausten dokumentointiin olemassa useita erilaisia pohjia, esimerkiksi standardissa IEEE 829 on määritelty pohja testitapausten kuvaamiselle.

Testitapaukseen kirjattavan informaation tarkkuustaso riippuu paljolti siitä, ketkä tulevat testitapauksia käyttämään. Jos testitapauksia hyödyntävät henkilöt, jotka eivät tunne asiaa, tulee testitapauksissa olla tiedot hyvinkin tarkkaan kuvattuna. Tarkan tiedon puolesta puhuu myös se, että testaajan ei tarvitse itse tehdä olettamuksia tai johtopäätöksiä ja testaus on silloin luotettavampaa. Dokumentoinnin tarkkuutta pohtiessa tulee arvioida ketkä toimivat testaajina, mikä on heidän tietämystasonsa asiasta entuudestaan, onko testitapausta tarkoitus hyödyntää eri tilanteissa ja voidaanko testitapaukseen kuvata valmiita arvoja ja syötteitä. Tärkeintä testitapausten dokumentoinnissa on kuitenkin se, että toimintatavasta on sovittu, dokumentointitapaa käytetään yhtenäisesti ja se soveltuu hyvin testaustarpeeseen. Dokumentointitapaa tärkeämpi asia on kuitenkin se, miten testitapauksia suunnitellaan. (Black 2009, 105-107.)

Jokainen testitapaus tulee yksilöidä numeroinnilla, jotta testauksen aikana kommunikoidessa ja testauksen dokumentoinnissa voidaan viitata oikeaan testitapaukseen yksiselitteisesti. Numeroinnin lisäksi testitapauksen otsikossa kuvataan mitä ohjelmiston osiota, toimintoa tai esimerkiksi riskiä testataan.

Testitapausten kuvaus-osiossa kerrotaan, mitä testitapauksella on tarkoitus todentaa. Kuvauksesta tulee ilmetä mitä määrittelyn kohtaa sillä testataan. Kuvauksessa annettaan kaikki kaikki tarvittava informaatio siitä, mitä perusedellytyksiä testaukselle on ja mitä testiympäristöä ja testausaineistoa käytetään. Kuvauksessa kerrotaan testitapauksen alkutilanne, mitä alustavia toimenpiteitä tulee olla tehtynä alustavat toimenpiteet. Jos testitapaus on riippuvainen muista testitapauksista, tulee siihen kirjata mitä tulee olla testattuna ennen sitä. (Kasurinen 2013, 120.). Kuvauksessa otetaan myös kantaa testitapauksen kriittisyyteen ja tärkeyteen. Testitapaukseen voidaan liittää lisäksi dokumentteja, jotka tarjoavat lisäinformaatiota.

Tämän jälkeen kuvataan eri vaiheet, mitä testauksessa tulee suorittaa. Ensimmäisenä askeleena tulee olla kuvaus siitä tilanteesta, mistä testi aloitetaan, mitä perusedellytyksiä tehtävän suorittamiselle pitää olla valmiina. Tarvittaessa kirjataan syötteet, mitä testaajan tulee käyttää. Tässä yhteydessä testaajalle annetaan tarkat tiedot, jotta testaus voidaan suorittaa juuri siten kun on tarkoitettu ja testauksen tulos on luotettava. Lopuksi viimeiseksi askeleeksi kuvataan tilanne, johon testitapaus päättyy. Testitapaus pitäisi olla mahdollista toistaa täsmälleen samalla tavalla, jolloin on tärkeää, että testitapaus kuvaa kaikki vaiheet johdonmukaisesti.

Testitapauksessa tulee kuvata myös se, mitä odotetaan lopputulokseksi. Lopputuloksen määrittelyssä tulee kertoa esimerkiksi mikä on vasteaika, missä ajassa ohjelmiston pitää toimenpiteestä suoriutua, minkä arvon ohjelmiston tulee tulostaa ja mihin tilaan järjestelmän tulee jäädä toimenpiteiden jälkeen. Tämän perusteella testaaja tietää, toimiko ohjelmisto niin kuin on tarkoitettu. Odotetun lopputuloksen perusteella tehdään päätelmät siitä, onko suoritettu testitapaus onnistunut vai epäonnistunut. (Farrell-Vinay 2008, 149-152.) Kuviossa 4 on esimerkki hyvästä testitapauksen sisällöstä.

Testitapauksen yksilöivä numero: 123478

Otsikko: Kirjautumissivu - kirjautuminen sovellukseen
Kriittisyys: 1 Kriittinen

Kuvaus: Käyttäjä pystyy kirjautumaan sovellukseen syöttämällä etusivulla oleviin kenttiin oman käyttäjätunnuksen ja salasanan. Käyttäjä on jo aiemmin rekisteröitynyt palveluun.

Testin suorituksen vaiheet:
1. Käyttäjä avaa Internet Explorer versio 11 -selaimen
2. Käyttäjä kirjoittaa osoiteriville sovelluksen osoitteen (www.lasku.com)
3. Käyttäjä kirjoittaa käyttäjätunnus-kenttään sähköpostiosoitteensa (tepa.testaaja@lasku.com)
4. Käyttäjä kirjoittaa salasana-kenttään salasanansa (xfgletRger23)
5. Käyttäjä painaa hiirellä Kirjaudu-painiketta.
6. Sovelluksen aloitussivu avautuu käyttäjälle.

Odotettu lopputulos: Käyttäjälle avautuu osoitteesta Lasku-palvelun kirjautumissivu. Hän pystyy syöttämään kirjautumissivulla oleviin kenttiin omat tunnuksensa ja kirjautumaan sisään ilman huomautuksia. Sovelluksen pääsivu aukeaa kirjautumisen jälkeen ja on valmis käytettäväksi.

Kuvio 4: Esimerkki testitapauksen sisällöstä, mukaillen eri lähteistä.

Testitapauksen suorittamisen ohjeet on tärkeää kirjoittaa tarkasti, jotta tarvittaessa tiedetään mitkä toimenpiteet ovat johtaneet virhetilanteeseen. Näin virhetilanteen toistaminen on helpompaa ja ongelman ratkaisuun on paremmat lähtökohdat. Testitapausten suunnittelussa

onkin haasteellista tehdä riittävän tarkalla, mutta ei liian tarkalla tasolla, jolloin niiden laatiminen käy liian suureksi työksi ja testien suorittaminenkin voi muodostua hitaaksi kun ohjeistusta on liikaa. Testitapausten sisältöön vaikuttaakin se, ketkä suorittavat testauksen ja mikä on heidän osaamistaso asiasta entuudestaan. Yhteen testitapaukseen on hyvä sisällyttää asiakokonaisuuksia, mutta on huolehdittava myöskin siitä että testitapauksesta ei muodostu liian isoa testauskokonaisuutta. Testitapausten suunnittelun jälkeen ne on hyvä katselmoida, jotta mahdolliset aukot tai virheet testien suorittamisen toimenpiteissä tulee havaittua ja korjattua.

5 Tutkimusmenetelmä

Tämä opinnäytetyö oli toimeksiantajalle tehtävä tutkimuksellinen kehittämistyö, jonka lähtökohta oli toimeksiantajan kehittämistarve. Työssä kehitettiin yrityksen testausprosessia ja tuotiin siihen uutta ammatillista tietämystä. Ratkaisuna oli lista testitapauksia sähköisen laskutuksen integraatio- ja järjestelmätestaukseen. Toiminnallinen opinnäytetyö on usein projektiluonteinen kehittämistyö, joka etenee järjestelmällisesti lähtien liikkeelle suunnittelusta ja edeten hallitusti ja analysoiden toteutukseen. Ensin on tärkeää tunnistaa oikein ratkaistava ongelma ja löytää tietoperustaa siihen. Työ etenee aktiivisessa vuorovaikutuksessa toimeksiantajan kanssa, jotta varmistutaan siitä, että työ vastaa organisaation tarpeita. Toteutuksen jälkeen arvioidaan lopputuotosta, oman osaamisen kasvamista ja miten kerättyä tietoa hyödynnettiin ratkaisussa. (Ojasalo, Moilanen & Ritalahti 2015, 19-21.)

Toiminnallisen tutkimuksen tiedonkeruu vastaa laadullista tutkimusta ja soveltuvia menetelmiä ovat esimerkiksi havainnointi, haastattelut, kyselyt ja kirjalliset lähteet. Niiden tietosisältö analysoidaan ja arvioidaan, ja sen pohjalta käytetään osana organisaation ongelman ratkaisua (Kananen 2014, 77). Tärkeintä on valita kehittämistyötä parhaiten tukevat menetelmät. Tutkimuksen luotettavuutta lisää useamman tutkimusmenetelmän käyttö, tätä kutsutaan triangulaatioksi (Hirsjärvi, Remes & Sajavaara 2005, 218). Eri menetelmät voivat kuitenkin tuottaa erilaisia tuloksia, jolloin tutkijan on tärkeää perustella käyttämänsä menetelmät ja ratkaisut, jotta luotettavuutta voidaan arvioida (Kananen 2014, 136-137).

Testitapausten suunnittelun viitekehyksenä toimii painettu kirjallisuus sekä verkosta löytyvää materiaali. Kirjallisuuskatsauksesta poimitaan tutkimuksen kohteena olevan ongelman kannalta olennaiset asiat, jonka kautta saadaan testitapausten suunnittelulle selkeä runko, joka toimii kehittämistyön pohjana. Kirjallisuuskatsauksen kautta pyritään lisäämään työn tuotoksen luotettavuutta eli validiteettia. (Kananen 2014, 99.)

Testitapaukset pohjautuvat usein erilaisiin järjestelmädokumentteihin ja määrittelyihin. Näin ollen dokumenttianalyysi sopii tämän kaltaisen kehittämistyön yhdeksi tutkimusmenetelmäksi. Dokumenttianalyysi tarkoittaa sitä, että erilaisesta kirjallisesta materiaalista tehdään

päätelmiä, materiaalia voivat olla pöytäkirjat raportit, artikkelit tai kuvaukset. Dokumentit analysoidaan järjestelmällisesti ja sen perusteella saadaan lisää ymmärrystä kehittämiskohteesta. Dokumenttianalyysiä tehdessä on käytettävä lähdekritiikkiä ja huomioitava missä määrin tiedot ovat relevantteja. Sisällön analyysin perusteella saadaan kerättyä tietoa järjestelmän toimivuudesta. Olennaista on löytää toimintoja ja saada siten aineistoa testitapauksien suunnitteluun. (Ojasalo ym. 2015, 136-137.)

Dokumenttianalyysiä täydentävänä menetelmänä käytetään lisäksi havainnointia. Testitapauksien suunnitteluun kerätään aineistoa mm. projektikokouksista ja keskusteluista organisaation ja operaattorin asiantuntijoiden kanssa. Nämä ovat arvokasta käytännön tason tietoa. Havainnointi tapahtuu tässä tutkimuksessa aktiivisen osallistumisen kautta. Havainnoin kautta saatavan aineiston määrään vaikuttaa paljon se, mikä on tutkijan suhde tutkittavaan ongelmaan. Jos asiayhteydet ovat tutkijalle tuttuja, pystyy lyhyelläkin havainnoinnilla saamaan aineistoa. (Kananen 2014, 80) Tarkennuksia havaintoihin voi tehdä vielä asiantuntijoiden yksilöhaastatteluilla, jotta havainto voidaan pitää luotettavana. (Kananen 2014, 83.) Havainnot käydään läpi testitapausten suunnittelun yhteydessä ja varmistetaan, että esille tulleet asiat on huomioitu testitapauksissa. Kehittämistyössä havainnointia voi tehdä eri tilanteissa, riippuen siitä minkälainen mahdollisuus on osallistua organisaation toimintaan. Havainnointia voi tehdä käytännön työn ohessa erilaisissa keskustelutilanteissa ja kokouksissa. Tyypillisesti havainnointi tapahtuu pitkällä aikavälillä. (Ojasalo ym. 2015, 42.)

Toiminnallisen tutkimuksen luotettavuuden arviointi perustuu siinä käytettyjen menetelmien, kerätyn tiedon ja tulosten tarkkaan dokumentaatioon. Tämän perusteella voidaan arvioida tulosten luotettavuutta. Tutkimuksen luotettavuutta arvioidaan validiteetin ja reliabiliteetin kautta. Validiteetilla tarkoitetaan sitä, onko tutkimuksessa tutkittu oikeita asioita. Tämä varmistetaan käyttämällä oikeita tutkimusmenetelmiä ja mittaamalla oikeita asioita. Reliabiliteetilla arvioidaan sitä, ovatko mittareiden antamat tulokset oikeita ja toistettavissa. (Kananen 2014, 126-127.)

