Haaga-Helia
ammattikorkeakoulu Oy

Projektin tapaustutkimus: Integraatio ERP-järjestelmäuudistuksessa

Pekko Tuomisto

Opinnäytetyö
Tietojenkäsittelyn
koulutusohjelma
2016

Tiivistelmä

Haaga-Helia ammattikorkeakoulu Oy

Tekijä
Pekko Tuomisto

Koulutusohjelma
Tietojenkäsittely

Raportin/Opinnäytetyön nimi
Projektin tapaustutkimus: Integraatio ERP-järjestelmäuudistuksessa

Sivu- ja liitesivumäärä
39 + 26

Opinnäytetyö on tapaustutkimus integraatiosta kahden eri aikakauden toiminnanohjausjärjestelmän välillä. Alkuperäinen järjestelmä on 80-luvulta, ja uusi on moderni avoimen lähdekoodin web-pohjainen järjestelmä. Integraatioprojekti oli osa laajempaa asiakasprojektia, mutta tämä tutkimus ei käsittele sitä millään tavoin, vaan on rajattu vain integraatioprojektiin.

Tutkimuksen tavoitteena on käydä läpi projektin aikaiset tapahtumat ja niitä analysoimalla selvittää hyvät ja huonot toimintatavat, sekä miten toimintatavat muuttuivat projektin aikana. Tutkimuslajina on tapaustutkimus, joka soveltuu hyvin tosielämän tapahtumien analysointiin. Tietolähteinä on käytetty alan kirjallisuutta ja tutkimuksia sekä integraatioprojektin aikana tehtyjä dokumentteja, tuntikirjauksia ja haastatteluja.

Tutkimus on jaoteltu projektin vaiheiden mukaan, joita ovat projektin aloitus, ohjelmointityön aloitus, projektin pitkittyminen ja kuukausittainen kehitys. Projektin vaiheet selvisivät tuntikirjauksia tutkimalla, ja jokaisella vaiheella on omat ominaispiirteensä. Kaikki vaiheet käydään läpi samalla tavalla: ensin kerrotaan mitä vaiheen aikana tehtiin, esitellään esimerkein erikoistapauksia ja lopuksi kerätään vaiheen päätelmät ja havainnot.

Vaiheiden läpikäynnin jälkeen kaikki havainnot kootaan yhteen ja niistä kehitetään loppupohdinnat ja -päätelmät sekä vastataan tutkimuskysymyksiin. Ensimmäiseen kysymykseen, mitkä olivat kriittisimmät virheet projektin aikana, löytyi kolme päävirhettä: väärät oletukset tietomallien yhdistämisestä ja käyttäjien tarpeiden riittämätön selvitys; sisään tulevan tiedon heikko laatu; sekä projektin etenemisen vaatimien töiden viivytykset.

Toinen tutkimuskysymys oli miten virheisiin reagoitiin ja olisiko ne voitu estää. Ensimmäiseen virheeseen reagoitiin käyttämällä aitoja tulosteita molemmista järjestelmistä ja kysyttiin käyttäjiltä mitä näistä tiedoista he halusivat näkyville. Vanhan järjestelmän tulosteilta selvitettiin mistä jokainen tiedonpala haetaan. Tällä tavoin kehitysvirheitä tuli huomattavasti vähemmän.

Toiseen virheeseen reagoitiin lisäämällä sisään tulevan tiedon tarkastuksia. Sisään tuleville tiedostot olisi pitänyt ajaa XML-tarkastajan läpi ennen hyväksyntää, mutta tämä olisi saattanut hidastaa uuden tiedon saantia ennestään.

Kolmanteen virheeseen reagoitiin huonosti, joka kostautui projektin pitkittymisenä. Tietoja olisi pitänyt pyytää voimakkaammin ja projektin työntekijöiden työtaakat olisi pitänyt varmistaa paremmin.

Asiasanat
Integraatio, toiminnanohjaus, tonni, odoo, zato, tapaustutkimus, integraatioprojekti

Abstract

Haaga-Helia ammattikorkeakoulu Oy

Author
Pekko Tuomisto

Program
Business Information Technology

Thesis title
Project Case Study: Integration in an ERP-system upgrade

Number of pages and appendices
39 + 26

This thesis is a case study of an integration between two ERP-systems from different eras. The original system is from the 80's and the new one is a modern open source web-based system. The integration project was part of a larger project for the client but this study does not examine it in any way, rather the scope is narrowed down to only concentrate on the integration project itself.

The aim of this study is to go through the project and by analyzing the events, find out the good and bad methods used and how the methods changed during the project. The study method is a case study, which suits well in analyzing real life events. The sources of information are industry literature and other scientific studies as well as documents, timesheets and interviews from the project.

The study is sectioned by the project stages which are: Project Initialization, Start of Programming, Project Delays and Monthly Development. These stages were found by analyzing the timesheets and each stage has particular features. Each stage is analyzed the same way: first elaborate what was done, demonstrate with examples special cases and finally gather the observations and conclusions.

After going through the stages all the observations are collected together and developed into the final analyses and conclusions, followed by answering the research questions. The first question, what were the most critical errors during the project, gave three errors: wrong assumptions about joining the data models and insufficient knowledge of user requirements; poor quality of incoming data; and delays in critical development.

The second research question was how were the errors reacted to and could they have been prevented? The reactions to the first error were to use genuine printouts from both systems and asking the users what data they needed visible. In addition, the origin of each piece of information from the old system's printouts was documented. This action was shown to decrease the amount of mistakes done in development.

The reaction to the second error was to increase the amount of data checks for incoming data. The incoming files should have been run through an XML-validator, but this could have further slowed down the rate of getting new data.

The third error got a meager reaction, which played havoc on the project schedule. The data inquiries should have been asked for more forcefully and the workload of the project developers should have been checked more thoroughly.

Keywords
Integration, Enterprise Resource Planning, Tonni, Odoo, Zato, Case Study, Integration project

Sisällys

1 Johdanto

IT-järjestelmät päivittyvät ja muuttuvat jatkuvasti. Uudet järjestelmät tuovat mukanaan uusia toimintoja ja parantavat vanhoja. Valitettavasti järjestelmäversiosta toiseen, saati järjestelmästä toiseen, siirtyminen vie paljon aikaa ja työtunteja niin käyttäjiltä kuin järjestelmän ylläpitäjiltä. Tämän takia eri organisaatiot usein pitkittävät tai jättävät tekemättä näitä omalle toiminnalleen hyödyllisiä järjestelmäpäivityksiä.

Erityisesti silloin, jos samaa järjestelmää on käytetty jo vuosia, tai jopa vuosikymmeniä, siirtyminen uuteen järjestelmään on mahdotonta ilman työn hidastumista ja jopa työkatkoja. Tämän takia siirtymäaikana halutaan pitää vanha järjestelmä käytössä samalla kun uutta järjestelmää opitaan ja otetaan käyttöön toiminnallisuus tai komponentti kerrallaan. Mutta kuinka pidetään huoli siitä, että molemmissa järjestelmissä on samat tiedot käytössä?

Tämä tutkimuksen aiheena oleva projekti käynnistettiin ratkaisemaan juuri tämä ongelma. Asiakasyritys oli modernisoimassa toimintaansa ja siirtymässä vanhasta 80-luvulta peräisin olevasta merkkipohjaisesta toiminnanohjausjärjestelmästä uuteen avoimen lähdekoodin web-käyttöliittymällä varusteltuun Odoo -järjestelmään. Tähän siirtymään vaadittiin integraatiojärjestelmä pitämään molemmat järjestelmät ajan tasalla.

1.1 Tutkimuksen taustoja

Integraatioprojektissa on paljon ulkoisia tekijöitä ja vaatimuksia, tehden siitä monimutkaisen projektin suunnitella ja hallinnoida. Lisäksi integraatio-ohjelmistoa ei ole käytetty aiemmin, eli projektiin liittyy myös tuotekehitystä. Kolmantena riskitekijänä on se, että kehittäjiä on käytännössä vain yksi ja hänen kokemuksensa käytetyistä teknologioista tai ohjelmointikielestä on lähes olematonta.

Projektille annettiin heti alussa reilusti aikaa, mutta projekti on silti aikataulustaan reilusti myöhässä ja sen kehitystyö oli kestänyt, sekä kalenterissa että tuntityömäärässä, lähes kaksinkertaisesti suunnitelmia pidempään. Projektin edetessä yllämainitut riskitekijät aiheuttivat suuren määrän ongelmia ja esteitä, jonka lisäksi integraation vaatimien ulkoisten tekijöiden työ oli myös täynnä esteitä ja hankaluuksia.

Tutkimuksen pääasiallinen tarve tulee yrityksen sisäisestä tarpeesta saada kattava analyysi projektista. Toimittajayritys on kiinnostunut projektitasolla tietämään, missä tilanteissa projektin etenemistä voi helpottaa ja millaisin menetelmin ongelmia ei pääse synty-

mään. Toimittajayritys on kiinnostunut myös yleisellä tasolla parantamaan toimintaansa projektinhallinnan ja -johdon kannalta. Heitä kiinnostavat myös mahdolliset toimiviksi löydetyt työ-, suunnittelu- ja dokumentointimenetelmät.

Odoo-toiminnanohjausjärjestelmän integraatiosta muihin järjestelmiin löytyy paljonkin kirjallisuutta ja muita tapaustutkimuksia, mutta jokaisella integraatioprojektilla on omat erityispiirteensä. Internetistä löytyy jopa muita Zato-integraatiojärjestelmää käyttäviä Odoo-integraatioita, mutta niissä integroitavat järjestelmät ovat kohtuullisen moderneja eivätkä lähes 30 vuotta vanhoja kuten Tonni -toiminnanohjausjärjestelmä.

1.2 Tutkimuksen rakenne

Tutkimus aloitetaan tiedon keruulla. Tietoa kerätään neljästä lähteestä: Työn kestot ja ajankohdat saadaan tuntikirjauksista; taustamateriaalit projektin aikana tehdyistä muistioista ja taulukoista; muiden työntekijöiden mielipiteet ja -kuvat projektista saadaan haastatteluista; sekä projektisuunnittelu ja -hallintateoria saadaan kirjallisuudesta. Kerätty tieto jäsennellään aikajanalle ja sieltä etsitään teemoja, jotka liittyvät toisiinsa samalla ajanjaksolla. Näitä ajanjaksoja kutsutaan vaiheiksi, joilla projektin eteneminen jaotetaan omiksi osikseen.

Tutkimuksessa läpikäydään integraatioprojekti vaihe vaiheelta, kertoen mitä projektissa tehtiin. Kertomisen apuna käytetään tuntikirjauksia ja projektin materiaaleja. Erityishuomiota annetaan onnistumisille ja epäonnistumisille. Kertomukseen yhdistetään teoriaa ja haastatteluja, jotta saadaan kokonaiskuva siitä, mitä kannattaa tehdä toisin ja mikä onnistui hyvin.

Lopuksi nämä onnistumisten ja epäonnistumisten huomiot kerätään yhteen ja jaotellaan ryhmiin sekä syy-seuraus-suhteisiin. Näin saadaan vastaukset tutkimusongelmiin ja -kysymyksiin.

1.3 Tutkimusongelma ja kysymykset

Tutkimusongelmana on, kuinka näitä riskitekijöitä voidaan nähdä etukäteen? Miten tunnistettuihin riskeihin pystytään varautumaan tai jopa estää niiden toteutuminen? Jälkiviisaana on helpompi tarkastella näitä kysymyksiä, joten projektista etsitään sen aikana nousseita ongelmia ja pyritään keksimään ratkaisuja niihin. Osaan ratkaisu saattoi löytyä jo projektin aikana, osaan täytyy perehtyä tarkemmin kirjallisuuden avulla.

Tämän työn lopputuloksia tullaan käyttämään kehittäjäyrityksen sisällä tarkastelemaan heidän työtapoja ja toivottavasti parantamaan niitä. Näiden vaatimusten pohjalta tutkimus-kysymyksiksi valikoituivat:

- Mitkä olivat kriittisimmät virheet projektin aikana?
- Miten virheisiin reagoitiin ja olisiko ne voitu estää

2 Asiakasprojekti

Tässä luvussa kuvaillaan asiakasprojektiin liittyvät sidosryhmät sekä projektissa käytettyjä termejä, teknologioita ja ohjelmistoja. Kuvailtaviksi on rajattu vain integraatioprojektiin liittyvät asiat. Laajemman asiakasprojektin termistöä ei ole kuvailtu tässä lainkaan.

2.1 Sidosryhmät

Toimittajayritys on 10–20 työntekijän IT-alan pk-yritys, jolla on kaksi päätoimialaa. Yrityksen alkuperäinen toimiala on internet-sivujen suunnittelu, toteuttaminen ja ylläpito. Uudempana toimintana edellisen viiden vuoden aikana on voimakkaasti kasvanut Odoo-toiminnanohjausjärjestelmien asennus-, räätälöinti- ja ylläpitopalvelut.