Tutkimuksellista kehittämishanketta arvioidaan vertaamalla tutkimuksessa esitettyjä tavoitteita lopputulokseen. Arvioinnin tekee yleensä toimeksiantaja, jolla on paras käsitys työn tuloksellisuudesta. Toiminnallisen tutkimuksen tulokset voidaan katso olevan pätevät vain käsiteltyyn tapaukseen. Mutta jos tutkimuksen lähtökohdat on kuvattu hyvin, voidaan niiden perusteella tehdä päätelmiä siitä, ovatko tulokset siirrettävissä toiseen tilanteeseen ja miten yleistettävissä tulokset ovat. (Kananen 2014, 134-137, 154.) Tutkimusta voidaan pitää reliaabelina, jos enemmän kuin yksi arvioija päätyy samaan tulokseen (Hirsjärvi ym. 2005, 216).

6 Sähköisen laskutuksen testitapausten suunnittelu

Kohdeyrityksessä uudistetaan talouden tietojärjestelmiä osana suurempaa IT-arkkitehtuurin muutosta. Uudistukset koskevat mm. talouden käytössä olevan SAP-järjestelmän myyntilaskutusta, jossa käyttöönotetaan uusia ominaisuuksia sekä parannetaan olemassa olevia toiminnallisuuksia. Tämä hanke on alkanut vuonna 2016 ja talouden SAP-järjestelmän osalta käyttöönotto tapahtuu vuonna 2018. SAP-järjestelmään tehtävät muutokset vaikuttavat järjestelmästä lähtevään laskusanomaan, joka välitetään sähköisen laskutuksen operaattorille. Operaattorin tehtävä on muodostaa asiakkaalle lasku laskusanoman perusteella tai välittää laskusanoma eteenpäin asiakkaan käyttämälle operaattorille. Testauksen kohteena olevalla sähköisellä laskutuksella tarkoitetaan tässä yhteydessä kaikkea lähtevää laskuliikennettä, mitä talouden SAP-järjestelmästä lähetetään asiakkaille, koska varsinaista laskun muodostamista ja välittämistä asiakkaalle huolehtii operaattori.

Testitapausten suunnittelua varten oli tarpeen saada käsitys sähköisen laskutuksen kokonaisuudesta ja eri toiminnoista, prosessissa tapahtuvista virhetilanteista sekä SAP-projektin myötä tehtävistä muutoksista nykyiseen toteutukseen. Testauksen onnistumisen kannalta on tärkeää tuntea olemassa oleva prosessi ja siihen liittyvät toiminnot, jotta voidaan arvioida muutosten vaikutukset ja riskit. Tiedossa olevien virhetilanteiden läpikäynti ja huomioiminen testauksessa on myöskin tärkeää, koska laskutusprosessin tulee toimia mahdollisimman häiriöttä ja häiriötilanteista toipuminen tulee tapahtua nopeasti niin, että siitä ei synny merkittävää haittavaikutuksia yrityksen maineelle tai taloudelle.

6.1 Nykyprosessi

Ennen SAP-projektin määrittelyjen valmistumista, aloitettiin läpikäymään nykyistä laskutusprosessia ja siihen liittyviä toimintoja. Lisäksi läpikäytiin minkälaista dokumentaatiota prosessista on olemassa, ja mitä lisäselvityksiä on tarpeen tehdä testitapausten suunnittelua varten. Nykyprosessi tulee tuntea hyvin, jotta testauksessa voidaan varmistua siitä, että olemassa olevat toiminnallisuudet toimivat ja miten uudet toiminnallisuudet vaikuttavat niihin. Kokonaisuuden ymmärtäminen auttaa ymmärtämään yksittäisiä toiminnallisuuksia ja niiden vaikutuksia, esimerkiksi virhetilanteessa. Prosessiin liittyviä toiminnallisuuksia on kuvattu kohdeyrityksen sisäisessä dokumentissa, operaattorin ratkaisukuvauksessa sekä erillisissä tarkentavissa dokumenteissa.

Kohdeyrityksessä sähköisen laskutuksen eri laskutyyppejä voivat olla sekä yrityksille että kuluttajille lähetettävä veloituslasku, hyvityslasku ja korkolasku. Samaa reittiä välitetään lisäksi asiakkaiden maksumuistutukset. Asiakastiedot tulevat SAP-järjestelmään erillisestä Master Data –järjestelmästä. Asiakastieto sisältää asiakasnumeron ja osoitetietojen lisäksi myös tiedon siitä, millä kielellä lasku muodostetaan sekä laskutuskanavan, mitä kautta asiakas haluaa

vastaanottaa laskun. Laskutuskanavia voivat olla paperilasku, josta käytetään myös termiä e-kirje, verkkolasku sekä kuluttajan e-lasku tai suoramaksu. Maksumuistutukset ja korkolaskut lähetetään nykyprosessissa asiakkaille aina e-kirjeinä. E-kirjeitä voidaan voidaan välittää Postin Netposti-palveluun, jos asiakas on rekisteröitynyt sen käyttäjäksi.

Laskutusta tapahtuu useissa laskutusyksiköissä kuukauden jokaisena päivänä, mutta erityisesti kuukauden ensimmäisinä päivinä laskutusta tapahtuu hyvin paljon. Myyntitilauksista muodostuu laskutusajon jälkeen laskutusaineisto, jossa jokaisesta veloitus- ja hyvityslaskusta muodostuu Invoic idoc-muotoinen laskusanoma. Invoic idoc on SAP-järjestelmän käyttämä tiedonsiirron sanomamuoto. Maksumuistutuksista ja korkolaskuista muodostuu aina valmis pdf-tiedosto, koska ne toimitetaan asiakkaille aina e-kirjeinä. Laskusanomista muodostetaan eriä, yksi erä voi sisältää maksimissaan 1000 idoc-tiedostoa. Erä voi olla kooltaan maksimissaan 20 Mt. Pdf-tiedostoista muodostetaan erillinen erä. Eristä koostuva aineisto siirretään liittymätyökalujen avulla tiedonsiirtopalveluun, josta aineisto lähetetään muuttumattomana FTP-yhteydellä operaattorille. Aineistoja siirretään ajastetusti päivän aikana viisi kertaa ja siirto voi sisältää useita eriä.

Operaattorin palvelussa on jokaiselle laskuja lähettävälle yritykselle oma lähetystili, jota käytetään laskujen välitykseen. Lähetystilin kautta voidaan tarkastella sen kautta kulkevaa laskuliikennettä, ja sitä kautta voidaan todentaa laskujen käsittelyajat ja tarkemmat tiedot mihin laskut on välitetty. Kun operaattori vastaanottaa laskusanomat, se purkaa erät yksittäisiksi laskusanomiksi. Operaattori kuittaa tiedostot vastaanotetuksi, jonka jälkeen kohdeyritykselle toimitetaan csv-muotoinen raportti vastaanotettujen tiedostojen määristä yrityksittäin. Raportti toimitetaan FTP-yhteydellä kohdeyritykselle, joka vertaa saatua tietoa SAP:sta lähteneiden tiedostojen lukumäärään. Näin valvotaan, että operaattori on vastaanottanut kaikki laskusanomat. Tällä hetkellä keskimääräinen laskusanoman käsittelyaika on noin kolme minuuttia, mutta kooltaan suurimpien laskujen kohdalla käsittelyaika voi olla noin 8 minuuttia.

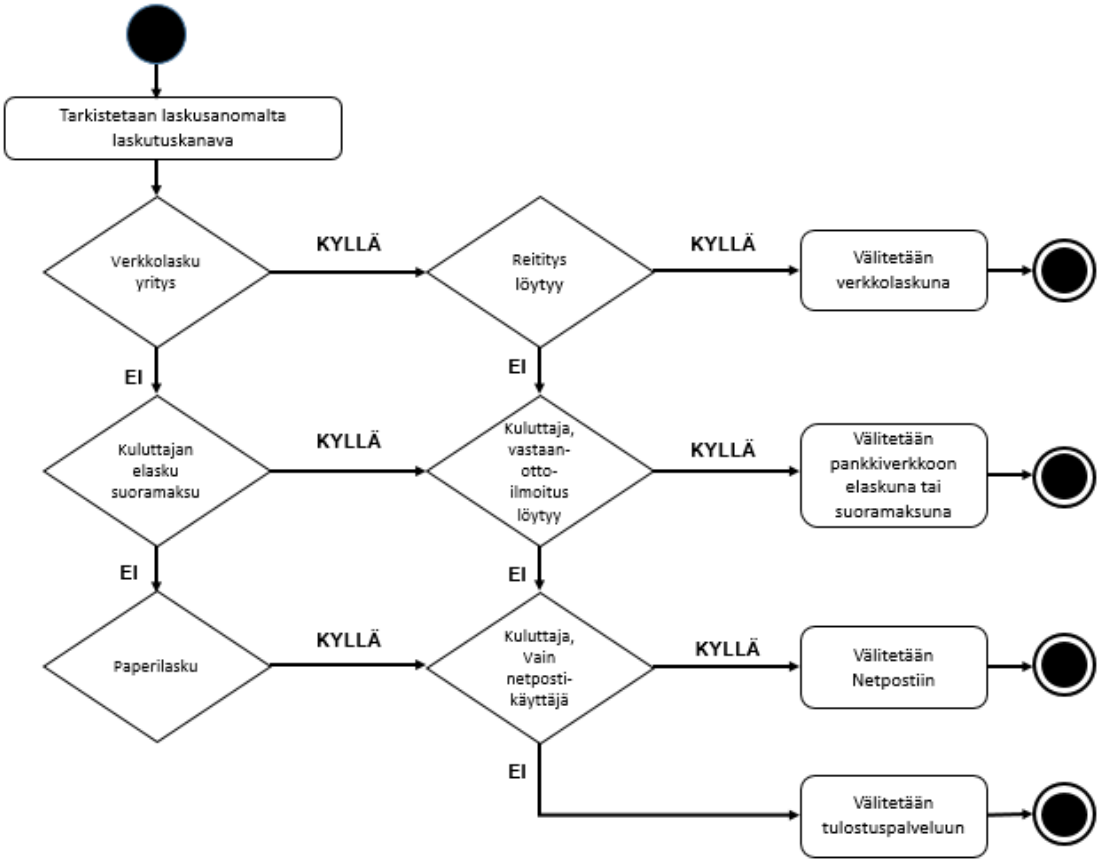
SAP-järjestelmän laskutuksesta muodostuvasta Invoice idoc-muotoisesta laskusanoman sisällöstä on tehty määritykset, mitä tietoja sanoman eri segmenteissä välitetään. Nämä on kuvattu erillisessä teknisessä määrittelydokumentissa, jonka perusteella operaattorin on mahdollista rakentaa omaan palveluun tietojen poiminta laskusanomalta ja siirto eri laskuformaatteihin. Kuviossa 5 on esitelty ote idoc-kenttien määrittelydokumentin sisällöstä. Lisäksi laskun kuvasta on tehty tekninen määrittely. Siinä kuvataan laskun ulkoasun lisäksi myös ne tiedot, mitä laskusanomalta poimitaan ja mihin kohtaan kohtaan tieto tulee tuottaa.

Idoc segment	Segment definition	Field	Rule / Qualifier	Translation	Definition in Finnish	SAP Table	SAP Field
E1EDK14	E2EDK14	orgid[35]	qualf = 003	Idoc organization (003 : Delivering company code)	Yhtiökoodi	VBRK	BUKRS
E1EDKA1	E2EDKA1	name1[35]	parvw = BK	Invoicing company name (BK : Company code address)	Laskuttavan yrityksen nimi	ADRC	NAME1
E1EDKA1	E2EDKA1	name1[35]	parvw = RE	Name 1 (RE : Invoice recipient)	Nimi 1	KNA1	NAME1
E1EDKA1	E2EDKA1	name2[35]	parvw = RE	Name 2 (RE : Invoice recipient)	Nimi 2	KNA1	NAME2
E1EDKA1	E2EDKA1	stras[35];	parvw = RE	Street and house number 1	Lähiosoite 1	KNA1	STRAS
E1EDKA1	E2EDKA1	land1[3]	parwe = RE	Country Key	Maakoodi	KNA1	LAND1
E1EDKA1	E2EDKA1	pstlz[9]	parvw = RE	Postal code	Postinumero	KNA1	PSTLZ
E1EDKA1	E2EDKA1	ort01[35]	parvw = RE	City	Paikkakunta	KNA1	ORT01
Z1LISAO	Z2LISAO	zztied[70]	zzqualf = 500	Name of the country	Maan nimi	T005T	LANDX

Kuvio 5: Esimerkki idoc-laskusanoman määrittelystä.