Asiakasyritys on 40–50 työntekijän rautakauppa-alan yritys. Yrityksen toimialana on teräs, metalli, ruuvi ja muiden rautakauppa-alan tuotteiden tukku- ja vähittäiskauppa, tuonti ja vienti sekä metallialan tuotteiden valmistus. Yrityksen tuotannonohjausjärjestelmät hallinnoivat ostoja, myyntiä, valmistusta ja varastonhallintaa.

Integraatiojärjestelmällä itsellään ei ole muita käyttäjiä kuin järjestelmänvalvojat, sillä se toimii vanhan ja uuden järjestelmän välisenä liimana. Integraatiossa liikkuvalla tiedolla on kuitenkin käyttäjänsä, jotka ovat myyjät ja varaston työntekijät. Integraatioprojektin sidosryhmiin kuuluvat asiakasyrityksen johto, keräilyjärjestelmän kehittäjät, Odoo -järjestelmän kehittäjät sekä Tonni -järjestelmän kehittäjät.

Johdon tehtävänä tässä projektissa on kuvailla asiakasyrityksen toimintatapoja ja heidän vaatimuksiaan integraatiolle. Keräily- ja Odoo -järjestelmien kehittäjillä on omat vaatimuksensa siitä, mitä tietoa integraation tulee siirtää järjestelmien välillä. Tonni -järjestelmän kehittäjän tehtävänä on luoda tietolähteet ja tapa jolla tietoihin pääsee käsiksi. Lisäksi hänen tehtävänään on luoda puitteet tiedon lukemiseksi takaisin Tonniin.

2.2 Systeemikuvaus

Integraatioprojektiin kuuluu kolme osaa: kaksi toisistaan erillään olevaa integroitavaa järjestelmää ja itse integraatiojärjestelmä, jonka avulla kaksi edellistä pystyvät vaihtamaan informaatiota. Tässä projektissa asiakkaan tarpeena oli pitää kahdessa eri toiminnanohjausjärjestelmässä olevat tiedot synkronoituna mahdollisimman reaaliaikaisesti. Asiakkaalle oltiin luomassa uuttaa järjestelmää, mutta siirtymävaiheen aikana sekä uuden että vanhan piti olla käytettävissä samanaikaisesti. (Liite 2.)

2.2.1 Toiminnanohjausjärjestelmä

Toiminnanohjausjärjestelmä on ohjelmisto, joka auttaa jäsentämään ja ylläpitämään tieto- ja liiketoiminnasta. Järjestelmistä käytetään yleisesti myös englanninkielistä lyhennettä ERP (Enterprise Resource Planning). Tunnettuja toiminnanohjausjärjestelmiä ovat esimerkiksi SAP ja Microsoft Dynamics. (Tech-FAQ 2016.)

Toiminnanohjausjärjestelmien tarkoituksena on säilyttää tieto yrityksen kaikesta toiminnasta yhdessä järjestelmässä, jolloin tiedon tallennus, haku ja analysointi ovat nopeita ja tehokkaita. Muutamia esimerkkejä tallennetusta tiedosta ovat myynnit, työtunnit, varastonhallinta, laskutus ja projektien aikataulutus. Järjestelmät myös linkittävät nämä tiedot toisiinsa siten, että käyttäjien on helppo sekä hahmottaa laajempia kokonaisuuksia, että porautua yksityiskohtiin. (Tech-FAQ 2016.)

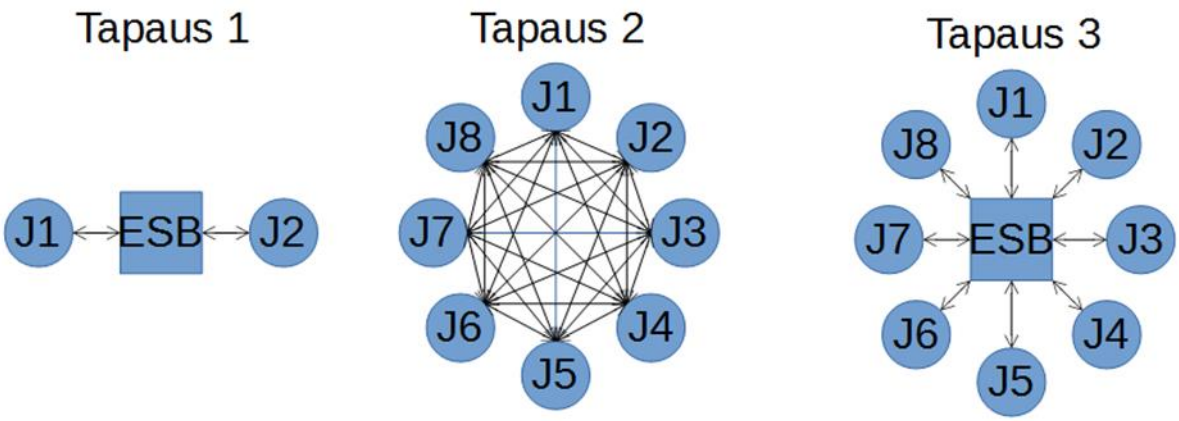
Toiminnanohjausjärjestelmät automatisoivat monia työtehtäviä. Esimerkiksi myynnin varmistuttua järjestelmä luo myyntitietojen perusteella varastolle keräilylistan. Jos joitain tuotteita puuttuu varastosta, järjestelmä luo tarjouspyynnön tavarantoimittajille. Tuotteiden keräilyn ja toimituksen jälkeen järjestelmä lähettää automaattisesti laskun, ja muuttaa varastosaldojen tiedot. Tällä tavoin säästetään valtavasti työtunteja ja myös estetään inhimillisten virheiden synty. (Tech-FAQ 2016.)

2.2.2 ESB-järjestelmä

ESB on lyhenne sanoista Enterprise Service Bus. ESB-järjestelmän tehtävänä on vastaanottaa ja siirtää tietoa usean eri järjestelmän välillä. ESB-järjestelmässä on kaksi eri tiedonsiirtoprosessin käynnistystapaa: Yleisempi on kanava, eli web-osoite, johon tieto lähetetään. Toinen tapa on ajastettu tiedonhakija, joka tietyin väliajoin käy hakemassa tietoa ulkoisesta lähteestä, esimerkiksi ftp-palvelimelta. Jos luettu tieto ei ole oikeassa muodossa, ESB-järjestelmän ensimmäinen tehtävä on muuntaa se järjestelmän sisällä käytettävään muotoon. Mikäli muunnosta ei pystytä tai haluta tehdä, järjestelmän voi määrittää vastaanottamaan vain oikeassa muodossa olevaa tietoa. (Ryan 2011.)

Tiedon luvun jälkeen se lähetetään eteenpäin määritysten mukaisesti yhteen tai useampaan ulospäin menevään kanavaan. Nämä kanavat voivat olla lähes mitä vain, kunhan ESB-järjestelmä tukee kanavaa. Esimerkkeinä toiminnanohjausjärjestelmä, ftp-palvelin, tai Twitter. Mikäli ulkoiset kanavat eivät ymmärrä ESB-järjestelmän sisäistä tietotyyppia, lähetettävä tieto täytyy muuntaa oikeantyyppiseksi ennen lähettämistä. (Ryan 2011.)

ESB-järjestelmän avulla tietoa voi helposti lähettää lukuisten eri järjestelmien välillä välittämättä siitä, missä muodossa muut järjestelmät ovat valmiita vastaanottamaan tiedon (Ryan 2011.) Kahden erillisen järjestelmän väliseen kommunikaatioon ESB-järjestelmän tarjoama hyöty on tietotyyppien muuntaminen (Kuva 1: Tapaus 1), mutta jos erillisiä järjestelmiä on useita, tarvittavien ja ylläpidettävien yhteyksien ja muunnoksien määrä kasvaa eksponentiaalisesti (Kuva 1, Tapaus 2). ESB-järjestelmän avulla jokainen erillinen järjestelmä vaatii vain yhden yhteyden (Kuva 1, Tapaus 2).



Kuva 1, ESB-järjestelmän hyödyt (Zato 2016.)

2.2.3 Tonni

Asiakkaan nykyinen toiminnanohjausjärjestelmä on 1980-luvun loppupuolelta oleva Unix-pohjainen Tonni. Tonnista ei ole saatavilla mitään ulkopuolista dokumentointia tai tietoa, joten sen toiminnasta ei ole saatavilla muita lähteitä kuin minun omakohtainen kokemus. Tonni on täysin tekstipohjainen ja itsenäinen järjestelmä, eli se ei ole mitenkään kytköksissä ulkomaailmaan. Integraatioprojektia varten Tonnin toimittaja lisäsi järjestelmään xml-tiedostojen kirjoitus- ja lukutoiminnallisuudet. Tonnin tietojen muuttuessa tietue tallennetaan levylle ftp-palvelimen ”out”-hakemistoon, jota Zato käy tarkistamassa minuutin välein. Tiedon kirjoitus toimii päinvastoin: Zato kirjoittaa ftp-palvelimen ”in”-kansioon tiedot, jotka Tonni käy hakemassa minuutin välein.

Tonnissa ei ole mitään tietoeheyden tarkistusta, ei edes taulujen välisten tunnisteiden olemassaolon vaatimusta. Tämä aiheutti integraatioprojektin suurimman haasteen, sillä minkään sisään tulevaan tiedon kanssa ei voinut luottaa sen olevan vaaditussa muodossa.

2.2.4 Zato

Zato on Python -kielellä ohjelmoitu avoimen lähdekoodin ESB-järjestelmä. Zaton asennus ja käyttö on ilmaista, mutta sille on tarjolla maksullisia tuki- ja koulutuspalveluita. Zato tukee lukuisia protokollia, järjestelmiä, alan standardeja ja tietotyyppejä. Zaton toiminnan voi jakaa karkeasti seuraaviin osa-alueisiin: käyttöliittymä; sisään tulevat kanavat sekä kuormituksen tasaaja ja palvelimet; sekä sisäiset palvelut ja ulkoiset yhteydet. (Zato 2016.)

Zatolla on sekä selainpohjainen graafinen- että komentorivikäyttöliittymä, joiden lisäksi osa konfiguraatiosta tehdään suoraan tiedostoihin. Kaikkea Zaton toimintaan liittyvää voi kuitenkin suorittaa ja konfiguroida molemmista käyttöliittymistä. Zato käyttää kahta eri tietokantaa, joihin tallennetaan asetukset ja historiatiedot: relaatiokantaa kuten MySQL tai Postgres ja Redis-kantaa, joka on avainparikanta. Graafisessa käyttöliittymässä on lisäksi monitorointitiedot. Yhtenä tärkeänä ominaisuutena valtaosaa Zaton asetuksista ja konfiguroinneista voi muuttaa lennosta, ilman että järjestelmää tarvitsee koskaan käynnistää uudelleen. (Zato 2016.)

Sisään tulevaa tietoa varten Zatossa on kanavat, jotka ovat järjestelmän kuuntelijoita. Kanavien asetuksissa määritetään nimi, osoite, autentikointitapa ja sisään tulevan tiedon tyyppi. Kanaville tulevat kutsut lähetetään eteenpäin kuormantasaajalle. Zato käyttää kuormantasaajana haproxy-nimistä ohjelmistoa, joka on avoimen lähdekoodin kevyt ja erittäin nopea välityspalvelin. Jokainen kanavan kutsu käy kuormantasaajan läpi. Se jakaa kutsun pienimmässä kuormituksessa olevalle palvelimelle. Zatossa voi olla yhdestä useampaan palvelinta päällä samanaikaisesti. Palvelimet ovat klooneja, joten uuden pystyttäminen ei vaadi muuta kuin yhden numeron vaihtamista asetuksista. (Zato 2016.)

Kutsun saavuttua palvelimelle palvelin käynnistää kanavan asetuksissa määritetyn palvelun, ja välittää sille mahdollisesti annetut tiedot. Palvelun voi käynnistää myös ajastetusti. Palvelut voivat kutsua toisiaan, ja niiden tarkoitus on käydä läpi kaikki integraation ohjelmallinen logiikka. Palvelut voivat käyttää asetuksissa määriteltyjä ulkoisia yhteyksiä lukeakseen tai lähettääkseen tietoa. Zatossa on suuri määrä ulkoisia yhteyksiä, joita voi käyttää tiedon lukemiseen, lähettämiseen tai kirjoittamiseen. (Zato 2016.)

2.2.5 Odoo

Odoo on Python- ja JavaScript -kielillä ohjelmoitu avoimen lähdekoodin toiminnanohjausjärjestelmä. Järjestelmän konfigurointi tapahtuu XML-tiedostojen avulla. Odoon käyttöliittymä on selainpohjainen, mutta toiminnoiltaan ja ulkonäöltään se vaikuttaa työpöytäohjelmalta. (Odoo 2016.)