Kun operaattori vastaanottaa aineiston, se validoidaan. Tässä vaiheessa tarkastetaan, että sanoma on eheää ja että se sisältää kaikki tarvittavat tiedot käsittelyä varten. Operaattori tutkii laskusanomasta vastaanottajan tiedot ja selvittää, missä muodossa lasku pitää välittää eteenpäin. Operaattorin tehtävä on muuntaa laskusanoma vastaanottajan käyttämään sanomamuotoon sekä muodostaa sanomasta myös varsinainen laskun kuva annettujen määritysten mukaisesti. Sen jälkeen laskusanoma ja kuva välitetään vastaanottajan tarvitsemassa muodossa eteenpäin. Jos asiakkaan laskukanavaksi on merkitty paperilasku, lasku toimitetaan tulostuspalveluun, jossa lasku tulostetaan paperille, kuoritetaan ja postitetaan asiakkaalle. Jos asiakkaalla on käytössä Netposti, laskusta välitetään kopio sinne. Jos asiakas on valinnut vastaanottaa laskut vain Netpostiin, paperitulostetta ei muodosteta.

Laskusanoman tulee sisältää vastaanottajan verkkolaskuosoitteen, jos laskutuskanavaksi on merkitty verkkolasku. Verkkolaskuosoitteen perusteella operaattori etsii reititystietokannasta reitin, mihin lasku tulee toimittaa ja laskusanoma muunnetaan esimerkiksi TEAPPS- tai Finvoice-muotoon riippuen asiakkaan vastaanottamasta formaatista. Aineiston muunnoksen jälkeen tehdään vielä tarkistus, että muunnoksessa on kaikki tarvittava informaatio oikeassa muodossa. Tämän jälkeen verkkolaskut välitetään suoraan vastaanottajalle, jos vastaanottaja käyttää samaa operaattoria. Muussa tapauksessa sanoma välitetään vastaanottajan operaattorille, operaattorina voi toimia myös pankki. Kuviossa 6 on kuvattu laskusanoman laskukanavan päättely.



Kuvio 6: Laskutuskanavan päättely

Sähköisen laskutuksen nykyprosessiin tutustuessa läpikäytiin useita dokumentteja. Materiaalista poimittiin toiminnallisuuksia, jotka vaikuttavat laskuaineiston välittämiseen operaattorille, laskun tietosisältöön, laskun ulkoasun muodostumiseen sekä laskun välittämiseen asiakkaalle. Muistiinpanoja kerättiin Microsoftin OneNote-sovellukseen, jossa eri kokonaisuudet kirjattiin eri sivuiksi ja koottiin sen jälkeen sivulle kaikki aiheeseen liittyvät tiedot. Nykyisen prosessin tuntemus auttoi rakentamaan testauksen runkoa ja keskittämään testausta erityisesti niihin kohtiin, jotka tulevat muuttumaan kehitysprojektissa. Testitapausten suunnittelussa on huomioita asiakkaiden eri laskutuskanavat, mitä tietoja niitä varten laskusanomalta poimitaan ja miten laskusanomalta poimitaan tiedot verkkolaskusanomille ja laskun kuvaan. Käyttötapausten kuvausten kautta saadaan selvyys eri tilanteista (kuluttaja-asiakas, yritys-asiakas, eri operaattorit, verkkolaskuformaatit), jotka tulee huomioida testitapauksissa. Huomioitavaa on myös laskuaineiston välityksen eri vaiheet ja siihen liittyvät valvonnat.

SAP-järjestelmän myyntitilauksien perusteella tehtävään laskutukseen ei pääsääntöisesti ole mahdollista lisätä erillisiä liitetiedostoja. Yhdelle laskulajille on tehty kuitenkin poikkeuskäsittely, koska liitteet ovat tietyissä tilanteissa tarpeellisia, jotta laskuille saadaan kirjanpitolain edellyttämät riittävät tiedot. Tämän laskulajin laskuja on vuodessa noin 230.000 kpl ja

näistä liitteen tarvitsevia laskuja on noin 24.000 kpl. Laskumäärä on oleellinen, jolloin liitteistyksen toiminnallisuudet on tärkeää huomioida testauksessa. Liitteistyksen epäonnistuessa lisätyötä aiheutuu niin laskutukseen kuin laskun vastaanottavalle asiakkaallekin.

Liitteellisten laskujen prosessi alkaa siitä, kun myyntitilaukselle merkitään tieto, kun siihen liittyy erillinen liite. Myyntitilauksesta muodostuu normaaliin tapaan laskuaineisto operaattorille, mutta liitetiedosto toimitetaan operaattorille erikseen sähköpostitse myyntitilauksen käsittelijän toimesta. Sähköpostilla toimitettava liitetiedosto tulee nimetä aina tietyn kaavan mukaisesti: yritysnumero_kustannuspaikka_liitenumero_järjestysnumero, jossa järjestysnumero kertoo montako sivua liitetiedosto sisältää. Tiedoston nimi voi olla siis esimerkiksi 100_319100_1234567_001.pdf. Jos tiedosto on nimetty väärin, operaattori hylkää tiedoston ja lähettäjälle lähtee ilmoitus, joka sisältää hylättyjen liitteiden nimet ja kehotuksen lähettää liitteet uudelleen. Hylättyjen liitteiden nimet tulee korjata ja lähettää uudelleen.

Liitetiedostoja on mahdollista lähettää vain ennalta määritellyistä erikseen sallituista sähköpostiosoitteista. Sallitut osoitteet on ilmoitettu operaattorille, ja jos liitteitä lähetetään muista osoitteista, operaattori ei ota tiedostoja käsittelyyn vaan sähköpostin lähettäjälle lähtee hylkyilmoitus.

Operaattori poimii saapuneesta laskuaineistosta ne laskut, joiden idoc-sanomassa on merkintä liitetiedostosta. Operaattori yhdistää laskun ja liitteen tiedoston nimeen annettujen tietojen perusteella. Mikäli laskulle ei löydy liitettä, lasku käsitellään eteenpäin ilman liitettä. Laskulle yritetään etsiä liitetiedostoa maksimissaan 5 vuorokauden ajan. Samoin jos operaattorin palvelussa olevalle liitetiedostolle ei löydy 5 vuorokauden kuluessa laskua, liite hylätään ja lähettäjälle lähetetään sähköposti-ilmoitus. Poistetuista liitteistä tai ilman liitteitä lähteneistä laskuista lähetetään kohdeyritykselle erillinen virheraportti. Käsittelyprosessiin liittyy poikkeus, mikä tulee huomioida testitapauksissa: jos asiakas vastaanottaa laskuja Finvoice-muodossa ja laskuun kuuluu liite, ohjataan lasku liitteineen tulostuspalveluun, koska pankkien käyttämä Finvoice-verkkolaskusanoman mukana ei voida välittää liitteitä.

Testitapauksissa on huomioitava liitteen merkitseminen laskusanomalle, liitteen lähetykseen liittyvät eri vaiheet ja säännöt sekä operaattorin liitteistykseen liittyvät toimenpiteet. Lisäksi tulee läpikäydä erilaiset virhetilanteet, esimerkiksi liitteen nimi on väärin muodostettu. Finvoice-muotoisen laskusanoman aiheuttama poikkeuskäsittely nostaa esiin myös tarpeen ymmärtää eri verkkolaskusanomatyyppeihin liittyvät vaatimukset. Finvoice laskusanoma on kuvattu Finanssiala ry:n dokumentissa Finvoice 2.1. soveltamisohje (2017) ja Teapps-sanoma on kuvattu Tieto Finland Oy:n ylläpitämässä dokumentissa TEAPPSXML v2.7.2 soveltamisohje (2011).

6.2 Virhetilaston läpikäynti

Testauksessa tulee lisäksi tulee huomioida ne seikat, mitä tapahtuu jos käsiteltävässä laskuaineistossa on virheitä. Pelkästään toiminnot eivät ole niitä, jotka näkyvät asiakkaille vaan testauksessa tulee huomioida myös se, että ohjelmisto toimii vakaasti ja suorituskyky riittää todellisen käytön tarpeeseen, käsiteltävä tieto on oikeaa ja ohjelmisto toipuu virhetilanteista. Tästä syystä on tärkeää läpikäydä nykyisessä tuotantoympäristöstä havaittuja virhetilanteita ja poimia sieltä testaukseen sellaisia, jotka todennäköisesti tapahtuvat myös tulevaisuudessa ja niitä kriittisiä tilanteita, joiden kohdalla ohjelmiston toiminnan olisi pitänyt jo korjaantua.

Operaattorille välitetään laskusanomia useita kertoja päivässä automatisoidusti. Vastaanotettuaan aineiston, operaattori validoi jokaisen sanoman. Jos operaattorille välitettävässä laskuaineistossa ilmenee virheitä, operaattori lähettää siitä virheilmoituksen kohdeyrityksen sähköisen laskutuksen palveluista vastaavalle tiimille. Nämä ilmoitukset kirjataan Jira-sovellukseen, jonne kirjataan myös tilanteiden ratkaisut. Testitapausten suunnittelua varten läpikäytiin Jira-sovellukseen kirjatut tapahtumat ajalta 1.1.2016-25.10.2017. Tapahtumat tallennettiin Excel-taulukoksi, jossa tapahtumat luokiteltiin sen mukaan, mikä oli virheen aiheuttanut syy (liite 1). Tapahtumia tällä aikavälillä oli yhteensä 526 kappaletta ja erilaisia virhetyyppejä löytyi 20 kappaletta.

Sähköisessä laskutuksessa eniten virhetilanteita (441 kpl) oli aiheuttanut asiakkaiden verkkolaskuosoitteen muuttuminen. Verkkolaskun välitys ei onnistu esimerkiksi silloin, kun asiakas on vaihtanut verkkolaskuosoitetta tai operaattoria, mutta ei ole ilmoittanut tilausta tehdessään uusia tietoja. Toinen yleinen virhe on puutteellinen arvonlisäveron erittely laskusanomalla (25 kpl). Esimerkiksi UBL- ja Peppol-verkkolaskuformaateissa on tiukemmat vaatimukset arvonlisäveron erittelylle kuin Suomessa yleisesti käytettävissä Finvoice- ja Teapps-formaateissa. Kohdeyrityksessä käytettävä laskutusjärjestelmä ei vielä toistaiseksi tuota laskusanomalle vaadittavaa erittelyä, joten tästä syystä UBL- tai Peppol-formaatissa laskuja vastaanottavien asiakkaiden verkkolaskut hylkääntyvät.

Liitetiedostoihin liittyviä ongelmatilanteita oli 13 kpl, joten niihin liittyvät toiminnot tulee testata tarkasti. Vapaateksti-kenttään kirjoitettu liian pitkä teksti oli aiheuttanut virhetilanteen kahdeksan kertaa, mikä tulee huomioida testauksessa ja pyrkiä löytämään ratkaisu siihen, ettei liian pitkää tekstiä ole mahdollista edes syöttää laskun tietoihin. Tuplalaskutusta oli tapahtunut kahdeksan kertaa, jolloin testauksessa on huomioitava toiminnot, jotka estävät tuplalaskun lähtemisen asiakkaalle. Seitsemän kertaa aineistossa on ollut erikoismerkkejä, jotka eivät ole sallittuja. Erikoismerkit ovat pääsääntöisesti lähtöisin Master Dataan tallennetusta asiakkaan nimestä tai osoitteesta, joten tämä on tärkeää huomioida testauksessa.

Tilastosta löytyy kaksi liittymään liittyvää tapahtumaa. Toisessa operaattorille siirrettyyn laskuaineistoon on muodostunut NULL-merkkejä, jotka estivät laskun prosessoinnin eteenpäin. Toinen operaattorin palvelun suorituskykyyn liittyvä tilanne on esiintynyt viimeisen vuoden aikana kerran, kun laskuaineisto sisälsi kooltaan niin suuren idoc-tiedoston, että laskun käsittely hidastui ja käytännössä keskeytyi kokonaan. Ongelmatilanteissa olennaista on se, että tilanne saadaan ratkaistua nopeasti ja että tilanne ei estä koko laskuerän tai tulevien laskuerien käsittelyä.

Virhehallinta on yksi testauksen tärkeimmistä osa-alueista, koska virheetöntä järjestelmää ei ole, olennaista on se miten virhetilanteet havaitaan ja miten niistä toivutaan aiheuttamatta liiketoiminnalle merkittäviä häiriöitä. Laskuliikennettä valvotaan prosessin eri vaiheissa, jotta voidaan varmistua siitä että kaikki laskut on välitetty eteenpäin asiakkaille. Jos verkkolaskuosoitteelle ei löydy reittiä, lasku ohjataan tulostuspalveluun ja lähettävälle yritykselle lähetetään ilmoitus virheellisestä osoitteesta. Myös muista mahdollisista virheistä, lähtee virheilmoitus. Jos vuorokauden aikana virheellisiä laskuja on yli 10, laskuja ei ohjata tulostukseen vaan silloin on syytä tutkia onko kyseessä suurempi ongelmatilanne. Tämä toiminnallisuus on tärkeää käydä läpi testauksessa. Tämän fallback-toiminnon lisäksi virhetilanteiden läpikäynnistä tunnistettiin testitapauksiin mm. alv-erittelyt, suorituskyky, virheilmoitukset ja Valvontaraportit.

6.3 Muutosprojektin läpikäynti

SAP-järjestelmän uudistamisen yhteydessä tehtiin muutoksia myyntilaskutukseen ja sillä oli vaikutusta idoc-muotoisen laskusanoman sisältöön ja muodostettavan laskun ulkoasuun. Näillä muutoksilla oli vaikutusta sähköisestä laskutuksesta vastaavan operaattorin palveluun, joka vastaanottaa laskusanoman ja muodostaa varsinaisen laskun ja reitittää sen asiakkaalle. Testitapausten suunnittelua varten oli tärkeää kartoittaa muuttuvat kohdat ja mitkä olivat niiden vaikutukset olemassa oleviin toimintoihin.

SAP-projektissa tuotettiin toiminnallinen määrittely, jossa kuvattiin laskusanoman välitykseen liittyvät toiminnallisuudet. Lisäksi tuotettiin erilliset määrittelyt laskusanoman kenttien sisällöstä sekä laskun ulkoasusta. Ensimmäinen versio määrittelydokumenteista saatiin käyttöön toukokuussa, jonka pohjalta lähdettiin suunnittelemaan muutostarpeita operaattorin palveluun. Keskusteluiden pohjalta operaattori muodosti työmääräarvion ja erittelyn siitä, mitä muutostöitä on tarpeen tehdä. Aiemmin keväällä käydyissä alustavissa keskusteluissa muodostunut käsitys oli se, että muutoksia tulisi vain jonkin verran SAP-järjestelmän muodostaman idoc-laskusanoman sisältöön ja siten operaattorin tulisi tehdä muutoksia vain muuttuneiden kenttien osalta tietojen poimintaan verkkolaskusanomille. Määrittelydokumenttia läpikäydessä ilmeni kuitenkin, että muutoksia tulee paljon niin laskusanomaan kuin muodostettavaan laskun kuvaankin. Koska työmäärä osoittautui suureksi, päädyttiin lopulta siihen, että samassa

yhteydessä päivitetään myös operaattorin palvelun ohjelmisto uusimpaan versioon. Käytännössä tämä tarkoitti sitä, että operaattorin täytyi tehdä kaikki määritykset lähes alusta asti uudelleen. Versiopäivitys olisi lähitulevaisuudessa muutoinkin edessä, joten muiden muutostöiden ohessa, voidaan samalla testaustyöllä testata versiopäivityksen tuomat muutokset. Testaukseen tulee kuitenkin lisätä hieman enemmän toiminnallisuuksia, jotta voidaan todeta että kokonaisuus toimii kuten halutaan.

Muutosdokumentteihin tutustuessa kävi ilmi, että myös Master Data –järjestelmään oli tulossa muutoksia ja sillä oli vaikutusta asiakkaan laskutuskanavan merkintään laskusanomalle. Aikaisemmin laskutuskanava tulkittiin laskusanomalta eri positioista, uudessa toteutuksessa eri laskukanavat on merkitty selkeästi eri numeroin. Tämä tulee ottaa huomioon testitapauksien suunnittelussa. Laskukanavan oikealla tulkinnalla on olennainen merkitys siinä, että asiakas saa laskun toivomallaan tavalla, mutta sillä on myös merkitystä sähköisestä laskutuksesta aiheutuviin kustannuksiin, sillä paperilaskun hinta on verkkolaskua merkittävästi suurempi.

Hyväksytty versio toiminnallisista määrittelyistä valmistui kesällä, jonka jälkeen niitä aloitettiin käydä läpi testitapausten suunnittelua varten. Määrittelyjä läpikäydessä todettiin, että laskutyypin merkitsemiseen on tarpeen tehdä muutos, koska jatkossa oli tarkoitus muodostaa korkolaskusta e-kirjeen sijaan verkkolasku, mutta sen ulkoasu on erilainen kuin tavallisen laskun. Näin ollen laskusanomalla tulee olla selkeästi ilmaistuna laskutyyppi. Lisäksi määrittelyistä puuttui operaattorin tarvitsemia positiotietoja, jotka jouduttiin vielä lisäämään dokumenttiin. Nämä olivat hyvä osoitus siitä, että määrittelydokumenttia on tärkeä käydä läpi aikaisessa vaiheessa. Kun määrittelydokumenteista julkaistiin uudet versiot, ne käytiin vielä uudelleen läpi ja korjattiin tarvittavilta osin testitapausten suunnitteluun poimittua asioita.

Korkolaskun sekä laskujen uuden liitteistys toiminnallisuuden osalta määrittelyjen saaminen koettiin haastavaksi johtuen SAP-projektin aikatauluista. Nämä olivat kuitenkin yksittäisiä kokonaisuuksia, joten testauksen suunnittelussa pystyttiin etenemään myös ilman tarkempia tietoja niistä. Määrittelyjen valmistuttua myöhemmin voidaan vielä tarkistaa onko testitapauksien suunnittelussa käytetyt tiedot olleet oikein.

Muutoksia läpikäytiin kesän ja syksyn 2017 aikana useissa projektipalavereissa sekä sähköposteissa niin SAP-projektin kuin operaattorinkin kanssa. Näistä keskusteluista kerättiin testitapausten suunnittelua varten muistiinpanoja OneNote-sovellukseen (liite 2), jonne jo nykyprosessin läpikäynnin yhteydessä oli kerätty muistiinpanoja toiminnoittain. Muistiinpanojen jäsentelyjen jälkeen SAP-projektin muutoksista testattavien asioiden listalle nousivat erityisesti laskulla näkyvä logo, laskun kuvan otsikot, kielikäännökset (FI, SV, EN), idoc-laskusanomalta tiedon poiminta eri verkkolaskuformaatteihin, operaattorin palvelun versiopäivityksen

vaikutukset, laskujen reititys asiakkaalle, suoraan SAP-järjestelmästä tulevat laskujen liitteet, arvonlisäveron erittelyn oikeellisuus sekä korkolaskun käsittely kokonaisuutena. Yhteenveto suunnitelluista testattavista kokonaisuuksista on esitelty liitteessä 3.

6.4 Testauksen suunnittelu

Sähköisen laskutuksen testaussuunnitelma tehtiin yhteistyössä operaattorin kanssa. Pääpiirteet johdettiin suoraan SAP-projektin testaussuunnitelmasta, koska se määritteli hyvin paljon toimintatapoja. Siinä määriteltiin testauksen strategia, testauksen kohteet, testausresurssit, vastuunjako, testauksen aikataulu, riskianalyysi sekä tuotettavat dokumentit. Testauksen työnjaoksi sovittiin, että operaattori yksikkötestaa oman palvelunsa. Yksikkötestaus on kooditason testausta eli lasilaatikkomenetelmällä tehtävää testausta ja se aikataulutettiin tehtäväksi ennen integraatio- ja järjestelmätestauksen aloittamista. Integraatio- ja järjestelmätestauksen vastuu oli kohdeyrityksellä ja testauksen osallisena oli operaattorin lisäksi myös SAP-järjestelmän toimittaja. Tämän testausvaiheen menetelmäksi voitiin valita harmaalaatikkotestaus, koska testaajilla oli osaamista sekä toiminnallisuuksista että tietorakenteista varsin tarkalla tasolla, mutta ei kuitenkaan tiedetty tarkalla tasolla, miten operaattorin toteutus oli tehty. Idoc-aineiston ja laskun ulkoasun testaamisen osalta tehtiin päätös, että ne testataan tarkasti määrityksiin perustuen siten, että jokaisen idoc-kentän siirtyminen verkkolaskusanomille tarkistetaan ja jokainen laskun kuvalle tuotava tieto läpikäydään. Tämä päätös tehtiin siitä syystä, että nämä ovat erityisesti kohdeyrityksen asiakkaille näkyviä tietoja ja virheellinen toiminta aiheuttaa ylimääräistä työtä ja virheiden korjaaminen tuotantovaiheessa tulisi olemaan erittäin vaikeaa. Myös laskun reititys eri laskutuskanaviin koettiin tärkeäksi testauskohteeksi, koska sen osalta laskusanomaan tulee muutoksia ja vaikutukset asiakkaalle ja tiimin ylläpidolliseen työhön on suuri. Testaussuunnitelmaan merkittiin, että hyväksymistestaus tehdään myöhemmin SAP-järjestelmän hyväksymistestauksen yhteydessä ja sen tarkempi suunnittelu jätettiin vielä tästä testisuunnitelmasta pois.

Testausmetodina oli manuaalinen testaus, koska automaatiotestauksen resursseja ei ollut käytettävissä. Testaussuunnitelman mukaisesti testausprosessi läpikäytiin operaattorin sekä SAP-projektin testauksesta vastaavien henkilöiden kanssa. Keskusteluissa sovittiin tarvittavat testausaineistot ja niistä informointi, sekä havaintojen kirjaamisen ja läpikäynnin periaatteet. Testitapausten suunnittelun työkaluna käytettiin HP Application Lifecycle Management –järjestelmää, jota käytettiin lisäksi testien suorittamiseen ja virhehavaintojen raportointiin. Ympäristönä käytettiin SAP-järjestelmän testiympäristöä, josta toimitettiin testiaineistoa operaattorin palveluun käyttämällä sähköisen laskutuksen testitilejä. Testaus rajattiin koskemaan pelkästään operaattorin palvelua ja siihen liittyviä liittymiä, jota pitkin testiaineistoa toimitetaan operaattorille.

Testauksen aloituskriteereiksi kirjattiin:

- Määrittelyt on katselmoitu ja hyväksytty
- Testisuunnitelma on laadittu ja hyväksytty
- Testitapaukset on dokumentoitu
- Testauksen ja havaintojen hallinnoinnista ja käsittelystä on sovittu
- SAP-testiympäristö on perustettu
- Operaattorin testiympäristö perustettu ja testitilit liitetty siihen
- Testiaineistoa on saatavilla
- Yksikkötestaus on suoritettu ja dokumentoitu.

Testaussuunnitelman tärkeimmäksi katsottava osio on testauksen lopetuskriteerit. Niiden tehtävä on määritellä, milloin testaus on ollut riittävää ja ohjelmisto vastaa laatuvaatimuksia niin, että testaus voidaan lopettaa. Testaussuunnitelmaan kirjattiin lopetuskriteereiksi:

- Kaikki testitapaukset on käyty läpi
- Kaikki kriittiseksi luokitellut havainnot on korjattu, testattu ja suljettu
- Avoimille virhehavainnoille on tehty toimenpidesuunnitelma
- Testauksen päätösraportti on tehty
- Projektin ohjausryhmä hyväksyy testauksen ja antaa luvan siirtyä tuotantoon.

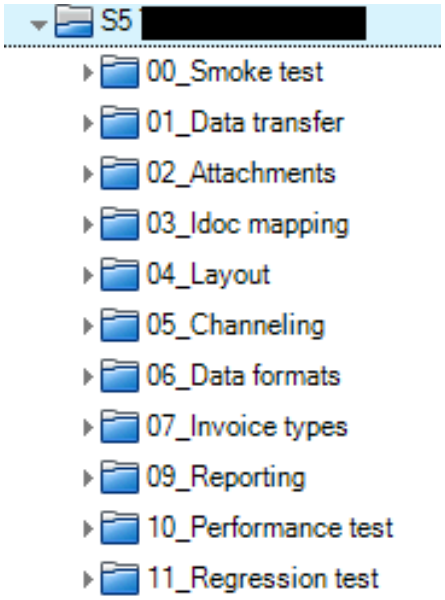
6.5 Testitapausten laatiminen

Testitapausten suunnittelua varten läpikäytiin sähköisen laskutuksen nykyprosessi ja siihen liittyvät toiminnallisuudet, prosessissa tapahtuneet virhetilanteet lähes kahden vuoden ajalta sekä SAP-järjestelmän aiheuttamat muutokset operaattorin palvelun toteutukseen. Kun näitä pohjatietoja sähköisen laskutuksen ja operaattorin toiminnasta sekä projektin tuomista muutoksista oli kerätty OneNote-sovellukseen ja testauksen suunnittelua oli käyty läpi operaattorin kanssa, aloitettiin varsinaisten testitapausten suunnittelu. Läpikäynnin perusteella laadittiin laskutusprosessin yleisimmät käyttötapaukset, joista esimerkkejä on esitelty liitteessä 4. Niiden avulla saatiin kokonaisvaltaisempi ymmärrys siitä, millaisia laskuja prosessissa kulkee ja miten niiden prosessoinnit eroavat toisistaan sekä millaisia vaihtoehtoisia tilanteista prosesseihin liittyy.