Odoo ei ole asennettaessa välittömästi toimintavalmis paketti, vaan kaikki järjestelmän toiminnot on pilkottu osiin, joita kutsutaan moduuleiksi. Näitä moduuleja ovat esimerkiksi Kirjanpito, Myynti, Tuotannonohjaus ja Varastonhallinta. (Odoo 2016.)

Odoossa on ulkoisia komentoja varten käytössä XML-RPC protokolla. RPC, eli Remote Procedure Call, on tapa käyttää palvelimen komentoja toiselta laitteelta. XML-RPC:n avulla Odoosta voi lukea, luoda, kirjoittaa ja poistaa tietoja. Lisäksi Odoon sisäisille toiminnoille voi lähettää komentoja, kuten esimerkiksi varmista myynti tai tulosta lasku. (Odoo 2016.)

2.3 Asiakasprojekti

Tämän tutkimuksen kohteena oleva integraatioprojekti oli osa asiakkaalle tehtyä laajempaa projektia, jossa uudistettiin täysin heidän tuotannonohjausjärjestelmänsä ja työtapoja muutenkin. Asiakkaan vanha tuotannonohjausjärjestelmä on ollut käytössä 80-luvulta alkaen, joten heillä oli tarve vaihtaa järjestelmä täysin uuteen. (Liite 2.)

Asiakas ei kuitenkaan halunnut aloittaa puhtaalta pöydältä ja siirtyä välittömästi uuteen järjestelmään, vaan liiketoiminnan jatkumisen kannalta oli tarpeen ajaa molempia järjestelmiä rinnakkain siirtymäajan yli. Tähän tarpeeseen tarvittiin integraatiojärjestelmä, jotta molempien järjestelmien tiedot täsmäävät toisiinsa. (Liite 2.)

Asiakkaalle oltiin uuden toiminnanohjausjärjestelmän lisäksi samanaikaisesti uusimassa heidän keräilytoimintojaan. Keräilyä hoidettiin vanhaan tapaan tulostamalla lähete paperille, viemällä se varastolle ja lopuksi merkitsemällä samalle paperille tehdyt toimenpiteet. Tämä korvattiin sähköisellä lähetteellä ja varastossa käytetyillä tablettitietokoneilla. Tämä tablettikeräilyjärjestelmä vaati integraatiota siirtämään myyntitiedot ja lähetteet uuteen järjestelmään, ja keräilyn valmistuttua lähettämään tiedot takaisin vanhaan järjestelmään. (Liite 2.)

2.3.1 Tarpeet ja tavoitteet

Laajemman asiakasprojektin tavoitteena oli täysin uudistaa ja modernisoida yrityksen toiminnanohjaus- ja hallinnointijärjestelmät. Asiakas tarvitsi integraatioprojektia kahdesta syystä: Nykyisten tietojen siirtämiseen vanhasta järjestelmästä uuteen ja kahden järjestelmän tietojen synkronoinnin siirtymävaiheen aikana. Integraation tehtävänä oli siirtää asiakasyrityksen asiakas-, toimittaja-, tuote- ja myynti- ja varastonhallintatiedot. (Liite 2.)

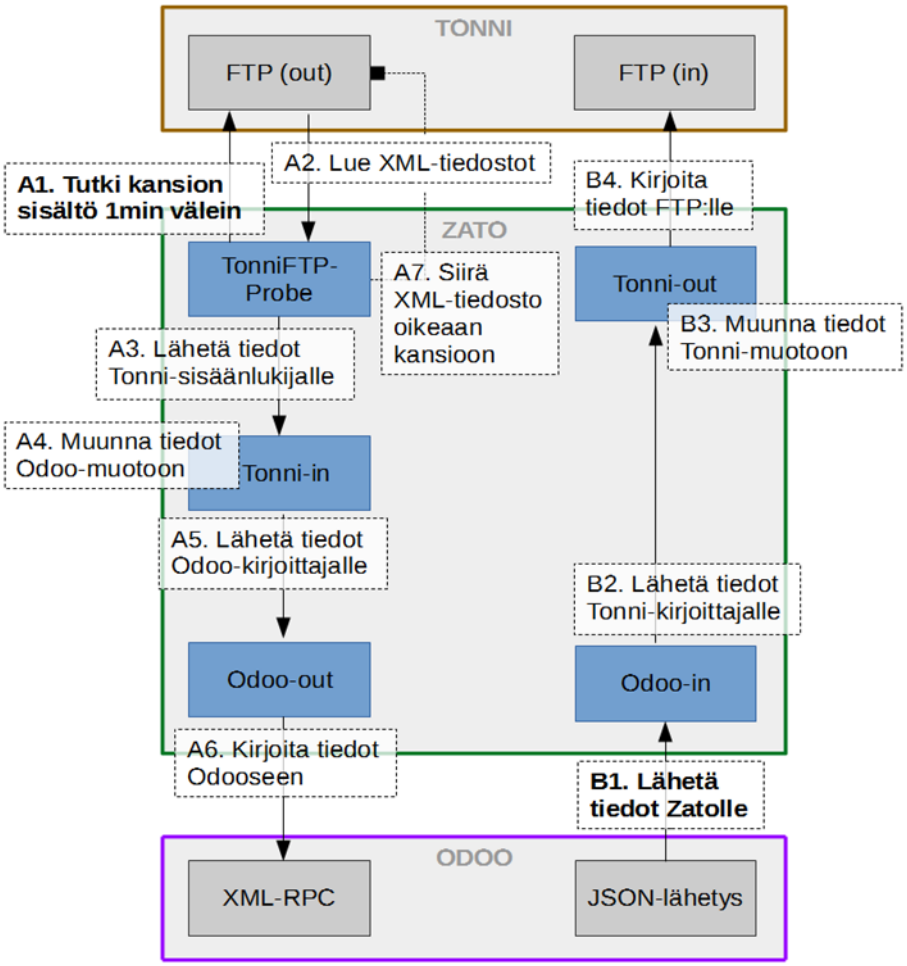
Integraatioprojektin tehtävänä oli luoda järjestelmä, joka reaaliaikaisesti synkronoi asiakkaan vanhan ja uuden toiminnanhallintajärjestelmän tiedot keskenään. Lisäksi integraatio kopioi kaikki vanhasta järjestelmästä tarvitut tiedot uuteen. Asiakkaan vanha järjestelmä on unix-pohjainen 80-luvun lopulla käyttöön otettu Tonni. Asiakkaan uusi järjestelmä on vapaan lähdekoodin web-käyttöliittymällä käytettävä Odoo. (Liite 2.)

Integraatioprojektin ehdoton ykkösprioriteetti oli tiedonsiirron 100-prosenttinen toimivuus. Toisena tavoitteena oli, että tiedonsiirto tapahtuisi reaaliaikaisesti. Tiedonsiirron varmistamiseksi oli tärkeää, ettei prosessi saanut kaatua tai keskeytyä ilman ilmoituksia. (Liite 2.)

2.3.2 Integraation toiminnot

Integraatioprosessi käynnistyy, kun Tonnissa tallennetaan uusi tai muokataan vanhaa tietuetta. Tällöin Tonni kirjoittaa tiedot XLM-tiedostoon FTP-palvelimelle. Integraatiojärjestelmä käy kerran minuutissa lukemassa FTP-palvelimelta XML-tiedostot (Kuva 2: A1, A2, A3), jotka se muuntaa Odoon käyttämään muotoon (Kuva 2: A4, A5.) Muunnoksen jälkeen integraatiojärjestelmä muokkaa tai luo tiedot Odooseen (Kuva 2: A6.) Lopuksi integraatio siirtää XML-tiedoston FTP-palvelimelle valmiiden, tai virhetapauksessa virheellisten, kansioon (Kuva 2: A7.)

Integraatio toimii myös toiseen suuntaan jolloin tietoa viedään Odoosta Tonniin. Järjestelmäkokonaisuuden Master Datana toimii Tonni, joten Odoosta siirretään vain keräilyn tietoja: kerätyt tai luodut tilausrivit sekä tilauksen käsittelytilan muutokset (Liite 2.) Integraatioprosessi käynnistyy, kun keräily merkitään valmiiksi. Tällöin tiedot lähetetään integraatiojärjestelmälle (Kuva 2: B1), joka muuntaa ne XML-pohjan mukaiseen muotoon (Kuva 2: B2, B3) ja lopulta kirjoittaa XML-tiedoston Tonnin käyttämälle FTP-palvelimelle (Kuva 2: B4.) Uuden tiedoston huomatessaan Tonni kirjoittaa tiedot omaan tietokantaansa.



Kuva 2, Integraatiojärjestelmän toiminta.

2.4 Projektinhallinta

Projektinhallinta on tiedon, taitojen, työkalujen ja tekniikoiden soveltamista projektin toimintoihin, jotta tavoitetaan projektin vaatimukset. Projektin hallinnointiin kuuluu vaatimusten selvittäminen; sidosryhmien tarpeiden, huolien ja odotusten käsittely projektia suunnitellessa ja sen edetessä; sekä projektin kilpailevien rajoitusten tasapainottelu. Näitä rajoituksia ovat esimerkiksi projektin laajuus, aikataulu, budjetti, resurssit, riskit ja lopputuloksen laatu. (PMBOK 2008, 6.)

Rajoitusten määrä ja niiden tärkeysasteet vaihtelevat projekteittain, jonka lisäksi eri sidosryhmillä saattaa olla eriäviä mielipiteitä siitä, mitkä rajoitteista ovat tärkeimpiä. Rajoitukset ovat toisiinsa kytköksissä siten, että jos yksi niistä muuttuu, se vaikuttaa myös muihin rajoitteisiin. Esimerkiksi, jos projektin aikataulua kiristetään, projektin budjettia täytyy kasvattaa, jotta ylimääräisiä resursseja saadaan käyttöön. Jos nämä muutokset eivät ole mahdollisia, projektin laajuutta tai laatua pitää laskea. (PMBOK 2008, 6.)

Kaikkien näiden muutosten vaikutuksena myös projektin riskin saattavat kasvaa tai uusia riskejä ilmestyä. Muutosmahdollisuuksien vuoksi projektinhallintasuunnitelma elää projektin mukana. Suunnitelmaa on tarkennettava ja paranneltava projektin etenemisen tuoman tiedon ja kokemusten mukaan. (PMBOK 2008, 6.)

2.4.1 Prosessiryhmät

Projektinhallinta on valtava ja monimutkainen ala, PMI:n ohjeiden mukaan se on jaoteltu 42:een eri projektinhallintaprosessiin (PMBOK 2008, 43.) Prosessit soveltuvat jokaiseen projektiin toimialasta riippumatta ja projektin laajuus määrittelee sen, kuinka tarkasti eri prosesseja pitää suunnitella ja toteuttaa (PMBOK 2008, 38.) Prosessit on ryhmitelty viiteen eri prosessiryhmään: aloitus, suunnittelu, toteuttaminen, seuranta ja valvonta sekä sulkeminen (PMBOK 2008, 39.)

Aloitusryhmän prosessit määrittävät uuden projektin tai projektivaiheen saamalla valtuutukset projektin tai vaiheen aloittamiseen. Suunnitteluryhmän prosessit määrittelevät projektin laajuuden ja tavoitteet sekä vaadittavat toimenpiteet niiden aikaansaamiseksi. Toteuttamisryhmän prosesseissa edellä mainitut vaatimukset toteutetaan, jotta tavoitteet saadaan täytetyksi. Seuranta- ja valvontaryhmän prosesseja tarvitaan projektin etenemisen ja tehokkuuden seurantaan, mittaukseen ja säätämiseen. Sulkemisprosessit viimeistelevät ja sulkevat kaikki muut aiemmat tehtävät kaikista muista prosessiryhmistä. (PMBOK 2008, 39.)

2.4.2 Integraatioprojektin näkökulma

Projektinhallinnassa on prosessiryhmien lisäksi toinen jaottelu, osaamisalueet. Osaamisalueet jakautuvat projektin hallinnoinnin eri osiin: integraation-, laajuuden-, ajan-, kustannusten-, laadun-, henkilöstön-, kommunikaation-, riskien- ja hankintojen hallintaan (PMBOK 2008, 67.) Tämän työn tarkoituksena ei ole kuvailla tai opastaa projektinhallintaa kokonaisuutena, joten keskityn vain ensimmäiseen osaamisalueeseen, eli projekti-integraation hallintaan. Projekti-integraatio ja integraatioprojekti eivät samantapaisista nimistään huolimatta ole toisiinsa liittyviä termejä, vaan projekti-integraatio tarkoittaa projektin eri osa-alueiden yhteisten liitosten ylläpitoa (PMBOK 2008, 71.)

Projekti-integraatioon kuuluu viisi eri hallintaprosessia, yksi jokaista prosessiryhmää kohden (Taulukko 1.)

Taulukko 1 (PMBOK 2008, 71.)

Aloitus	Suunnittelu	Toteuttaminen	Seuranta ja valvonta	Sulkeminen
Kehitä hanke-suunnitelma	Kehitä hallin-tasuunnitelma	Johda ja hallinnoi toteuttamista	Seuraa ja ohjaa työtä, hallinnoi muutoksia	Sulje projekti tai vaihe

Aloitusvaiheessa kehitetään hankesuunnitelma. Tehtävänä on luoda työkirja, joka sekä valtuuttaa projektin tai vaiheen käynnistämisen että dokumentoi projektin sidosryhmien tarpeet ja odotukset täyttävät alustavat vaatimukset. Suunnitteluvaiheessa kehitetään hallintasuunnitelma. Tehtävänä on dokumentoida vaadittavat toiminnot, joiden avulla projektin suunnitelmat määritetään, valmistellaan, integroidaan ja koordinoidaan. Toteutusvaiheessa toteutetaan hallintasuunnitelman mukaiset tehtävät. Seuranta ja valvontavaiheessa seurataan, tarkistetaan ja säädetään projektin etenemistä, jotta se tapahtuu hallintasuunnitelmassa määritettyjen tavoitteiden mukaan. Lisäksi tässä vaiheessa tarkistetaan kaikki muutospyynnöt, hyväksytään tarpeelliset pyynnöt ja hallinnoidaan muutokset. Sulkemisvaiheessa viimeistellään kaikkien muiden vaiheiden toiminnot ja saadaan projekti tai vaihe valmiiksi ja suljettua. (PMBOK 2008, 71.)

Integraatioprojekti on vain yksi osaa laajempaa asiakasprojektia, ja sen tehtävänä on tuottaa järjestelmä, joka toimii liimana tiedonsiirrossa kahden eri ESB-järjestelmän välillä. Toisin sanoen, asiakasprojekti ei voi onnistua tavoitteiden mukaisesti ilman toimivaa integraatiojärjestelmää. Täten integraatioprojekti vaatii onnistuakseen tehokasta projekti-integraationhallintaa, jotta asiakasprojektin eri palasten kehitystyö etenee suunnitellusti ja aikataulussa.

2.4.3 SCRUM

SCRUM on yksinkertainen ja iteratiivinen ketterän ohjelmistokehityksen malli, jossa monimutkaisen ja suuren tuotteen kehitys pilkotaan pienempiin ja yhä pienempiin osiin, joita kutsutaan täskeiksi. Näille täskeille asetetaan tarkat vaatimukset, tavoitteet, aikataulut ja työarviot. Tämä auttaa kehittäjiä, joiden täytyy hallita vain yhden täskin sisältämä kokonaisuus kerrallaan. Toisaalta projektipäällikkö näkee selkeästi projektin etenemisen ja aikataulussa pysymisen tilan. (Weckström 2014, 10.)

Tässä projektissa emme käyttäneet täyttä SCRUM-mallia, vaan otimme siitä käyttöön lähinnä täskityksen ja Sprintit. Sprint on SCRUM-mallissa kehityksen ajanjakso, yleensä 30 päivän eli noin kuukauden mittainen. Sprinttiä ennen suunnitellaan ja lyödään lukkoon kaikki Sprintin aikana tehtävät täskit, eikä niitä saa lisätä Sprintin aikana. Tämä parantaa

projektin etenemisen ennustettavuutta. Sprintin lopuksi pidetään Retrospektiivi, jossa käydään läpi Sprintin täskit ja kerätään hyvää ja huonoa-lista Sprintin ajan tapahtumista. (Weckström 2014, 12.)

3 Metodologia

Tutkimusstrategiana on tapaustutkimus, jolla pyritään selvittämään projektin tapahtumien syy-seuraus-suhteita ja löytämään havaintoja käymällä läpi projektin etenemistä. Etenemisen tutkimisen apuna käytetään projektin aikaisia tuntikirjauksia ja projektia varten luotuja dokumentteja. Lisäksi haastatellaan asiakasprojektin muita kehittäjiä, joilta pyritään selvittämään uusia näkökulmia, tiedonpalasia ja kuinka heidän mielipiteensä vertautuvat omiin mielipiteisiini projektin tapahtumista.

3.1 Tapaustutkimus

Tutkimuksen tyyppinä on tapaustutkimus, englanniksi Case Study. Tapaustutkimus on empiirinen selvitys joka tutkii perusteellisesti ajankohtaista tosielämän ilmiötä. Tutkimustapa on erityisen hyödyllinen kun ilmiön ja kontekstin rajat eivät ole selkeästi havaittavissa. Tapaustutkimuksen selvitys on hyödyllistä, kun tutkittavana on teknillisesti omalaatuinen tapaus jossa on paljon enemmän muuttujia kuin datapisteitä. Tästä johtuen tapaustutkimus on riippuvainen useista todistuslähteistä, joiden tiedot yhdistyvät eri suunnista tai näkökulmista. Tapaustutkimus hyödyntää aiempia teoreettisia tutkimuksia tai ehdotuksia jotka ohjaavat tiedon keruuta ja analysointia. (Yin 2013, 18.)

Tapaustutkimuksen tarkoitus on kuvailla kausaalinen linkki tapahtuman ennen- ja jälkeentilojen välillä. Tähän päästään kuvailemalla tiettyjä aiheita tapahtuman aikana ja selvittämällä niitä kerättyjen faktatietojen avulla. (Yin 2013, 19.)

Tapaustutkimus on hyvä valinta tutkimustyypiksi silloin, kun pyritään vastaamaan "kuinka" tai "miksi" -kysymyksiin, tai kun tutkijalla on vain vähän kontrollia tapahtumien kulusta. Erityisesti silloin, kun tarkastellaan tämänhetkistä tapahtumaa tosielämässä, tapaustutkimus on tehokas tapa saada vastauksia tutkimuskysymyksiin. (Yin 2013, 8.)

3.1.1 Tapaustutkimussuunnitelma

Tapaustutkimuksen tutkimussuunnitelma on looginen suunnitelma siitä, kuinka päästään alkuperäisistä kysymyksistä lopullisiin päätelmiin. Tapaustutkimuksen suunnittelu aloitetaan valitsemalla tutkimuskysymykset. Tutkimuskysymysten tehtävänä on kysyä miten tai miksi tutkimuksen kohteena oleva tapaus tai aihe eteni tai päättyi siten kuin lopulta tapahtui. Kysymysten tarkoituksena on selvittää mitä päätöksiä tehtiin tai miten päätökset toimeenpantiin ja mistä syystä näin päädyttiin tekemään. (Yin 2013, 27–28.)

Ennen tutkimuksen aloittamista täytyy varmistaa, onko tutkimukselle tarvetta. Antavatko tutkimuskysymysten vastaukset mahdollisesti hyödyllistä, tarpeellista tai uutta tietoa? Jos samasta aiheesta on jo tehty aiempia tutkimuksia, tarkentaako tämä tutkimus aiempia tai antaako se uusia näkökulmia? (Yin 2013, 28.)

3.1.2 Tiedon mittaaminen ja analysointi

Kun tutkimuskysymykset on valittu ja tutkimuksen tarpeellisuus varmistettu, valitaan mitä tietoa on tarkoitus analysoida ja miten sitä mitataan. Tämän tiedon mittaamiseen täytyy myös valita analyysin mittayksikkö, jotta analyysia tehdään. Mittayksikkö voi olla suora arvo, kuten raha tai tunti, mutta myös abstraktimpi mittari tai indeksi. (Yin 2013, 29.)

Tutkimustyön perustana on löytää looginen linkki mitatun tiedon ja tutkimustulosten välillä. Tämän avulla todistetaan lopullinen tutkimustulos ja analyysi. Tutkimuksessa käytetään valittua mittayksikköä, jotta voidaan löytää ja todistaa korrelaatio ja linkit siihen, mitä tutkinnan alla olevan tapauksen aikana tapahtui. (Yin 2013, 29–33.)

Lopullisen todistetun tuloksen saanti vaatii vielä löydösten tulkkauskriteerit. Tulkkauskriteerien pohjalta tarkastellaan tutkittavan tapauksen aikaisia tapahtumia, päätöksiä ja muuta toimintaa. (Yin 2013, 33–35.)

3.2 Haastattelu

Yksi tutkimuksessa käytetty tietolähde on asiakasprojektissa työskentelevien henkilöiden haastattelut. Tutkimuksessa käytetään haastatteluja kyselyiden sijaan sen vuoksi, että haastattelut antavat paljon syvemmän ja monipuolisemman tietopohjan kuin kyselylomake. Haastattelukysymyksiä voidaan myös hieman muokata haastattelun aikana aiempien vastausten perusteella. (Turner 2010, 754.)

Haastatteluja on kahdenlaisia, rakenteellisia ja rakenteettomia. Rakenteellisessa haastattelussa kysymykset on päätetty etukäteen, ja vaikka vastauksen aikainen keskustelu saattaa polveilla, kaikilta haastateltavilta kysytään samat kysymykset. Rakenteellisissa haastatteluissa vastaukset ovat verrannollisia, mutta vähemmän monipuolisia kuin rakenteettomissa haastatteluissa. (Turner 2010, 755–756.)

Rakenteettomissa haastatteluissa kysymykset muuttuvat vastausten mukaan, ja ne muistuttavat enemmän keskustelua. Rakenteettomista haastatteluista voi saada uusia näkökulmia, mutta vastaukset eivät yleensä ole verrannollisia haastattelujen välillä. (Turner 2010, 756.)

3.2.1 Kysymysten suunnittelu ja eteneminen

Riippumatta haastattelutyypistä, haastattelua suunniteltaessa on tärkeää suunnitella sen kysymykset ja rakenne siten, että ne ovat mahdollisimman hyödyllisiä tutkimukselle. Kysymysten ei pidä yrittää suoraan vastata tutkimusongelmiin, vaan niiden pitää avata haastateltavan kokemuksia ja ymmärrystä tutkimuksen aiheesta. On erittäin tärkeää, että kysymyksiin ei voi vastata "kyllä" tai "ei", esimerkiksi "Menetkö töihin autolla?" sijaan kysytään "Millä kulkuvälineellä menet töihin?". Kuitenkin, jos haastattelulla halutaan saada laajempaa tietoa kuin kyselylomakkeella, kysymys tulee muotoilla avoimeksi ja keskustelua herättäväksi. Aiempaa esimerkkiä voi siis muokata muotoon "Kuvaile miten menet töihin." (Turner 2010, 757–758.)

Kolmas tärkeä asia on muistaa pitäytyä mahdollisimman neutraalina eikä yrittää muokata vastaajan mielipidettä kysymysten asettelulla. Tällaisia asetteluja ovat negatiivisten tai positiivisten sanojen käyttäminen, tai luomalla kuvitelma väärästä vastauksesta. Esimerkkinä kysymys "Mitä ongelmia koiran ulkoiluttajat aiheuttavat?", jonka neutraalimpi muoto on esimerkiksi "Mitä mieltä olet koirien ulkoiluttajista?". Poikkeuksena tähän sääntöön on, jos kysymyksen tarkoituksena on selvittää tuntemuksia tai kokemuksia jostain negatiivisesta tai positiivisesta tapahtumasta. Haastattelijan oma käytös on myös tärkeää, sillä ilmeillä tai kehon liikkeillä voi helposti viestiä omia mielipiteitä haastateltavalle. (Turner 2010, 758.)

Haastattelujen suurin etu on vastausten monipuolisuus. Monipuolisuus vaatii kuitenkin sitä, että haastateltavan annetaan kertoa vapaasti ja että haastattelija tarttuu mielenkiintoisiin aiheisiin jatkokysymyksillä. Jatkokysymyksiä käytetään tarkentamaan teemoja tai selventämään konsepteja, esimerkiksi "Mitä tarkoitat tällä?" tai "Voitko kertoa lisää tästä?" Jatkokysymykset voivat olla myös tiedustelevia, jolloin saadaan lisää syvyyttä ja yksityiskohtia. Tiedustelevien kysymysten tulee olla lyhyitä ja yksinkertaisia, jotta haastateltavan keskittymiskyky ei herpaannu. (Turner 2010, 758.)

Haastattelijan on myös oltava valmis muotoilemaan kysymys uudelleen, jos haastateltava ei ymmärrä mitä haastattelija ajaa takaa. Kysymyksiä voi muotoilla joko tarkemmiksi tai laajemmiksi tilanteesta riippuen. Kysymyksen voi myös yrittää liittää aiempiin annettuihin vastauksiin. Joissain tilanteissa kysymyksen voi ohittaa kokonaan, tai siihen voi palata myöhemmin, jos keskustelun rytmi katoaa. (Turner 2010, 759.)