Hankeohjeistuksen mukaan testitapaukset tuli luoda HP Application Lifecycle Management –järjestelmään. HP ALM on sovelluskehityksen ja testauksen tueksi tarkoitettu työkalu, jonne on mm. mahdollista luoda testaussuunnitelma ja testitapaukset. HP ALM:n kaltaisen työkalun käytön hyöty on erityisesti testauksen hallinnassa, testattavat asiat on dokumentoidaan järjestelmällisesti ja suoritetuista testeistä ja korjaustoimenpiteistä jää jälki. Kohdeyrityksen

hankeohjauksesta annettiin koulutusta ohjelman käyttöön sekä ohjeistus mm. millaista määritystä ja nimeämiskäytäntöä tulee noudattaa testitapausten asetuksissa, jotta hanketasoinen raportointi on mahdollista. Lisäksi ohjeistuksena oli, että käytettävä kieli on englanti.

Testitapaukset luotiin ensin TestPlan-osioon. Testitapausten jaotteluun käytettiin kansiorakennetta siten, että yhteen kansioon niputettiin yhteen kokonaisuuteen liittyvät testitapaukset ja siten, että ensimmäisenä ovat ne testitapaukset, jotka joiden tulee olla kunnossa, ennen kuin voidaan siirtyä seuraavan kansion testitapauksiin (kuvio 7). Idoc-laskusanomaan ja laskun ulkoasuun liittyvät testitapaukset nimettiin siten, että nimen alussa on kansion numero, joka kertoo mistä kokonaisuudesta on kyse ja sen jälkeen oleva numero kertoo, mistä määrityksen kentästä on kyse. Näin ollen testitapaus voidaan jäljittää sekä kohdeyrityksen laatimaan määrittelydokumentin kohtaan, mutta myös operaattorin omasta palvelustaan muodostamaan dokumentaatioon.



Kuvio 7: Testitapausten kansiorakenne

Testitapausten kuvauksiin kirjoitettiin tarkemmin, mitä testitapauksella on tarkoitus testata ja mihin dokumenttiin testitapaus perustui. Tämän jälkeen määriteltiin testissä suoritettavat askeleet, joiden perusteella testaaja tietää miten edetä. Askeleiden kuvauksiin kirjoitettiin kuvaus siitä, mitä testaajan tulee tehdä sekä toimenpiteen odotettu lopputulos. Ensimmäinen askel kuva aina tilanteen, mistä testi lähtee liikkeelle. Esimerkiksi idoc-laskusanoman testitapauksien ensimmäisenä ehtona on, että SAP-järjestelmästä lähetetään lasku, joka sisältää vaadittavan tiedon. Seuraavissa askeleissa kuvataan toimenpiteet toiminnon testaamiseksi. Esimerkiksi toisessa askeleessa tarkistetaan, että vastaanotettava idoc-sanoma sisältää tarvittavan tiedon oikeassa segmentissä. Tämän jälkeen tarkastetaan operaattorin tekemä tiedon

poiminta oikeaan verkkolaskusanoman elementtiin. Teapps- ja Finvoice-formaateissa on määritelty tiedoille tietyt elementit, joten testitapauksessa kuvataan mistä tiedon voi tarkistaa ja mikä on kyseinen elementti, jossa tieto tulee olla. Jos nämä askeleet toteutuvat, on testitapaus suoritettu hyväksytyksi. Jos taas ei, testitapaus hylätään. Askeleiden kuvauksia varten kerättiin tietoa dokumenteista ja aiemmin tehdyistä muistiinpanoista. Askeleiden kuvaus on tärkeää, koska ne auttavat testin toistettavuudessa ja virhetilanteiden jäljitettävyydessä kun tiedetään tarkasti mitä toimenpiteitä on suoritettu ennen virheen ilmenemistä.

Perustestitapausten jälkeen katsottiin tarpeelliseksi vielä miettiä tarkentavia testejä, joiden suunnittelussa pyrittiin huomioimaan ekvivalenssiluokka-ajattelu, raja-arvot ja prosessin aikana tapahtuvat tilamuutokset sekä summauksiin liittyvät tarkistukset. Näitä testitapauksia tarvittiin mm. laskun summa-kenttään, alennuksiin, laskutuslisiin sekä tuoterivien määrään ja vapaateksti-kenttiin liittyen. Näiden osalta on olennaista tarkistaa, onko saapuvassa aineistossa laskennan kautta tulevat tiedot oikein, mikä vaikutus on merkkien tai rivien määrällä tai tietojen puuttumisella.

Suunniteltuja testitapauksien sisältöjä käytiin läpi useampaan kertaan, sisältöjä tarkastettiin ja tarkennettiin sitä mukaa kun ymmärrys prosessista ja asioiden yhteyksistä vielä kasvoi. Tarkennusten yhteydessä joitain testitapauksia myös poistettiin tarpeettomina, koska havaittiin, että toinen testitapaus kattaa jo saman toiminnon. Valmiit testitapaukset katselmoitiin, jolloin varmistuttiin siitä, että niihin kirjatut tiedot ovat oikein ja ymmärrettävät. Katselmoinnin jälkeen testitapauksia vielä täydennettiin ja sen jälkeen testitapaukset hyväksyttiin ennen testauksen aloitusta.

Hyväksytyt testitapaukset voitiin poimia testattavaksi HP ALM:n TestLab-osioon ensimmäistä testauskierrosta varten. Hyvin suunniteltujen testitapausten etu on siinä, että niitä voidaan hyödyntää eri tilanteissa. TestLab-osioon on mahdollista laatia useita eri testausvaiheita, joihin TestPlan-osioon perustettuja testitapauksia voidaan poimia. TestLab:ssä testitapaukset tullaan ajamaan suunniteltujen ohjeiden ja askeleiden mukaisesti. Jos askeleiden suoritusten aikana virheitä ei havaita, testitapaus katsotaan hyväksytyksi. Jos taas jonkun askeleen kohdalla havaitaan virheitä, testitapaus hylätään ja testitapaukseen kirjataan mikä oli todellinen lopputulos. Sen pohjalta kirjataan havainto, joka osoitetaan korjattavaksi siitä vastaavalle taholle.

Testitapausten suunnittelussa hyödynnettiin määrittelyihin perustuvaa testaustekniikkaa, harmaalaatikkomenetelmää ja osaltaan myös riskiperusteista menetelmää. Määrittelyihin perustuvaa testausta tehdään idoc-laskusanoman, laskun ulkoasuun ja liitteistykseen liittyvään testaukseen, koska ne haluttiin testata täsmällisesti ja varmistua siitä, että SAP-järjestelmään

tehdyt muutokset on tehty oikein. Menetelmän käytön puolesta puhui lisäksi se, että määrittelyt oli tehty selkeästi Excel-taulukkoon kenttä kentältä, jolloin yksittäisten kenttien testaus on ylipäätään mahdollista. Kenttien määrä ei myöskään ole kohtuuton siihen nähden paljonko testausaikaa oli suunniteltu käytettävän. Arviolta yhden testitapauksen suorittaminen kestää noin 4 minuuttia, kun tarkistettavat kentät on selkeästi kuvattu. Jos myöhemmin laskusanomaan tehdään muutoksia ja on vain tarve testata muuttuneet tiedot, kenttäkohtaiset testitapaukset ovat helppo ylläpitää ja ottaa testauksen käyttöön. Testitapausten määrä kasvoi suureksi, mutta käytännössä ne ovat nopeita suorittaa. Yhdellä peruslaskulla saadaan testattua useampi testitapaus, ja jos testitapauksia jää sen jälkeen vielä auki, voidaan avoimien testitapausten jälkeen pyytää vielä erikseen testiaineistoa, joka sisältää puuttuvat tiedot. Esimerkkejä testitapauksista on esitelty liitteessä 5. Testitapauksia muodostui yhteensä 132 kappaletta, joista suurin osa koski idoc-laskusanomaa ja laskun ulkoasun muodostamista. Testitapauksissa huomioitiin myös savutestaus sekä regressiotestaus.

7 Arviointi

Toiminnallisen tutkimuksen luotettavuuden arviointi perustuu siinä käytettyjen menetelmien, kerätyn tiedon ja tulosten tarkkaan dokumentaatioon. Opinnäytetyön tulosten luotettavuudesta ja käytettävyydestä varmistuttiin siten, että prosessin aikana työn edistymistä läpikäytiin projektin statuspalavereissa sekä operaattorin, että SAP-projektin asiantuntijoiden kanssa, ja tarkistettiin eri näkökulmien toteutuminen ja olemassa olevien dokumenttien ajantasaisuus sekä testauksen kattavuus. Testauksen kattavuuden osalta päädyttiin siihen, että testitapaukset tehdään hyvin tarkalla tasolla perustuen laskusanoman ja laskun ulkoasun määrittelyihin. Testitapauksia verrattiin operaattorin laatimiin testitapauksiin (kuvio 8). Niihin verrattuna opinnäytetyön tuotoksena laaditut testitapaukset olivat huomattavasti tarkemmalla tasolla ja kattavammat.

Case	Object	Test	Date	Result	Responsible	Excpected results
9	Routing: B2B – einvoice					Routing gives correct channel to invoice.
10	Routing: B2C – einvoice					Routing gives correct channel to invoice.
11	Routing: B2C – direct payment					Routing gives correct channel to invoice.
12	Routing: B2B – iPost					Routing gives correct channel to invoice.

Kuvio 8: Esimerkki operaattorin testitapauksista.

Sähköisen laskutuksen testaukseen luodut testitapaukset katselmoitiin syksyn aikana kahteen otteeseen. Ensimmäisellä kerralla arvioitiin testitapauksia otsikkotasolla, jotta nähtiin että testaukseen on poimittu oikeita asioita ja testitapauksia lähdetään rakentamaan sillä tasolla, että niiden kattavuutta voidaan arvioida ja niitä on mahdollista ylläpitää ja hyödyntää myös myöhemmin. Toisella katselmointikierroksella läpikäytiin testitapausten sisältöä tarkemmin ja hyväksyttiin käytettäväksi testauksessa.

Toimeksiantajan arvioinnin testitapausten laadukkuudesta ja hyödynnettävyydestä antoi 1.11.2017 SAP-projektin testauspäällikkö:

- Testit jaoteltu hyvin omiin loogisiin kokonaisuuksiinsa hakemistoittain
    - HP ALM:n testitapausten hakemistorakenteesta jo hahmottaa testauksen kokonaisuuden/laajuuden
- Testitapaukset palasteltu tarpeeksi pieniin osiin, vain muutamia testisteppejä per testitapaus.
    - Testitapausten katselmointi helpompaa ja helpompi hahmottaa, että kaikki tarvittava on suunniteltu testattavan.
    - Helpottaa testauksen edistymisen seurantaa/raportointia
    - Yksittäiset testit helpommin suoritettavissa valmiiksi (vähentää ”not completed” tilaisten testitapausten määrää).
- Testitapauksista käy selville, mihin vaatimukseen ne perustuvat.
- Nimeämiskäytäntö on selkeä
- Testitapaukset tarkalla teknisellä tasolla.
    - Asiaa tuntevalle testaajalle/toteuttajalle hyvä malli. Tekee testaajan ja toteuttajan keskinäisestä kommunikaatiosta sujuvaa ja keskittyen olennaiseen.
    - Täysin asiaa tuntematon testaaja voisi toisaalta olla vaikeuksissa, sillä testitapaukset eivät ohjeista testaajaa yksiselitteisesti miten testituloksiin päästään. Tämä on tosin testitapauksia suunniteltaessa keskusteltu ja sovittu, että tämä taso on parempi, sillä testitapausten suorittajat tuntevat prosessin hyvin.
- Testitapaukset on hyvin uudelleen käytettävissä ja niistä on helposti valittavissa tärkeimmät testitapaukset mm. tulevaisuudessa tarvittaviin regressiotestauksiin. Regressiotestisettiä luodessa tulee kuitenkin harkita, tulisiko testitapauksiin lisätä muutamia alustavia testisteppejä, joissa selvennetään testitapausten esiehtoja ja tarvittavia toimenpiteitä, miten testituloksiin päästään. Esim. syötetiedostojen luonti/käsittely/syöttö järjestelmään.