Keskustelun rytmittämiseksi kysymysten järjestys on tärkeää ja siirtymisien aiheiden välillä pitää olla loogisia. Haastattelu on hyvä aloittaa helpoilla tai yleisillä kysymyksillä, esimerkiksi pyytämällä haastateltavaa kertomaan työstään ja tehtävistään. Alkuun voi olla hyvä kertoa yksinkertaisesti tutkimusaiheesta, erityisesti miten se liittyy haastateltavan tehtäviin. Arkaluonteiset ja kiistanalaiset kysymykset kannattaa kysyä haastattelun keskivaiheilla, sen jälkeen kun keskustelusuhde on syntynyt. Viimeinen kysymys tulee asetella siten, että haastateltavalla on mahdollisuus kommentoida läpikäytyjä teemoja, tai haastattelua itsessään. (Turner 2010, 759.)

3.3 Tietolähteet

Tutkimusta varten kerätään projektiin liittyvää tietoa kolmesta lähteestä. Kerätty tieto on numeerista, kirjallista ja haastateltavien keskusteluista litteroitua tekstiä. Numeerinen tieto tulee projektin aikana tehdyistä tuntikirjauksista, jotka tuodaan toiminnanohjausjärjestelmästä. Kirjallinen tieto haetaan projektin suunnitelmista, taulukoista ja muistiinpanoista. Projektin muuta henkilöstöä haastatellaan, ja haastattelujen nauhoitteet litteroidaan.

Pääasiallisena tutkimusmateriaalina käytetään tuntikirjauksia projektin ajalta. Tuntikirjaukset luokitellaan työtehtävittäin ja projektivaiheittain. Materiaalista etsitään erityisesti työtunteja, joita on tehty samalle työtehtävälle eri aikoina. Tällä tavoin yritetään hahmottaa milloin tehtäviä on jouduttu korjaamaan tai tekemään uudelleen.

Projektin aikana on luotu valtavasti dokumentteja, jotka ovat kaikki tallennettu projektia varten luotuun verkkokansioon. Dokumenteista etsitään projektia edeltäneitä tai projektin aikana lisättyjä suunnitelmia ja muistiinpanoja. Nämä ryhmitellään työtehtävittäin ja lisätään aikajanalle. Tarkoituksena on selvittää, miten suunnitelmat ja erityisesti määritykset ovat kehittyneet tai muuttuneet projektin edetessä.

3.3.1 Haastattelut

Haastattelujen tarkoituksena on selvittää muiden työntekijöiden, joiden työtehtävät ovat liittyneet tai vaatineet integraatiota, tietämystä projektista ja kuinka projekti on vaikuttanut heidän omaan työhönsä. Integraatioprojekti oli vain yksi osa laajempaa asiakasprojektia, joten haluan erityisesti selvittää kuinka hyvin eri projektien yhteistyö on sujunut ja kuinka hyvin projektien välinen kommunikaatio on toiminut.

Haastateltavina oli projektipäällikkö ja asiakasprojektin toisen osan, keräilyjärjestelmän, kehittäjä. Molemmat haastateltavat työskentelevät kanssani samassa yrityksessä. Projektipäällikkö on ollut IT-alalla 15 vuotta ja valmistunut diplomi-insinööriksi projektin aikana.

Hänen lopputyönsä aiheena oli Pk-yrityksen vaatimukset tuotannonohjaukselle avoimen lähdekoodin toiminnanohjausjärjestelmässä. Haastateltava kehittäjä on tutkintotodistushakemuksen lähettämistä vaille valmis tietotekniikan kandidaatti. Hän on ollut IT-alalla n. 2,5 vuotta.

Haastattelut pidettiin työpäivän aikana toimistolla. Haastatteluympäristö oli tuttu ja tilanne rento. Olin miettinyt ja kirjannut haastattelun kysymykset etukäteen, mutta en ollut kertonut niitä haastateltaville. Olin tarkoituksella jättänyt haastattelukysymykset avoimiksi ja keskustelua herättäväksi, jotta saan mahdollisimman laajan mielipiteen jokaisesta aiheesta.

4 Tutkimus ja tutkimustulokset

Projektin tutkimus on jaoteltu projektin etenemisvaiheittain. Nämä vaiheet eivät ole olleet osana projektisuunnitelmia, vaan muodostin ne tuntikirjausten ja projektidokumenttien analysoinnin pohjalta. Jokaisessa vaiheessa esitellään sen erityispiirteet, kehitystyön edistyminen ja lopulta analyysit.

Löysin projektista neljä eri vaihetta, joissa kaikissa oli omat haasteensa, tehdyt virheet ja kehitystyön etenemisen tahti. Viimeinen vaihe, kuukausittainen kehitys, on niin laaja, että se on jaettu Sprinteittäin omiin osiinsa.

4.1 Vaihe 1: Projektin aloitus ja suunnittelu

Integraatioprojekti syntyi tarpeesta siirtää tietoa Tonni- ja Odoo -järjestelmien välillä mahdollisimman reaaliaikaisesti. Tonni -järjestelmästä oli jo aiemmin siirretty tuotetietoja XML-muodossa FTP-palvelimelta, mutta se oli manuaalinen operaatio. Tarvittiin siis ohjelmisto, joka pystyi lukemaan tietoa FTP:ltä, muuntamaan tiedon, ja tallentamaan sen Odoon XML-RPC-rajapintaa käyttäen. (Liite 2.)

4.1.1 Ohjelmiston valinta ja opettelu

Integraatioprojektin ensimmäiset tuntikirjaukset ovat helmikuulta 2015 (Liite 7.) Projekti käynnistyi vertailemalla markkinoilla olevia avoimen lähdekoodin integraatio-ohjelmistoja. Sain listan vertailtavista ohjelmistoista ja lähdin etsimään niistä tietoa kuten ohjelmointikieli tai -kielet, valmiit liitännät muihin järjestelmiin sekä ohjeistuksen määrä ja taso. Välittömästi listan kärkeen nousi Zato kahden päätekijän vuoksi. Siinä oli valmiiksi Odoo -liitäntä ja se on ohjelmoitu Pythonilla. Odoo-järjestelmä ja asiakasprojektin muut osat ovat Pythonilla ohjelmoituja, joten myös integraation pitäminen Python -pohjaisena oli tärkeää. Zaton ainakin näennäisen ylivoimaisuuden takia muiden järjestelmien tarkastelu jäi vähemmälle huomiolle ja päädyimme käyttämään Zatoa.

Toinen syy Zaton valintaan oli sen nähtävästi hyvä dokumentaatio ja ohjeistus. Zaton asennukseen ja käyttöönottoon on olemassa hyvät ohjeet ja hitaasti etenevä opastus (Zato 2016.) Vaikka ohjeet olivat hyvät ja niitä oli melko paljon, eteneminen oli silti melko hidasta ja virheestä toiseen astelua. Tuntikirjausten mukaan käytin Zaton opetteluun ja vaatimusten täyttämiseen lähes 20 tuntia (Liite 7.) Olin aiemmin käyttänyt Pythonia, mutta siitä oli kulunut paljon aikaa. Osasyynä etenemistahtiin oli siis se, että opettelin samalla uutta ohjelmointikieltä. Lisäksi viralliset ohjeistukset kertoivat ainoastaan kuinka päästä

alkuun, mutta monipuolisemmat ominaisuudet olivat oman osaamisen tai tiedonhaun varassa. Ohjeissa ei myöskään kerrottu mitään siitä, kuinka korjata erilaisia virhetiloja.

Projektin aikataulutuksessa oli kuitenkin otettu tämä kaikki huomioon. Alkuvaiheessa aikataulu oli hyvin väljä. Projektipäällikkö kyseli tasaisin väliajoin etenemistäni ja olinko ajautunut joihinkin ongelmiin (Liite 2.)

4.1.2 Kenttien yhdistäminen

Integraatio-ohjelmiston valinnan jälkeen alkoi projektin tulevaisuuden kannalta ehkä kriittisin vaihe: Tonnin ja Odoon tietomallien kenttien yhdistäminen. Tonnin ja Odoon välillä on lähes 30 vuotta, joten pelkästään tietomallit ja tietojen tallennus- ja muokkaussäännöt olivat hyvin erilaiset (Kuva 4.)

Tonnista meille annettiin jokaisen tietomallin kentät ja niiden tietotyypit: oliko kentässä tekstiä, numeraaleja tai desimaaleja sekä kenttien maksimipituudet. Kuvittelimme, että voimme linkittää kenttiä toisiinsa nimien perusteella, mutta hyvin nopeasti järjestelmien erilaisuus alkoi näkyä linkitystyön hankaluuksina. (Kuva 4.)

Tonni-järjestelmässä valtaosa kentistä oli käyttäjien muokattavissa, joten vaikka esimerkiksi myyntitilauksen osoitetiedot haetaan asiakkaan tiedoista, käyttäjä voi muokata näitä jälkeenpäin. Tämä taasen ei ole lainkaan mahdollista Odoossa, vaan asiakas linkitetään myyntiin kannan id-numeron avulla. Integraation pitää siis käydä tarkistamassa, onko myynnissä annetut asiakkaan tiedot samat kuin Odoossa, ja jos eivät, luoda asiakkaalle uusi osoitetieto, johon myynti voidaan myöhemmin linkittää. (Kuva 4.)

Näitä järjestelmien yhteensopimattomuuden aiheuttamia poikkeuksia ja ongelmia ei ymmärretty projektin alkuvaiheessa riittävästi, ja ne olivatkin yksi suurimpia hidasteita projektin etenemisessä.

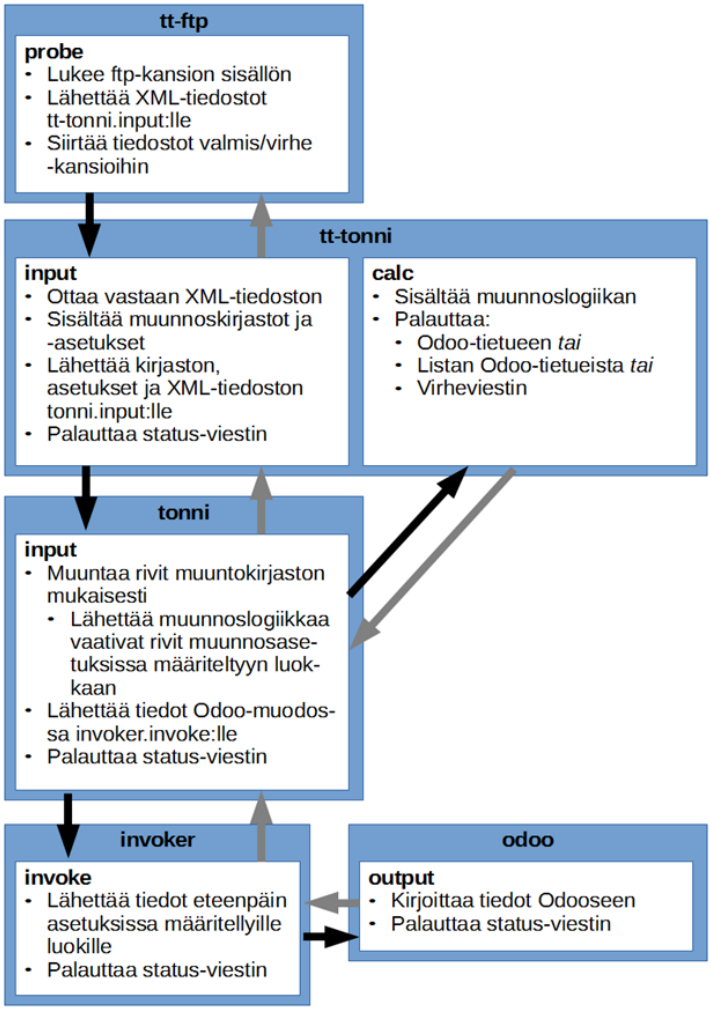
Tonni					Odoo	
Kenttä UI:ssa	Tekninen nimi	Tietotyyppi	Rajoitteet	Kuvaus	Lisätiedot	Kenttä UI:ssa	Tekninen nimi (res.partner)
Tunnus	yla_koodi	Teksti	x(10)	Asiakkaan uniikki ID		Viite	ref
Nimi-1	yla_nimi[1]	Teksti	x(30)			Nimi	name
Nimi-2	yla_nimi[2]	Teksti	x(30)	Asiakasyrityksen nimi, rivi 2		Osoite	street, jos määritetty
Osoite-1	yla_osoite[1]	Teksti	x(30)	Osoite, rivi 1		Osoite	street, jos Nimi-2 puuttuu, muuten street2
Osoite-2	yla_osoite[2]	Teksti	x(30)	Osoite, rivi 2	5 ekaa postinro, väli, p-toimipaikka	Zip ja Kaupunki	zip ja city
Puhnro	yla_puhno	Teksti	x(30)	Puhelinnumero(t)	Samassa kentässä voi olla useampi	Puhelin ja Matkapuhelin	phone ja mobile
Faxnro	yla_faxno	Teksti	x(30)	Faksinumero(t?)	Samassa kentässä voi olla useampi	Faksi	fax
Yhteyshenkilö	yla_yhdhlo	Teksti	x(30)	Yrityksen yhteyshenkilön nimi		Yhteyshenkilöt	child_ids[0].name
Maksuehto	yla_maksueh	Teksti	**x(03)**	Maksuehtokoodi	ennaltamääriteltyt arvot	Customer Payment Term	property_payment_term
Toimitusehto	yla_toimehto	Teksti	**x(03) EI PI**	Toimitusehtokoodi	ennaltamääriteltyt arvot VAP. ASIAK	Oletustoimitusehto	property_incoterms
Toimitustapa	yla_toimtapa	Teksti	**x(03)**	Toimitustapakoodi	ennaltamääriteltyt arvot	Delivery Method	property_delivery_carrier
~~Valuutta~~	~~yla_valuutta~~	~~Teksti~~	~~x(03)~~	~~Asiakkaan käyttämä valuutta~~	EI TUODA	~~<ei ole>~~	
Edustaja	yla_edustaja	Teksti	**x(04)**	TT:n myyntiedustajan koodi	ennaltamääriteltyt arvot	Salesperson	user_id
VAT-Tunnus	yla_lvtun	Teksti	**x(10)**	Y-tunnus	Muotoa: 1234567-9 tai FI12345679	TIN	vat
<ei tuoda>				vakio: fi_FI		Kieli	lang
-	-	-	-	Odoon vaatima kenttä		Customer	
-	-	-	-	Odoon vaatima kenttä		Supplier	
-	-	-	-	Odoon vaatima kenttä		Myyntireskontra	property_account_receivable
-	-	-	-	Odoon vaatima kenttä		Ostovelat	property_account_payable
Laskutusosoite:							child_ids / type = 'invoicing'
Nimi-1	ylo_nimi[1]	Teksti	x(30)	Laskutettavan yrityksen nimi, rivi 1		Nimi	name
Nimi-2	ylo_nimi[2]	Teksti	x(30)	Laskutettavan yrityksen nimi, rivi 2		Osoite	street, jos määritetty
Osoite-1	ylo_osoite[1]	Teksti	x(30)	Laskutettavan yrityksen osoite, rivi 1		Osoite	street, jos Nimi-2 puuttuu, muuten street2
Osoite-2	ylo_osoite[2]	Teksti	x(30)	Laskutettavan yrityksen osoite, rivi 2		Zip ja Kaupunki	zip ja city
Osoitekoodi	ylo_koodi	Teksti	x(10)	Osoitekoodi	Muotoa: L01, L02. HUOM voi olla tyl	Viite	ref
Osoitettyvppi	ylo_tyvppi	Teksti	x(01)	Osoitettyvppi	Muotoa:L	Address Type	type