8 Yhteenveto ja johtopäätökset

Testauksella on merkittävä rooli ohjelmistokehityksessä. Testauksella pyritään varmistamaan se, että ohjelmisto vastaa liiketoiminnan asettamia vaatimuksia, ohjelmisto on laadukas, ja että se toimii luotettavasti ja virhetilanteista toipuminen on hallittua. Testauksen suunnittelu tulee aloittaa jo kehitysprojektin alkuvaiheessa, jotta määrittelyihin ei jäisi virheitä tai aukkoja, jotka havaitaan vasta ohjelmiston toteutusvaiheessa tai myöhäisessä testausvaiheessa. Testauksen suunnittelussa tulee käyttää mahdollisimman paljon saatavilla olevaa materiaalia ohjelmiston vaatimuksista, määrittelyistä ja käyttöön liittyvistä ohjeistuksista, mutta myös

riskien analysointi ja käyttäjien kokemukset ovat tärkeää informaatiota testauksen suunnitteluun.

Eri testaustekniikoiden avulla voidaan laatia testitapauksia, joiden avulla ohjelmiston eri toiminnot voidaan testata johdonmukaisesti ja tehokasti siten, että eri näkökulmat tulevat huomioiduiksi ja samoja asioita ei testata turhaan useaan kertaan. Testitapausten tarkoitus on kuvata testattava toiminnallisuus ja viitata siihen liittyvään vaatimukseen. Testitapauksessa kuvataan testaustoimenpiteen suorittaminen mahdollisimman kattavasti testaajan osaaminen ja tietämys aiheesta huomioiden siten, että toimenpiteiden jälkeen testaaja voi todeta onko toiminnallisuus kunnossa vai onko siinä vielä korjattavaa.

Kohdeyrityksen sähköisen laskutuksen testaus oli osa suurempaa IT-arkkitehtuurin muutoshanketta. Testauksen suunnitteluun vaikutti suuresti SAP-järjestelmän myyntilaskutuksen uudistamisesta muodostetut määrittelydokumentit, joiden sisältö oli erittäin tarkasti ja selkeästi laadittu. Testauksen suunnittelun yhteydessä dokumenteista löytyi kuitenkin vielä tarkentamista vaativia kohtia. Sähköisestä laskutuksesta oli lisäksi paljon erillisiä dokumentteja saatavilla, jotka kuvasivat prosessia hyvin kattavasti. Dokumenttien pohjalta laadittiin ensin käyttötapauksia, ja niiden pohjalta lähdettiin poimimaan toimintoja testitapauksiin. Haasteellista testitapausten laadinnassa oli ymmärtää erilaisten verkkolaskuformaattien vaikutus laskusanoman prosessointiin ja liitteistyksen poikkeava prosessi. Laskun ulkoasun ja laskusanoman sisällön testaus päätettiin tehdä suoraan määrittelyihin perustuen, koska näin voitiin varmistua testauksen hyvästä kattavuudesta ja siitä, että perusasiat ovat varmasti kunnossa. Testitapausten suunnittelussa pyrittiin huomioidaan myös erilaisten syötteiden vaikutukset. Haasteena tässä työssä olikin miettiä miten eri syötteillä testaus on mahdollista testata, koska testiaineisto tulee ns. valmiiksi annettuna erillisessä projektissa toteutettavasta SAP-järjestelmän testauksesta. Testilaskujen sisältöön ei siis ainakaan testauksen alkuvaiheessa ole mahdollista vaikuttaa. Tästä johtuen lyhyet kenttäkohtaiset testitapaukset katsottiin olevan hyvä ratkaisu. Testauksen edetessä on mahdollista tarkastella vielä avoimena olevia testitapauksia ja pyytää tarvittavia syötteitä sisältäviä testilaskuja.

Kehittämiskohteeksi voidaan esittää sähköisen laskutuksen dokumentaation yhtenäistämistä operaattorin osalta. Laskusanoman käsittelyyn liittyy paljon yksityiskohtaisia kohtia, joita ei ole kuvattu dokumenteissa kovin tarkasti, tai tiedot ovat hajallaan eri dokumenteissa, joiden ajantasaisuudesta ei ole varmuutta. Palvelun kuvaukseen liittyvät asiat olisikin hyvä koota yhteen dokumenttiin ja tarkistaa tietojen ajantasaisuus.

Laadittuja testitapauksia katselmoitiin SAP-projektin testauspäällikön kanssa kahteen otteeseen työn toteuttamisen aikana. Testauspäälliköllä oli omakohtaista vahvaa osaamista sähköi-

sestä laskutuksesta sekä yhteistyöstä operaattorin kanssa, joten hän tunsi testattavan kohteen erittäin hyvin ja pystyi arvioimaan testitapausten laadun ja hyödynnettävyyden. Arvioinnissa testauspäällikkö totesi, että testitapaukset on laadittu kattavasti ja sillä tasolla, mitä työn alkuvaiheessa suunniteltiin. Testitapauksia voidaan poimia sellaisenaan myös toisen SAP-järjestelmän sähköisen laskutuksen testaukseen. Tarvittavat muutokset testitapauksiin on helppo tehdä, koska testitapaukset on laadittu riittävän pieniin osiin ja nimeämiskäytännön perusteella löydetään helposti määrittelyriviä vastaava testitapaus. Laadittua testitapausten listaa verrattiin lisäksi operaattorin laatimaan karkealla tasolla olevaan testitapauslistaan ja sen perusteella voitiin todeta, että laaditut testitapaukset olivat kattavat ja huomioivat monipuolisesti sähköisen laskutuksen eri tilanteet.

Lähteet

Painetut

Black, R. 2009. Managing the Testing Process. 3rd edition. New York: John Wiley & Sons, Inc.

Farrell-Vinay, P. 2008. Manage Software Testing. New York: Auerbach Publications.

Haikala, I. & Märijärvi, J. 2004. Ohjelmistotuotanto. 10. painos. Helsinki: Talentum.

Hirsjärvi, S., Remes, P. & Sajavaara, P. 2005. Tutki ja kirjoita. 11. painos. Jyväskylä: Gummerus Kirjapaino Oy.

Kaner, C., Bach, J. & Pettichord, B. 2002. Lessons Learned in Software Testing – A Context - Driven Approach. New York: John Wiley & Sons, Inc.

Kaner, C. & Fiedler, R. L. 2016. Test Design: A BBST Workbook. Context Driven Press.

Kasurinen, J. P. 2013. Ohjelmistotestauksen käsikirja. Jyväskylä: Docendo.

Kananen, J. 2014. Toimintatutkimus kehittämistutkimuksen muotona. Miten kirjoitan toimintatutkimuksen opinnäytetyönä? Jyväskylä: Jyväskylän ammattikorkeakoulu

Ojasalo, K., Moilanen, T. & Ritalahti, J. 2015. Kehittämistyön menetelmät. Uudenlaista osaamista liiketoimintaan. Helsinki: SanomaPro Oy

Sähköiset

Abbas, R., Bhatti, S.N., Shah, S. A. A. & Sultan, Z. 2017. Analytical Review on Test Cases Prioritization Techniques: An Empirical Study. International Journal of Advanced Computer Science and Applications, Vol.8. No. 2, 2017. Viitattu 3.11.2017. https://thesai.org/Downloads/Volume8No2/Paper_39-Analytical_Review_on_Test_Cases_Prioritization_Techniques.pdf

Danske Bank. E-lasku ja suoramaksu. Viitattu 11.11.2017. https://danskebank.fi/fi-fi/henkiloasiakkaat/paivittaiset-raha-asiat/maksut/kotimaan-maksut/e-lasku/pages/e-lasku.aspx

Finanssiala ry. Finvoice soveltamisohje versio 2.01. 26.10.2015. Päivitetty 2.1.2017. Viitattu 11.11.2017. http://www.finanssiala.fi/finvoice/dokumentit/Finvoice_2_1_soveltamisohje.pdf

Finnish Software Testing Board. ISTQB:n testaussanasto v. 2.3 Suomi – Englanti. Viitattu 1.10.2017. http://www.fistb.fi/sites/fistb/files/liitteet/istqb_sanasto_2015-04-30%202.3%20FI-ENG.pdf

Hooda, I. & Chhillar, R. S.2015. Software Test Process, Testing Types and Techniques. International Journal of Computer Applications (0975 – 8887). Volume 111 – No 13. Viitattu 3.11.2017. http://research.ijcaonline.org/volume111/number13/pxc3901433.pdf

OpusCapita Oyj. 2017a. Kaikki mitä olet aina halunnut tietää iPostista. Viitattu 29.10.2017. http://www.ipost.fi/usein-kysyttya/

OpusCapita Oyj. 2017b. OpusCapita PEPPOL: usein kysyttyjä kysymyksiä. Viitattu 28.10.2017. https://www.opuscapita.fi/media/1919656/OpusCapita-PEPPOL-FAQ.pdf

Suomen Taloushallintoliitto ry. 16.3.2015. Verkkolaskun vähimmäistietosisältö käyttöön. Viitattu 28.10.2017.

https://taloushallintoliitto.fi/ajankohtaista/verkkolaskun-vahimmaistietosisalto-kayttoon-16032015

Tieto Finland Oy. 17.3.2011. TEAPPSXML v.2.7.2 soveltamisohje. Viitattu 11.11.2017. https://www.tieto.fi/sites/default/files/migrated/documents/TEAPPSXML_ohje_v.2.7.2.pdf

Tieto Finland Oy. 9.6.2017. TEAPPSXML v.3.0 tulossa syksyllä 2017. Viitattu 28.10.2017. https://www.tieto.fi/sites/default/files/atoms/files/teappsxml_v.3.0_ennakkotiedote.pdf

Tieto Finland Oy. 2.3.2016. Verkkolaskun tietosisältöön huomiota. Viitattu 28.10.2017. https://www.tieto.fi/sites/default/files/atoms/files/verkkolaskun_tietosisaltoon_huomioita_02032016.pdf

Tietoyhteiskunnan Kehittämiskeskus ry. 2017. Verkkolaskusanasto. Viitattu 29.10.2017. https://www.tieke.fi/display/verkkolasku/Verkkolaskusanasto

Verkkolaskun kehitysnäkymät ja eOsoite. 23.5.2017. Tilisanomat. Viitattu 28.10.2017. https://tilisanomat.fi/artikkeli/verkkolaskun-kehitysnakymat-ja-eosoite

Kuviot

Liitteet

Liite 1: Sähköinen laskutus - Lähtevä laskutus tukipyynnöt 1.1.2016-25.10.2017

Aihe	Tikettien määrä
Muuttunut laskutusosoite	441
Alv-erittely puutteellinen	25
Tuplalaskutus	8
Free text liian pitkä	8
Liitetiedosto virheellinen tai tyhjä	7
Aineistossa merkkivirhe	7
Päiväraportti puuttuu	6
Liitteen lähetysosoite virheellinen	3
RI-sanoman välittyminen	3
Päiväraportin tietosisältö	2
Digitaalinen allekirjoitus puuttuu	2
Virheellinen merkki aineistossa	2
Liite puuttuu	2
Liittymävirhe	2
Aineisto käsitelty kaksi kertaa	2
Asiakkaalla poikkeava maksunsaaja	1
RF-viitteen paikka verkkolaskusanomalla	1
Laskustuskanavan muutos	1
Finvoice ja liite	1
Ajastusmuutos	1
Operaattorilla lähettäjätunnus ei kunnossa	1
Kaikki yhteensä	**526**

Liite 2: Testauksen suunnittelun muistiinpanot OneNote-sovelluksessa.