Kuva 4, Ote Tonni <> ESB <> Odoo taulukosta.

4.1.3 Integraation suunnittelu

Minulla oli aiempaa kokemusta integraatiojärjestelmistä ammattikoulun työharjoittelustani. Tämä tieto ja kokemus auttoivat minua integraatiojärjestelmää suunnitellessa. Integraatiosta haluttiin alusta alkaen luoda uusi myytävä palvelu, tai ainakin sitä on tarkoitus käyttää osana muitakin projekteja. Tämä takia tehtäväkseni tuli suunnitella integraation toiminta mahdollisimman modulaarisesti. Ensin jaoin integraation kolmeen osa-alueiseen: FTP-yhteys, Tonnin tietojen muunnos Odoo-muotoon ja Odooseen kirjoitus. Lisäksi Tonni-muunnoksen ja Odooseen lähettämisen väliin tuli invoker -luokka, jonka tehtävänä oli lähettää saamansa tiedot jokaiseen haluttuun luokkaan. Tämä lisättiin mahdollistamaan yhden asetuksen muuttamisella tietojen lähetyksen eri lähteisiin.

Alla olevassa kuvassa on karkean tason integraation toimintakuvaus Tonnista Odooseen. Toiminta alkaa tt-ftp.probe -luokalla, joka hoitaa luku-yhteyden Tonniin. Se löytää XML-tiedoston, siirtää sen ”tyonalla”-kansioon ja lukee sen tiedot sisään integraatioon. Tämän jälkeen xml-tiedot lähetetään tt-tonni.input -luokalle. (Kuva 3.)



Kuva 3, Integraatiojärjestelmän luokat.

4.1.4 Päätelmiä ja havaintoja

Toiminnanohjausjärjestelmien erilaisuutta ei täysin ymmärretty kehitystyön alkaessa. Projektin aloitusvaiheessa oli niin paljon erilaisia palasia ja toimintoja mietittävänä, että vaikeammat tai monimutkaisemmat ongelmat jäivät havaitsematta tai siirrettiin joko alitajuisesti tai päätöksin myöhemmin ratkaistaviksi. Moni projektin edetessä eteen tullut ongelma johtui tästä alkutöiden puutteellisuudesta.

Projektissa otettiin tietoinen riski, kun integraation pääkehittäjäksi valittiin minut, jolla ei ollut aiempaa Python -kehityskokemusta. Myös koko integraatiojärjestelmä oli täysin pilottiprojekti, joten kaikkea ei edes kyetty suunnittelemaan etukäteen. Minulla oli tosin aiempaa integraatiojärjestelmäkokemusta, josta oli apua suunnittelussa.

4.2 Vaihe 2: Ohjelmointityön aloitus

Valmistelu- ja suunnittelutyön jälkeen projekti siirtyi huhtikuussa seuraavaan vaiheeseen, eli itse ohjelmointityöhön. Työtä hankaloitti ja hidasti jo aiemmin mainittu kokemuksen puute. En ollut tehnyt mitään kehitystyötä Odoolle, käyttänyt Zatoa aiemmin, tai edes ohjelmoinut erityisen paljoa Python -kielellä (Liite 3.)

Lisäksi hidasteiksi koituivat Zaton virhetilat, joiden korjaamiseksi ei löytynyt kunnollisia ohjeita. Myös meille lähetettävän tiedon laatu oli puutteellista. Usein työ keskeytyi jo siihen, että XML-tiedostot piti korjata manuaalisesti, jotta niitä pystyi edes lukemaan sisään (Liite 3.)

4.2.1 Muuntologiikan ohjelmointi

Integraation vaativin tehtävä oli muuntaa sisään tuleva tieto Odoon ymmärtämään muotoon. Pintapuolisesti tehtävä vaikutti triviaalilta. Luodaan muuntokirjasto, jossa on kaikki sisään tulevien kenttien nimet ja Odoon vastaavien kenttien nimet. Tämä ei kuitenkaan ollut mahdollista tehdä yhden-suhde-yhteen kuin harvoissa tapauksissa. Esimerkkinä asiakkaan perustiedot, Tonnissa ne on tallennettu neljään kenttään: nimi1, nimi2, osoite1 ja osoite2. Odoossa taasen on viisi kenttää: name, street1, street2, zip ja city.

Tonnissa kentät ovat vapaita tekstikenttiä, eikä niillä ole muuta tiedoneheyssääntöä kuin se, että työntekijöille on kerrottu miten tieto pitää lisätä järjestelmään. Nimien ja osoitteiden tapauksessa muuntologiikka oli seuraavan lainen:

- nimi1 kirjoitetaan name-kenttään.
- Jos nimi2-kentässä on tietoa, se kirjoitetaan street-kenttään.

- Jos nimi2-kenttä on tyhjä, osoite1 kirjoitetaan street-kenttään.
- Jos nimi2-kentässä on tietoa, osoite1 kirjoitetaan street2-kenttään.

Lopuksi osoite2 kentässä pitäisi olla postinumero, väli, ja postitoimipaikka. Välejä saattoi kuitenkin olla useampia, ja postikoodien pituus vaihtelee maittain. Lopulta logiikaksi päätyi tämä:

- Poista välit osoite2-kentän alusta ja lopusta.
- Lue 5 ensimmäistä merkkiä, jos se on numero, kirjoita se zip-kenttään.
- Jos oli numero, lue loput osoite2-kentän merkit, poista välit merkkijonon alusta ja kirjoita tulos city-kenttään.
- Jos ei ollut numero, kirjoita koko osoite2-kenttä city-kenttään.

Lopputulos ei ole täydellinen, mutta tuhansien asiakkaiden jokaisen poikkeustilan huomioon ottaminen olisi vienyt liikaa työaikaa jo valmiiksi tiukasta budjetista. Yritin aina keksiä tavan, jolla tieto siirtyy mahdollisimman oikein ja luotettavasti. Mutta jos yksittäisen kentän poikkeukset laskettiin kymmenissä, antaa sen olla ja jätetään virheelliset tiedot käyttäjien manuaalisesti korjattaviksi.

Muuntologiikan ohjelmoinnin aikana eteen tuli paljon vastaavia ja vielä monimutkaisempia tapauksia, joissa joko tiedon eheys tai järjestelmien yhteensopimattomuus aiheuttivat päänvaivaa, ylimääräistä selvittämistä ja työaikaa. Tästä löytyy jälleen esimerkki asiakastiedoista. Tonnissa asiakkaan yhteyshenkilö on yksi vapaa tekstikenttä, jossa voi olla nimi, puhelinnumero, tai näiden kombinaatio. Odoossa taasen yhteyshenkilö on oma tietueensa, kaikkine tietoineen kuten nimi, osoite, puhelin, yritys, jne.

Oman haasteensa aiheutti minun Odoo -tietämys, tai ennemminkin sen puute. Jokaisen uuden tietuetyypin kohdalla minun piti opetella ja tutkia Odoon tietorakennetta ja kenttiä. Työtä vaikeutti erityisesti se, että Odoon dokumentointi on välillä melko heikkoa. (Liite 3.)

4.2.2 Zaton ongelmat

Monet edellä mainituista ongelmista olivat ennalta tiedossa, tai ainakin hahmotettavissa projektia suunnitellessa. Yllättävin etenemistä haittaava ongelma oli kuitenkin Zato itsessään. Kuten aiemmin kerroin, Zaton asennusta ja käyttöönottoa varten oli hyvät ohjeet ja tutoriaalit. Erityisesti ohjelmoinnin kannalta ohjeet olivat pääosin hyvät, vaikka joitain asioita ei oltukaan esitelty erityisen syvällisesti. Ongelmaksi koituivat Zaton järjestelmän antamat virheet, tai jopa toiminnan pysähtymiset, jotka tapahtuivat yllättäen öisin kun Zato ei ollut käytössä.

Virheiden syy oli aina tietokantayhteyksissä. Zato käyttää kahta eri tietokantaa. SQL-kantaa asetusten tallentamiseen ja Redis-kantaa statistiikan ja ajon aikaisten tietojen tallentamiseen. Ongelmana oli testipalvelimen MySQL -kanta, joka sulki Zaton yhteyden jos sitä ei käytetty tietyn ajan aikana. Yhteyden sulkemisen jälkeen Zato ei enää kyennyt lukemaan omia asetuksiaan ja oli käyttökelvoton. Ainut tapa korjata asia oli käynnistää Zato uudestaan, mutta uudelleenkäynnistys saattoi epäonnistua johtuen monista eri virheistä, joiden selvittämiseen ja korjaamiseen kului paljon aikaa.

Testipalvelin oli samalla web-kehityspalvelimena, joten en voinut lähteä muuttamaan tietokantaa esimerkiksi poistamalla kantayhteyksien aikakatkaisua, sillä tämä olisi hyvin nopeasti tukkinut tietokannan, joka toimi myös WordPress-kehityksen tietokantana.

4.2.3 Virheelliset XML-tiedostot

Yhtenä suunnitteluvaiheen tehtävänäni oli luoda xml-pohjat eri tietuetyypeille, jotta asiakkaan Tonni -järjestelmästä kirjoitettu xml-tiedosto on integraation luettavissa. Ongelmaksi koitui se, että Tonni -järjestelmä pyörii 80-luvun lopulta olevalla CentOS -Unix distribuutiolla, jossa ei ole minkäänlaisia nykyaikaisia kirjastoja tai ohjelmointikieliä.

XML-tiedostot olivat ongelmallisia heti alusta alkaen. Jo ensimmäisellä rivillä, missä väitettiin tiedoston olevan kirjoitettu UTF-8 koodauksella, oli väärin. Ymmärsimme nopeasti, että XML-tiedostojen kirjoittamisskripti oli tehty käsityönä, ilman mitään lopputuloksen tarkistuksia. Usein, saadessamme uuden tieto-otannan, tiedosto täytyi käydä käsipelin läpi ja korjata virheet. Tähän saattoi kulua jopa tunti, kun tiedosto oli satoja tuhansia rivejä pitkä.

XML-kenttien aloitus- ja lopetustägeillä saattoi olla eri nimet tai sitten jotkut tägeistä yksinkertaisesti puuttuivat. Erikoismerkkejä, erityisesti & -merkkejä, ei oltu kirjoitettu oikein. Muutaman kerran tiedostoissa oli sisällä korruptoitunutta tietoa. Esimerkiksi kenttä loppui kesken sanan ja jatkui muutamaa riviä alempana, ja välissä oli muita kenttiä.

Nämä virheet jatkuivat koko kehitystyön ajan ja olin itsekin yllättynyt, jos uuden tiedoston pystyi lähettämään integraatiolle ja sen luku sujui virheittä.

4.2.4 Päätelmiä ja havaintoja

Toiminnanohjausjärjestelmien tietomallien erilaisuus oli tiedossa jo projektia käynnistettäessä. Tietojen muunnos tietotyypistä toiseen oli haastava, muttei mahdoton tehtävä.

Tonnin tiedoissa oltiin myös varauduttu siihen, etteivät ne ole täysin eheitä ja ne sisältäisivät epämääräistä tietoa, olihan tietoja kertynyt useamman vuosikymmenen ajalta. Lopulta selvisi kuitenkin, että ongelma oli paljon pahempi kuin osasimme odottaa. Lähes jokainen kenttä oli vapaasti muokattavissa ja jopa numerokentissä saattoi olla tekstiä. Lisäksi XML:ää luotaessa ei ajettu minkäänlaisia tarkastuksia, joten meille tullut tieto saattoi olla lukukelvotonta ilman tiedoston manuaalista korjaamista.

Zato-integraatiojärjestelmä aiheutti omat ongelmansa. Olimme olettaneet joitakin virheitä ilmestyvän, koska kyseessä oli uusi järjestelmä, josta kenelläkään ei ollut aiempaa kokemusta, mutta virheet olivatkin pahempia kuin osasimme odottaa.

4.3 Vaihe 3: Projektin pitkittyminen

Kolmen kuukauden tehokkaan työskentelyn jälkeen projektin eteneminen alkoi hidastua huomattavasti. Heinäkuun jälkeen tehtyjen työtuntien määrä romahti (Liite 7.) Projektin alun suunnittelutyön puutteet alkoivat näkyä eri tavoin, jonka lisäksi lukuisat eri ulkoiset tekijät estivät työn tekemistä.

4.3.1 Järjestelmien yhteensopimattomuus

Tässä vaiheessa lähes kaikkien tietueiden muuntoprosessi oli vähintään osittain koodattu, mutta kaikkien Tonnin tietojen tallennus Odooseen ei onnistunut ilman uusien kenttien luontia. Tätä en itse kyennyt tekemään. Odoo-kehittäjillämme oli taasen muita projekteja jotka vaativat työtunteja ja joiden aikataulut alkoivat tulla vastaan.

Yhteensopimattomuus aiheutti myös muita ongelmia tietojen muunnoksissa ja tallentamisessa. Itselläni ei ollut täydellistä käsitystä mihin kaikkia kenttiä käytettiin tai miltä ne näyttivät tulosteissa. Käytössäni oli vain taulukko, johon oli kirjattu Tonnin kentät, kenttien tietotyyppi Tonnissa, kenttien vastaavuudet Odoon tietomallissa ja mahdolliset muunnoksessa tarvittavat lisälogiikat.

4.3.2 Vaatimusten muuttuminen

Asiakkaan tarpeita ei oltu selvitetty riittävän tarkasti projektin alussa. Tarpeiden tarkentuessa myös Tonnista tarvittiin lisää tietoa ja uusia xml-tiedostoja. Tonnin järjestelmäkehittäjällä oli kuitenkin myös omia kiireitä ja tietojen saanti osoittautui erittäin hitaaksi (Liite 3.)

Kuten jo aiemmin kerroin, kun nämä uudet XML-tiedostot vihdoin saatiin käyttöön, niissä oli lähes poikkeuksetta virheitä. Osa virheistä esti tiedon lukemisen ja vaati tiedostojen

korjaamista manuaalisesti. Osa virheistä liittyi Tonnin dataeheyden tarkistuksen puuttumisen, joka vaati suuren määrän datan tarkistamisen ohjelmointia integraation sisään, jotta prosessi ei hajoa esimerkiksi siihen, että kenttään jossa Tonnin mukaan voi olla vain numeraaleja, onkin tekstiä.

4.3.3 Muut työtehtävät

Näistä ongelmista huolimatta projekti ei olisi välttämättä edennyt paljoakaan nopeammin, sillä kesän jälkeen yrityksen toinen toimiala, web-sivusto- ja palvelukehitys, sai uutta tuulta siipiensä alle. Vastuullani oli monta eri sivustoa ja palvelua, jotka kaikki vaativat kymmeniä työtunteja ja haluttiin valmiiksi kuukauden-parin sisään.

Olin edelleen töissä osapäiväisesti opintojeni ohella, joten puolet työajastani oli varattu opiskeluun. Jouduinkin välillä joustamaan kursseilla paikallaolosta runsaastikin, jolla oli varmasti heikentävä vaikutus arvosanoihini. Olin kuitenkin varautunut tähän etukäteen, ja valinnut itselleni opiskeluprojekteissa tehtäviä jotka olivat lähellä työtehtäviäni. Tällä tavoin pystyin vähemmälläkin ajankäytöllä olla hyödyllinen tiimin jäsen opiskeluprojekteissa.

4.3.4 Päätelmiä ja havaintoja

Tonnissa oli useita kenttiä joita Odoosta ei löytynyt, ja moni Odoon vaatimista kentistä puuttui Tonnista tai ne oli tallennettu eri tietomallia käyttäen. Yksi hidaste oli uusien kenttien luonti Odooseen, jota en osannut tehdä ja heillä, jotka osasivat, oli muita kiireellisempiä tehtäviä. Projektin kehittäjillä ei muutoinkaan ollut riittävää tietoa mihin käyttötarkoituksiin kenttiä käytettiin. Muunnosten uudelleenkoodaamiseen ja tietojen tuontiin uudelleen järjestelmään kului ylimääräisiä työtunteja (Liite 7.)

Minullakin oli paljon muita tehtäviä web-kehityksessä ja kursseilla, joten integraation kehitystyö lähes pysähtyi useaksi kuukaudeksi.

4.4 Vaihe 4: Kuukausittainen kehitys

Joulukuuhun mennessä sekä asiakas että projektipäällikkö olivat lopen kyllästyneitä asiakasprojektin hitauteen ja työaikojen kasvuun. He halusivat konkreettisia tuloksia ja aikatauluja jotka myös pitivät.

Ratkaisumalliksi otettiin kevennetty SCRUM-kehitysmalli, josta otettiin käyttöön lähinnä täskitys ja Sprintit. Joka kuukauden Sprintille otetiin yksi osa-alue projektista tehtäväksi. Kaikki yhteen Sprinttiin liittyvä tekeminen pyrittiin suunnittelemaan etukäteen kerralla kun-

toon ja Sprintteihin päätettiin ottaa yksi Tonnin ja Odoon tietomalli kerrallaan. Tämän lähestymistavan toivottiin helpottavan suunnittelua ja asiakkaan tarpeiden selvittämistä (Liite 2.)

4.5 Sprint 1: Tuotteet

Ensimmäiseksi kehitettäväksi tietomalliksi otettiin tuotteet. Tuotetietoja tarvittiin jokaiseen integraatio- sekä laajemman asiakasprojektin osa-alueeseen. Lisäksi tuotetiedot olivat yksinkertaisin integroitava tietomalli, sillä sekä Tonnissa että Odoossa oli hyvin samatyyppiset kentät ja tietomallit. (Liite 4.)

Ensimmäisen sprintin päätavoite oli todellisen integraation aikaansaanti: Tonnin tuotetietomuutosten piti siirtyä reaaliaikaisesti Odooseen. Integraatioprosessi oli seuraavan lainen:
1) Tonnissa muokataan tuotetietoja ja tallennetaan muutokset.
2) Tallennuksen jälkeen luodaan tuotteen tiedot sisältävä XML-tiedosto.
3) Tiedoston luonnin jälkeen se lähetetään Zatolle.
4) Zato muuntaa tiedot ja tallentaa ne Odooseen.

Ongelmaksi koitui kuitenkin kohta 3. Tonnia pyörittävä käyttöjärjestelmä on 80-luvun lopun unix-distribuutio, jossa ei ollut tukea TCP/IP-protokollalle. Järjestelmä oli täysin suljettuna ulkopuolisesta maailmasta, joten sitä ei oltu koskaan tarvittu.

Kuukauden pyyntöjen ja yrittämisen jälkeen kävi ilmi, että tietojen reaaliaikainen lähettäminen Tonnin palvelimelta oli mahdotonta. Päädyimme vaihtoehtoiseen ratkaisuun, jossa XML-tiedosto tallennetaan FTP-palvelimelle ja Zato käy minuutin välein lukemassa sisään sieltä löytyvät XML-tiedostot. (Liite 6.)

4.5.1 Mittayksikkömuunnokset

Sprinttiä suunnitellessa tuoteintegraatioon tuli tärkeä muutos mittayksiköiden suhteen. Tonnissa useiden tuotteiden mittayksikkö on sata tai tuhat. Integraatio kuitenkin muunsi nämä kappaleiksi ennen niiden tallentamista Odooseen, kuten oli päätetty projektin suunnitteluvaiheessa. Sprinttiä suunnitellessamme tajusimme kuitenkin, että tämä yksiköiden muuntaminen aiheuttaa valtavasti päänvaivaa. Jokaisen tuotteisiin liittyvän tiedon lukemisessa tai kirjoittamisessa pitää aina tarkastaa Tonnin yksikkö ja tehdä sen perusteella määrä- ja hintamuutoksia. Lisäksi Odoossa pitää joka tapauksessa ylläpitää kahta eri mittayksikköä samanaikaisesti. Keräilyssä kappaleet ovat hyvä mitta, mutta myynti on käyttänyt sataa ja tuhatta yksikköinä vuosikymmeniä. (Liite 4.)

Päädyimme käyttämään Odoossa samoja mittayksiköitä kuin Tonnissa. Odoo ei kuitenkaan salli mittayksiköiden muuttamista jos tuotetta on käsitelty millään tavoin. Onneksemme Odoossa tuotteilla on kuitenkin sekä perusyksikkö että myyntiyksikkö. Myyntiyksikköä voi muuttaa kunhan yksikkötyypit ovat samat. Esimerkiksi jos perusyksikkönä on metri, myyntiyksikkönä ei voi olla tunti.

Ratkaisimme ongelman siten, että integraatio päivitti ainoastaan tuotteiden myyntiyksiköt. Tämän jälkeen Odoo-kehittäjämme muokkasi suoraan tietokantaan jokaisen tuotteen yksikön samaksi kuin sen myyntiyksikkö. Lopuksi integraatio kirjoitti perus- ja myyntiyksiköt, joilla tarkistimme muutokset, sillä perusyksikön vaihtamisen yrittäminen laukaisee virheilmoituksen. Yhteensä käytin tähän muutokseen yli 30 tuntia työaikaa mukaan lukien suunnittelun, virheiden selvittämisen, testaamisen ja tuotantoon viemisen (Liite 7.)

4.5.2 Tekemättömät tehtävät

Sprint 1:n aikana tuotetietojen kirjoitus ja myyntiyksiköiden muutokset saatiin tehtyä, mutta tuoteintegraatiota ei saatu käyttöön eikä tuotteille saatu kirjoitettua asiakaskohtaisia koodeja. Nämä kaksi tehtävää siirtyivät siis Sprint 2:een.

Tuoteintegraatio oli alusta asti rakennettu siten, että Zato vain kuuntelee ja Tonni lähettää sille tietoja. Kun ilmeni, että tämä on mahdotonta, suunnittelin ja toteutin ftp-lukijan, joka käy läpi ftp-kansion sisällön ja lähettää kaikki xml-tiedostot eteenpäin kuten aiemmin suunniteltiin. Tämä ftp-lukija asetettiin käynnistymään ajastetusti minuutin välein. Ongelmaksi koitui kuitenkin FTP-palvelin, joka päästi liikennettä sisään vain tietyistä IP-osoitteista. Meillä ei myöskään ollut mitään suoraa yhteydenottotapaa palvelimen ylläpitäjään joten palvelimen asetusten muuttamisessa kesti yli viikon.

Tuotteiden asiakaskoodien kirjoittaminen viivästyi aluksi sen vuoksi, että saimme uudet tuotetiedot vasta viikkoa ennen käyttöönottoa. Muu valmistelutyö Sprintin loppuvaiheessa vaati paljon työtunteja, joten lopulta päädyimme siihen, että myös niiden kirjoitus siirtyy Sprint 2:een.