- Suunnitelmaa
  - Aikataulu
  - Avoimet asiat
  - Projektin tilanneseuranta
  - Yhteyshenkilöt
  - Testauksen suunnitelma
    - Testitapauksiin poimitut
    - Testiympäristöt
  - Statuspalaverit
  - Ohjausryhmä
- Muutosprojektin määrittelyt
  - Lasku layout
  - I-00237 Interface to Service provider
    - Alustava määrittelydokumentaatio arviointi
    - Kommentteja määrittelyihin
    - Func. Spekseistä puuttuvat tiedot
  - Liitteellisten laskujen kehitys
    - Liitteistyspalaverit
    - Laskutuskuitin jakaminen
  - ALV-käsittelyn palaverit
  - Operaattorin työmääräarvio
    - Operaattorin tarkennuksia määrittelyihin
- Liittymät
  - Liittymäpalaverit
  - Hakemistot
  - Liittymäaineiston nimeäminen
  - S2 Liittymädokumentti
- Idoc Mäppäystä
  - Verkkolaskusanoma/idoc - palaverit
  - Rivitason summauksista
  - Alvkoodi mäppäys
  - Y-materials alv-laskenta
  - Tositteiden summaus
- Logomäärittelyt
  - Laskun logosta ja tulostuksesta - palaveri 11.8.
  - Logo-speksit
- Laskukanavan päättely
  - Laskukanava kuvaus
  - Laskukanava Masterdatassa
    - Laskukanava general / company code levelillä
  - Laskukanava laskulajeille
- Korkolaskut verkkolaskuina –kuvaus
- Kuluttajan elasku
  - RI-validation
- Teapps-formaatti
- Finvoice erikoisuudet
- Muutoksia InExchangelle menevissä laskuissa
- Virhekäsittelyt / valvonta
- Raportointi
  - S2 raporttimäärittelyt
  - SAP queryt
- HP ALM

Liite 3: Testaukseen poimitut toiminnallisuudet.

Testikokonaisuus	Testattava toiminto	Jäljitettävyys
1 Liittymät (Data transfer)		
	Maksumuistutusten välitys	I-01351 Functional specifikation
	Liitteettömien laskujen välitys	I-01352 Functional specifikation
	X-järjestelmän liitteellisten laskujen välitys	I-01353 Functional specifikation
	SAP-järjestelmän liitteellisten laskujen välitys	I-01371 Functional specifikation
	Liitteiden välitys	I-01370 Functional specifikation
	Tiedostojen nimeämiskäytännöt	I-01352 Functional specifikation
	Ajastukset	Sähköisen laskutuksen ohje.doc
	Virhehallinta	Riskiperusteinen
2 Liitteistys (Attachments)		
	Liitteen käsittely, jos asiakkaan operaattorina pankki	Yleistoiminnallisuus
	X-järjestelmän liitteet	Liitteelliset X laskut - Toiminnallinen kuvaus.pdf
	SAP-järjestelmän liitteet	CR18 määrittelydokumentti
	Virhehallinta	Riskiperusteinen
3 Idoc-laskusanoma (Idoc mapping)		
	Idoc-sanoman tietojen oikeellisuus	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906
	Laskun perustiedot, mäppäys Teapps ja Finvoice	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906, mci-35-invoice-mapping-X-xml-X
	Otsikkotason tiedot, mäppäys Teapps ja Finvoice	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906, mci-35-invoice-mapping-X-xml-X
	Rivitason tiedot, mäppäys Teapps ja Finvoice	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906, mci-35-invoice-mapping-X-xml-X
	Yhteenvedon tiedot, mäppäys Teapps ja Finvoice	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906, mci-35-invoice-mapping-X-xml-X
	Alatunnisteen tiedot, mäppäys Teapps ja Finvoice	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906, mci-35-invoice-mapping-X-xml-X
4 Laskun ulkoasu (Layout)		
	Otsikkotason tiedot	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Rivitason tiedot	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Yhteenvedon tiedot	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Alatunnisteen tiedot	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Verkkolaskun ulkoasu	Riskiperusteinen
	Paperilaskun ulkoasu	Riskiperusteinen
	Kielikäännökset	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Monisivuinen lasku	M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915
	Vertaus SAP-laskun kuvaan	Riskiperusteinen
5 Reititys (Channeling)		
	B2B verkkolasku	X Solution Document_ver_0 9.pdf
	B2B paperilasku	X Solution Document_ver_0 9.pdf
	B2C suoramaksu	X Solution Document_ver_0 9.pdf
	B2C elasku	X Solution Document_ver_0 9.pdf
	B2C paperilasku	X Solution Document_ver_0 9.pdf
	Netposti	X Solution Document_ver_0 9.pdf
	Maksumuistutukset paperilaskuna	X Solution Document_ver_0 9.pdf
	Korkolasku kun asiakkaan operaattori on pankki	X Solution Document_ver_0 9.pdf
	Virhehallinta	Riskiperusteinen
6 Verkkolaskuformaattien erityispiirteet (Data formats)		
	TEAPPSXML	Riskiperusteinen
	Finvoice	Riskiperusteinen
	PeppolBIS	Riskiperusteinen
	UBL/Svefaktura	Riskiperusteinen
	Alv-erittelyt	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906
7 Laskutyyppien erityspiirteet (Invoice types)		
	Hyvityslasku	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906
	Korkolasku	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906
	Maksumuistutus	I-00237 I-01351 I-01352 Billing from SAP to Service provider MS_NEW_20170906
8 RI- ja SI-sanomat (RI- and SI-messages)		
	Kuluttajan elasku- tai suoramaksuilmoituksen vastaanotto	Sähköisen laskutuksen ohje.doc
9 Raportointi (Reporting)		
	Välitetyt laskut	TAL-laskutus ja laskuliikenteen valvonta ja tilastointi
	Käsitellyt laskut	TAL-laskutus ja laskuliikenteen valvonta ja tilastointi
	Virheilmoitukset	Sähköisen laskutuksen ohje.doc
10 Suorituskyky (Performance test)		
	Suuri tiedostokoko	Riskiperusteinen
	Useita idoc-laskusanomia	Riskiperusteinen

Liite 4: Esimerkki käyttötapauksista.

1. Käyttötapaus	**Kuluttaja- ja yritysasiakkaan paperilasku, kieli suomi**
Tarkoitus	Lähetetään asiakkaalle lasku, asiakastiedoissa laskutuskanavana paperilasku ja kielenä suomi
Esiehto	Laskutusasiakkaaksi on valittu asiakas, jonka asiakastiedoissa on laskutuskanavana 3 (paperilasku) ja kieli FI (suomi)
1. Aloitus	SAP-järjestelmässä myyntitilauksesta muodostuu laskusanoma liittymään.
2. Liittymä	Laskusanoma välittyy operaattorille
3. Validointi	Operaattori validoi laskusanoman
4. Laskukanavan tunnistaminen	Operaattori tulkitsee laskulta laskutuskanavan 3 (paperilasku)
5. Osoitetietojen poiminta	Operaattori muodostaa asiakkaan osoitetiedot xml-tiedostoon laskun tulostusta ja postitusta varten
6. Asiakkaan kielen valinta	Operaattori tunnistaa laskusanomalta asiakkaan kielen (FI)
7. Laskun muodostus	Operaattori muodostaa laskusta pdf-muotoisen laskun kuvan suomenkielisen lomakepohjan mukaisesti
8. Reititys	Operaattori välittää laskun kuvan ja osoitetiedot tulostuspalveluun
Poikkeustapaukset	
2.A Laskusanoma ei välity operaattorille	Aineistossa on virhe, joka tulee korjata ja lähettää korjattu aineisto uudelleen
3.A Laskusanomasta puuttuu tietoja ja aineistoa ei voida käsitellä	Aineistossa on virhe, joka tulee korjata ja lähettää korjattu aineisto uudelleen
5.A Osoitetiedoissa on erikoismerkkejä, jotka estävät tiedoston muodostamisen	Aineistossa on virhe, joka tulee korjata ja lähettää korjattu aineisto uudelleen
Vastaavat käyttötapaukset myös kielistä ruotsi, englanti, joilla on oma laskulomake.	

2. Käyttötapaus	**Kuluttaja-asiakkaan elasku, kieli suomi**
Tarkoitus	Lähetetään henkilöasiakkaalle lasku, asiakastiedoissa laskutuskanavana elasku ja kielenä suomi
Esiehto	Laskutusasiakkaaksi on valittu henkilöasiakas, jonka asiakastiedoissa on laskutuskanavana 2 (kuluttajan elasku), verkkolaskuosoite ja kieli FI (suomi)
1. Aloitus	SAP-järjestelmässä myyntitilauksesta muodostuu laskusanoma liittymään.
2. Liittymä	Laskusanoma välittyy operaattorille
3. Validointi	Operaattori validoi laskusanoman
4. Reititys	Operaattori etsii osoitteistostaan reitin asiakkaan verkkolaskuosoitteelle, joka poimitaan laskusanomalta
5. Laskukanavan tunnistaminen	Operaattori tulkitsee laskulta laskutuskanavan 2 (kuluttajan elasku)
6. Asiakkaan kielen valinta	Operaattori tunnistaa laskusanomalta asiakkaan kielen (FI)
7. Laskusanoman muodostaminen	Operaattori muodostaa asiakkaan vastaanottaman laskusanoman (Finvoice) sen mukaan mikä kielivalinta asiakkaalla löytyi
8. Välitys	Operaattori välittää laskusanoman asiakkaan operaattorille (pankkiverkko)
Poikkeustapaukset	
2.A Laskusanoma ei välity operaattorille	Aineistossa on virhe, joka tulee korjata ja lähettää korjattu aineisto uudelleen
3.A Laskusanomasta puuttuu tietoja ja aineistoa ei voida käsitellä	Aineistossa on virhe, joka tulee korjata ja lähettää korjattu aineisto uudelleen
4.A Asiakkaan verkkolaskuosoitteelle ei löydy reittiä	Asiakkaan elaskua ei voida välittää. Hylkäyksestä lähtee lähettäjälle sähköposti-ilmoitus ja lasku ohjautuu tulostuspalveluun (jatkuu käyttötapauksen 1 kohdasta 5).
8.A Asiakkaan verkkolaskuosoite ei ole enää voimassa	Asiakkaan operaattori hylkää välitetyn laskusanoman. Hylkäyksestä lähtee lähettäjälle sähköposti-ilmoitus ja lasku ohjautuu tulostuspalveluun (jatkuu käyttötapauksen 1 kohdasta 5).

Liite 5: Esimerkkejä luoduista testitapauksista

Test ID : 4276 - 00_I-01352 Invoices from SAP to Invoice sending Operator

Field Label	Field Value	Field Label	Field Value
Test Name	00_I-01352 Invoices from SAP to Invoice sending Operator	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	10/21/17
Execution Status	Not Completed	Subject	00_Smoke test

Description
This test case confirms that connection SAP-EAI-Operator is working and the data is in correct format so that Operator is able to create the invoice. The process is descriped in the document: Interface I-00237 I-01351 I-01352 I-01353 Billing from YYY ERP SAP to Service provider v1.7

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send invoice from SAP including all mandatory fields. Mandatory fields are descriped in the document *Invoice Sending Technical Specification FiNo ENG 18.08.2017.pdf, pages 6-7*	SAP-team is able to create the invoice with basic data and line item information
Step 2	SAP sends billing data to EAI.	Data is in correct format (idoc flat file)
Step 3	EAI receives INVOIC iDoc, One INVOIC message contains one billing document but the messages are collected into batches.	No changes in data
Step 4	EAI processes files automatically and sends them to Operator (/interfaces/TEST/STT/elasku/out). Format should be BIN.	No changes in data, only delivery
Step 5	Operator receives billing data to /out/SAPTinvoic/data	Test data is delivered to correct server/folder: /out/SAPTinvoic/data. Data is in correct format (idoc flat file)
Step 6	Document prosessing is possible	Data validation passes and document is found from Operator Console in pdf, teapps and finvoice format and include information from idoc file.

Test ID : 2694 - 01_I-01370 Attachment from SAP to Operator

Field Label	Field Value	Field Label	Field Value
Test Name	01_I-01370 Attachment from SAP to Operator	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	9/14/17
Execution Status	No Run	Subject	01_Data transfer

Description
The process is descriped in the document: Interface I-00237 I-01351 I-01352 I-01353 Billing from YYY ERP SAP to Service provider v1.7

Design Steps

Step Name	Description	Expected
Step 1	In SAP there is included attachment to the salesorder. Ask SAP-team to send invoice which include the attachment information.	The attachment from the salesorder should be picked up to EAI in same time than invoice data. Attachment should be in separate file, not included to invoice file.
Step 2	SAP sends attachment files to EAI, format is pdf.	Data is in correct format
Step 3	EAI sends files to Operator (/interfaces/TEST/STT/eliite/out)	No changes in data, only delivery
Step 4	Operator receives attachments to out/SAPTinvoic/data	Test data is delivered to correct server/folder: out/SAPTinvoic/data Data is in correct format and Operator is able to process it.

Test ID : 4534 - 03_I-01351 Data transfer problem special character

Field Label	Field Value	Field Label	Field Value
Test Name	03_I-01351 Data transfer problem special character	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	11/6/17
Execution Status	No Run	Subject	01_Data transfer

Description
There might be problems while reminders are processed in Operator service, there should be monitoring for caughting errors. The notification is sent and the data can be fixed and resend. Special characters are usually from customer name and address copied from some webpage to the Master Data.

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send reminders to EAI, the customer name or address should include special character like arrow (->).	
Step 2	SAP sends Reminders in pdf-format to EAI	Data is in correct format (pdf)
Step 3	EAI receives pdf-files, processes them and sends them to Operator. (/interfaces/TEST/STT/ekirje/out)	No changes in data, only delivery
Step 4	Operator receives reminders to /out/SAPTinvoic/data but is not able to process the xml creation	Test data is delivered to correct server/folder: /out/SAPTinvoic/data Data is in correct format (pdf) but Operator is not able to process xml-file for customer name and address information which is needed for printing service.
Step 5	The data should not include special characters or null-values which are not allowed, Operator cannot process the data	Operator is able to notice the error by monitoring. The incorrect idoc is possible to remove. Correct idoc data is resend from SAP and it is possible to process.

Test ID : 1181 - 05_I-01352 Double invoice checking

Field Label	Field Value	Field Label	Field Value
Test Name	05_I-01352 Double invoice checking	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	7/12/17
Execution Status	No Run	Subject	01_Data transfer

Description
Operator rejects the invoice if there is already invoice with same invoicenumber in Invoice Sending service.

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send invoice from SAP including all mandatory field. Mandatory fields are descriped in the document *Invoice Sending Technical Specification FiNo ENG 18.08.2017.pdf, pages 6-7.* Pick up the invoice number for testing double invoicing.	
Step 2	SAP sends billing data to EAI	Data is in correct format (idoc flat file)
Step 3	Operator receives billing data to /out/SAPTinvoic/data,	Test data is delivered to correct server/folder: /out/SAPTinvoic/data. Data is in correct format (idoc flat file) and Operator is able to process it.
Step 4	Document prosessing is possible.	Document is found from Operator Application
Step 5	Ask SAP-team to resend the same invoice from SAP, but don't delete the original invoice from Application.	Resending the invoice is possible from SAP,
Step 6	SAP sends billing data to EAI.	Data is in correct format (idoc flat file) and EAI is processing the data to Operator.
Step 7	Operator receives billing data to /out/SAPTinvoic/data,	Operator rejects the invoice if there is already invoice with same invoicenumber in Application. The emal notification is sent SLP team.
Step 8	Delete the original invoice from Application.	Deleting is possible from Application.
Step 9	Ask SAP-team to resend the invoice from SAP.	Resending the invoice is possible.
Step 10	SAP sends billing data to EAI	Data is in correct format (idoc flat file) and EAI is processing the data to Operator.
Step 11	Operator receives billing data to /out/SAPTinvoic/data,	Test data is delivered to correct server/folder: Data is in correct format (idoc flat file) and Operator is able to process it because the original invoice is deleted from Application.

Test ID : 1216 - 03_002_Invoicetype

Field Label	Field Value	Field Label	Field Value
Test Name	03_002_Invoicetype	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	7/21/17
Execution Status	No Run	Subject	03_Idoc mapping

Description
The precondition is that SAP is able to send invoices through EAI to Operator. Invoices should include the information mentioned in testcase headline. SAP-field 2 Invoice type (Invoice, Interest invoice, Credit invoice). Specification document: I-00237 I-01351 I-01352 Billing from YYY ERP and SAP to Service Provider MS_NEW_20170906.xlsx.

Design Steps

Step Name	Description	Expected
Step 1	Operator receives the normal invoice, with mandatory fields and total sum is positive (for example 1 eur).	The invoice is processed succefully.
Step 2	Check the idoc file	The idoc file includes the invoice type information
Step 3	Check mapping to Teapps from Operator Application. This is header information , invoicetype defines if the invoice is Invoice, Credit Invoice, Interest Invoice, Reminder	In Teapps element /HEADER/INVOICE_TYPE should be „00" which is Invoice
Step 4	Check mapping to Finvoice from Operator Application. This is header information , invoicetype defines if the invoice is Invoice, Credit Invoice, Interest Invoice, ReminderReminder	In Finvoice element InvoiceDetails/InvoiceTypeCode should be „INV01" which is Invoice

Test ID : 1218 - 03_029_Invoice Date

Field Label	Field Value	Field Label	Field Value
Test Name	03_029_Invoice Date	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	7/21/17
Execution Status	No Run	Subject	03_Idoc mapping

Description
The precondition is that SAP is able to send invoices through EAI to Operator. Invoices should include the information mentioned in testcase headline. SAP-field 29 Invoice date, this is Date when invoice was issued. Specification document: I-00237 I-01351 I-01352 Billing from YYY ERP and SAP to Service Provider MS_NEW_20170906.xlsx.

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send invoice from SAP including all mandatory fields. Mandatory fields are descriped in the document *Invoice Sending Technical Specification FiNo ENG 18.08.2017.pdf, pages 6-7*	SAP-team is able to create the invoice with basic data and line item information. The Invoice date is mandatory field. The invoice include the invoice date information from SAP Table VBRK, field FKDAT
Step 2	Operator receives the invoice.	The invoice is processed succefully and the invoice is found from Application.
Step 3	Check idoc file	The information is in idoc segment E2EDK02, field datum, qua. 009
Step 4	Check mapping to Teapps from Application	The invoice date is mapped to Teapps element /HEADER/INVOICE_DATE/DATE
Step 5	Check mapping to Finvoice from Application	The invoice date is mapped to Finvoice element InvoiceDetails/InvoiceDate
Step 6	Check that invoice in Application includes same information than invoice printed directly from SAP	Information should be same.

Test ID : 4263 - 04_053-54_Invoice total

Field Label	Field Value	Field Label	Field Value
Test Name	04_053-54_Invoice total	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	10/21/17
Execution Status	No Run	Subject	04_Layout

Description
The precondition is that SAP is able to send invoices through EAI to Operator and Operator is able to create the invoice layout from idoc. Invoices should include the information mentioned in testcase headline. This test case is for checking invoice layout created by service provider Operator. Follow the document M04NNF0001_PrintoutSpecification_Sales Billing Document Printout_v2_20170915

Design Steps

Step Name	Description	Expected
Step 1	Operator receives an invoice data in idoc format from SAP and process it. The total sum should be the maximum length which is possible to have in SAP SD (for example 9.999.999.999,99 euros). Invoice should have the language code FI.	Operator is able to process the idoc and creates invoice layout which is found from Application
Step 2	Check the layout from Application: Headline in position should be „Laskun loppusumma". The headline is language specific , Font is Calibri 10 Bold.	Headline is in correct position (53) and translation is correct (Laslun loppusumma). Font is Calibri 10 Bold.
Step 3	Check the layout from Application: Invoice total with tax 9.999.999.999,99 fits to the field in position 54. Font: Calibri 11 Normal.	Invoice total with tax fits to layout in one row in position 54.. Font: Calibri 11 Normal.

Test ID : 1186 - 05_003_B2B and B2C Printed on paper_economy class

Field Label	Field Value	Field Label	Field Value
Test Name	05_003_B2B and B2C Printed on paper_economy class	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	7/12/17
Execution Status	No Run	Subject	05_Channeling

Description
The precondition is that SAP is able to send invoices through EAI to Operator and Operator is able to create the xml-file from idoc message for customer name and address and invoice layout as pdf from idoc. A user-defined Identification number and ID Type field called Communication method (laskukanava) in Business partner master data defines how the IDoc message is processed in the service provider. If the value for ID Type is '3' then the service provider deliver the file to the printing service and the invoice is sent to the customer on paper. Same process to Company customers (B2B) and Consumer Customers (B2C).

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send invoice from SAP including all mandatory fields. Mandatory fields are descriped in the document *Invoice Sending Technical Specification FiNo ENG 18.08.2017.pdf, pages 6-7* and ask to use customer which have the communication method 3 Paperilasku in Master Data. For example customer number 1657542.	SAP-team is able to create the invoice with basic data and line item information. The customer's communication method is 3 Paperilasku.
Step 2	Operator receives the invoice and the idoc include the information of communication method.	In idoc file, there should be communication method 3.
Step 3	Operator process the invoice and route it to printing service. Check the routing from Application	In Application there should be information that invoice has delivered to iPost printing service
Step 4	Check that the invoice is printed to Economy class (this information need to be asked from Posti)	The invoice is printed to Economy class

Test ID : 1743 - 06_005_UBL-format

Field Label	Field Value	Field Label	Field Value
Test Name	06_005_UBL-format	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	8/28/17
Execution Status	No Run	Subject	06_Data formats

Description
The precondition is that SAP is able to send invoices through EAI to Operator and Operator is able to create the invoice in UBL-format format from idoc. UBL needs specific VAT information. This is basic test case for checking that invoices in UBL format contain relevant data from data mapping.

Design Steps

Step Name	Description	Expected
Step 1	The invoice is delivered to a customer (134566) in UBL-format, check that there's no errors in validation. Invoice is including invoice fee.	UBL format contain relevant data from data mapping. The invoice is delivered to a customer UBL-format and there's no errors in validation
Step 2	Invoice B2B UBL VAT-calculation rules: rows and summary must match including invoice fee.	Invoice B2B UBL VAT-calculation rules, rows and summary matches including invoice fee. UBL format calculation rule validation is passed

Test ID : 2111 - 09_02_Delivered invoices_Daily report

Field Label	Field Value	Field Label	Field Value
Test Name	09_02_Delivered invoices_Daily report	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	9/1/17
Execution Status	No Run	Subject	09_Reporting

Description
The precondition is that SAP is able to send invoices through EAI to Operator and Operator is able to create the invoices and deliver them to customers. This report is checking that Operator has processed all invoices towards customers which are sent from SAP. Dailyreports are delivered from Operator to the customer's receiving server Report: Lähtevät laskut Filename: Paivaraportti-dd-mm-yyyy_nnnnnnnn.csv From: //sapftp/in/sapOpirep/data (production) To: \\80011000srXXX\BWRoot\rata Scheduling: Daily at 00.15 am This report includes now Invoices and Interest invoices (einvoices or iPost) Based on document: TAL-laskutus ja laskuliikenteen valvonta ja tilastointi Tekninen dokumentaatio

Design Steps

Step Name	Description	Expected
Step 1	Invoices are delivered from SAP to Operator	Invoices are delivered to Operator
Step 2	Operator has processed invoices to customers during the same day	Invoices are delivered to customers
Step 3	Operator creates the report in csv, information included to report is defined in the document Solution Document_ver_0 9, slide 32	Information defined in the solution document is included to the csv file in separate columns.
Step 4	The file is picked up via FTP from: //sapftp/in/sapOpirep/data daily at 00.15 am. EAI is responsible for transfering the file.	The file is picked up from //sapftp/in/sapOpirep/data.
Step 5	The file is delivered to \\80011000srxxx\BWRoot\rata.	Report is found in \\80011000srXXX\BWRoot\rata.
Step 6	The copy of the file is delivered to \\80011000srxxx\BWRoot\rata\Archive.	Report is found in \\80011000srXXX\BWRoot\rata\Archive.
Step 7	Check the file and the content, compare invoices which were delivered day before from SAP (information is found from table xx_amount_matching or from the SAP-team)	The content of the report is what is defined and includes invoices and interest invoices delivered from SAP day before.

Test ID : 2375 - 10_I-01352 Large invoice

Field Label	Field Value	Field Label	Field Value
Test Name	10_I-01352 Large invoice	Type	MANUAL
Designer	sg.matmarj(Mattola Marjo)	Creation Date	9/8/17
Execution Status	No Run	Subject	10_Performance test

Description
Testing with batch which include 5 invoices with over 300 pages (Case XXX-XXX)

Design Steps

Step Name	Description	Expected
Step 1	Ask SAP-team to send 5 invoice from SAP. These invoice data should copy from productin Case XXX-XXX, item lines should include also ean-codes.	SAP-team is able to create the invoice with basic data and line item information
Step 2	SAP sends billing data to EAI. One INVOIC message contains one billing document but the messages are collected into batches.	Data is in correct format (idoc flat file)
Step 3	Check from SAP-team that files are named: SAPINVOIC.SAP**STT**01. STT = SAP test environment	Files from SAP and YYY ERP are named differently. SAP files named: SAPINVOIC.SAP**STT**01. STT = SAP test environment
Step 4	EAI receives INVOIC iDoc, processes them and sends them to Operator (/interfaces/TEST/STT/elasku/out). Format should be BIN.	No changes in data, only delivery
Step 5	Operator receives billing data to /out/SAPTinvoic/data, process them. The maximum time for processing should be 10 minutes	Test data is delivered to correct server/folder: /out/SAPTinvoic/data. Data is in correct format (idoc flat file) and Operator is able to process the invoices in 10 minutes.
Step 6	Document prosessing is possible	Documents are found from Operator Application