4.5.3 Päätelmiä ja havaintoja

Tarkat vaatimukset ja työmääräarviot sekä Sprintin mukainen tehtävien pilkkominen noin kahden tunnin pituisiksi kokonaisuuksiksi auttoivat suunnittelussa ja aikataulussa pysymisessä. Saimme lähes kaiken suunnitellun työn tehtyä ajallaan ja jopa vähemmällä työllä kuin arvioitu. (Liite 6.)

Sprint 1 ei päässyt täysin tavoitteisiinsa johtuen sekä ulkoisista että sisäisistä tekijöistä. Integraation kriittisin osa, tiedonvälityskanava, muuttui vain viikkoa ennen suunniteltua käyttöönottoa. Lisäksi kaikkia vaadittavia tieto-otantoja ei toimitettu meille riittävän aikaisin. (Liite 6.)

4.6 Sprint 2: Myyntitilaukset

Sprint 2 alkoi laajuuden määrittämisellä ja tehtävien suunnittelulla. Päätimme, että Sprint 1 myöhästyneiden töiden lisäksi laitamme myyntitilausten ja lähetteiden koko käsittelyprosessin kuntoon. Prosessi alkaa siitä, kun Tonnissa myyjä luo uuden myyntitilauksen ja tallentaa sen. Tonni luo myynnistä XML-tiedoston ja tallentaa sen FTP-palvelimelle. Seuraavaksi integraatio lukee XML-tiedoston, muuntaa sen Odoo-muotoon ja kirjoittaa sen Odooseen.

Keräilyjärjestelmä käyttää Odoon tietoja ja muodostaa uuden lähetteen. Varaston työntekijä kerää lähetteen rivit ja tallentaa sen kerätyksi. Kerätyksi merkitseminen käynnistää Odoossa tapahtuman, jolloin Odoo lähettää keräilyn tiedot integraatiolle, joka muuntaa ne Tonnin XML-muotoon ja kirjoittaa uuden XML-tiedoston FTP-palvelimelle.

4.6.1 Myöhästyneet tehtävät

Ennen myynti- ja keräilyintegraatiota piti kuitenkin saada FTP-lukija ja -kirjoittaja toimintaan. Kirjoitin ensin testit, joilla tarkistettiin että kaikki tarvittavat kirjoitus- ja lukuoikeudet sekä kansiot olivat olemassa ja toiminnassa FTP-palvelimella. Kuitenkin vasta kolme viikkoa Sprint 2:n alkamisesta saimme FTP-yhteyden auki. Yhteyden avaamisen jälkeen kaikki sujui kuitenkin hyvin ja tuoteintegraatio oli vihdoin toiminnassa.

Toinen myöhästynyt osa oli tuotteiden asiakaskoodit. Testiajojen aikana huomasin, että meiltä puuttui suurin osa uusien tuotteiden asiakaskoodien asiakkaiden tiedoista. Tämä aiheutti ketjureaktion ja lopulta näitä uusia tuotteita ei kirjoitettu Odooseen, sillä niiden tarvitsemaa tietoa ei ollut olemassa. Uudet asiakastiedot saapuivat vasta neljä viikkoa Sprintin aloituksen jälkeen, mutta niitä odotellessa integraation muunnoslogiikka oli jo valmis ja hyvin testattu. Ajoin uudet asiakastiedot sisään ja testipalvelimella myös tuotteiden asiakaskoodit kirjoitettiin ongelmitta Odooseen.

4.6.2 Kehitystyön eteneminen

Myöhästyneiden tehtävien lomassa myös työ Sprintin lopullista tavoitetta eli myyntitilausten integraatiota varten jatkoi etenemistään. Valtaosa työstä oli jo tehty aiemmissa kehitysvaiheissa ja nyt piti varmistaa, että kaikki tarvittava tieto tallentuu oikein. Suunnittelin keräilyjärjestelmän kehittäjän kanssa testitiedostoja ja tehtävälistaa kenraaliharjoitusta varten. Toinen, aiemmin tekemätön vaihe oli Odoo-datan muunto Tonnin XML-pohjan mukaiseen muotoon ja XML-tiedoston kirjoittaminen FTP-palvelimelle. Tämä saatiin hyvän selvitys- ja suunnittelutyön sekä kirjoitettavan tiedon vähyyden vuoksi nopeasti tehtyä ja testattua. Myös FTP-yhteyden virheidenhallinta parani erityisesti suurien tietomäärien kanssa.

Ongelmiakin oli, sillä Sprintin puolivälissä ilmeni, että integraatio tallensi tilauksen käsittelijäksi asiakkaan myyjän eikä suinkaan myynnin luonnista vastanneen henkilön tiedot kuten lähetteellä piti olla. Tämä sekaannus johtui pääasiassa siitä, ettei kenttien todellista käyttötarvetta oltu selvitetty riittävän hyvin, vaan oletimme, että "myyjä"-nimisessä kentässä on myynnin tekijän tunnus. Myyntitiedon muutos vaati Tonnin XML-generoinnin muuttamista ja uuden XML-pohjan. Saimme uuden testipohjan kuitenkin yllättävän nopeasti ja kun kerroin, että se toimi hyvin saimme myös valtavan otannan kahden vuoden myyntitietoja.

4.6.3 Ensimmäinen kenraaliharjoitus

Sprintin loppua kohden alkoi näyttää siltä, että pysymme lähes aikataulussa. Aloimme suunnitella esimerkkitietoja ja tein niille XML-tiedostot, jonka lisäksi tein skriptitiedostot joilla tiedot pystyi lähettämään integraatiolle. Viimeistelimme myös yksityiskohtaiset suunnitelmat kenraaliharjoitusta varten, jonka oli tarkoitus simuloida moduulien asennusta, konfiguraatiota sekä asiakkaiden ja myyntien viemistä tuotantopalvelimelle.

Ensimmäinen osa harjoitusta oli asiakastietojen sisäänkirjoitus. Tämä onnistui hyvin ja ongelmitta. Seuraava osa oli myyntitietojen sisäänkirjoitus, joita oli valtavasti. Tiedostot oli jaettu kuuteen osaan ja jokaisen kirjoittamiseen meni n. 30 minuuttia. Alussa vastaan tuli virheitä, mutta niiden korjaamisen jälkeen tietueet menivät sisään ilman ongelmia. Harjoituksen aikana ilmeni, että Tonnista on mahdollista poistaa asiakkaita tai tuotteita vaikka vanhat myynnit saattoivat viitata niihin. Meidän piti siis luoda muutama tyhjä asiakas ja tuote, jotta integraatio pystyi kirjoittamaan myyntihistorian virheittä.

4.6.4 Äkkipysähdys

Juuri kun olimme saaneet kenraaliharjoituksen tehtyä, käyttöönottosuunnitelman valmiiksi ja koulutukset sovittua selvisi, että meiltä puuttuivat täysin Tonnin varastosaldon muutokset. Asiakasprojektin pääkehittäjät olivat keräilyjärjestelmään ja integraatioon keskittyneitä, joten tämä hallinnan tarve oli mennyt meiltä täysin huomiotta tai olimme olettaneet, että saldomuutostieto liikkuu lähetteiden mukana.

Yhtäkkiä käyttöönottosuunnitelmat peruttiin ja jäimme taas odottamaan uutta tietoa. Saimme nämä kahden viikon kuluttua. Onneksemme niissä käytettiin samaa pohjaa kuin aiemmissakin myyntitilauksissa. Lisäksi nyt sekä saldomuutokset että myös asiakastietojen ja myyntien muutokset kirjoitettiin FTP-palvelimelle automaattisesti muutosten jälkeen, joten kaikki oli valmiina integraation käyttöönottoa varten.

Asiakkaan myyjien aikataulut eivät kuitenkaan enää sopineet yhteen aikataulujemme kanssa, joten koko projekti laitettiin jäihin siksi aikaa kunnes heillä oli taas vapaata aikaa. Sillä välin kävin läpi kaikki siihen hetkeen saakka integraatiossa esiintyneet virheet ja korjasin niistä jokaisen. Nyt projekti oli siinä vaiheessa, että Sprint 2:n tulokset voidaan ottaa käyttöön hyvinkin nopealla aikataululla.

4.6.5 Käyttöönottoon valmistautuminen

Lopulta, kuukauden odotuksen jälkeen, asiakkaalta löytyi tarvittavaa aikaa ja innostusta ottaa järjestelmä käyttöön. Väliajalla integraatio oli kytketty pois päältä, sillä tuotanto-Odoo ei kyennyt vastaanottamaan tuotteiden asiakaskoodeja, joka aiheutti virheen jokaisen tuotteen sisäänkirjoituksessa.

Tonni kuitenkin loi XML-tiedostoja koko tämän ajan, joten ne piti käydä hakemassa FTP:ltä. FTP:ltä piti myös hakea ja jäsennellä kaikki aiemmin virheitä aiheuttaneet XML-tiedostot, joita oli kertynyt yli tuhat kappaletta. Lähetin kaikki nämä XML-tiedostot integraation kautta testijärjestelmään ja kirjasin käyttöönottosuunnitelmaan kaikki esiin tulleiden virheiden tarvitsemat korjaukset.

Pidimme asiakkaan myyjille ja varastomiehille uuden järjestelmän koulutuspäivän keskiviikkona ennen käyttöönottoviikonloppua. Koulutuspäivän aikana ilmeni selvästi, että osalla myyjistä oli suurtakin muutosvastarintaa. Olimme kuitenkin varautuneet suurimpaan osaan heidän huolenaiheistaan. Oli myös monia myyjiä jotka odottivat innolla uutta järjestelmää, joka myös osaltaan rauhoitti muiden huolia.

4.6.6 Käyttöönotto

Käyttöönotto tapahtui viikonloppuna. Käynnistimme työt perjantaina klo 15:45. Pyysin Tonnin kehittäjältä uusimmat tieto-otannat myynneistä ja asiakkaista alkaen edellisestä otannan luonnista. Kävimme läpi käyttöönottosuunnitelman kohta kohdalta. Lopulta, noin klo 20 aikoihin, olimme käyneet läpi kaikki tarvittavat toimenpiteet. Tämän jälkeen pääsin aloittamaan XML-tiedostojen tuonnin integraation kautta tuotantoon. XML-tiedostoja oli yhteensä yli 70 megatavun verran. Nämä sisälsivät kaikki asiakastiedot ja tuotetiedot sekä myös myyntitiedot edelliseltä kahdelta ja puolelta vuodelta.

Olin ilmeisesti unohtanut kirjata osan tekemistäni muutoksista testi-Odooseen, sillä integraatio heitti muutamia virheitä tuontien aikana. Nämä eivät olleet mitään vaikeita korjattavia, sillä tuotantojärjestelmästä puuttui muutamia vaadittavia tuotteita tai asiakkaita jotka oli poistettu Tonnista, mutta Odoo vaati ne.

Nämä virheet kuitenkin työllistivät minua ja veivät ylimääräistä aikaa. Integraatiolle pystyi lähettämään XML-tiedostoja noin 45 minuutin välein, mutta käytännössä lähetysväli oli pidempi sillä en ollut yhtäjaksoisesti töissä viikonloppuna. Lisää harmaita hiuksia aiheutti integraatiopalvelimen täydellinen romahtaminen lauantai-iltana, eikä sitä saatu korjattua kuin vasta sunnuntai-aamuna. Syynä oli ilmeisesti palvelimen ylikuormitus. Lopulta kaikki tiedot saatiin vietyä tuotantoon sunnuntai-iltana. (Liite 7.)

Järjestelmän käyttöönotto asiakkaalla tapahtui heti maanantai-aamusta. Olimme paikan päällä valmiina tukemaan käyttäjiä uuden järjestelmän käytön kanssa ja korjaamaan mahdollisia esiin tulevia virheitä. Muutamia virheitä tulikin. Suurimpana oli se, että käytimme vääriä käyttäjätunnuksia. Emme olleet koskaan saaneet Tonnista käyttäjätunnuksia, ja käyttöönoton jälkeen selvisi, että jokaisella käyttäjällä saattoi olla useampi kuin yksi tunnus Tonnissa. Selvisimme kuitenkin ensimmäisestä päivästä hyvin ja käyttäjiltäkin tuli hyvää palautetta.

Ensimmäisen viikon aikana käyttöönoton jälkeen tein virheraportin asiakasyritykselle, jossa kerroin, että integraatio ei ollut toiminut ainoastaan viidessä eri tilanteessa. Yhdessä tapauksessa Tonnin käyttäjätunnusta ei löytynyt yhdeltäkään Odoo-käyttäjältä. Kahdessa tapauksessa tuotteen myyntiyksikköä oli muutettu Tonnissa, joka ei Odoossa ole mahdollista. Lisäksi oli 6 tapausta, joissa jo kerran hyväksyttyä myyntiä oli muokattu, jota Odoo ei myöskään salli. Asiakasyrityksen johtaja oli erittäin tyytyväinen lopputulokseen ja uuden järjestelmän käyttöönoton kivuttomuuteen.

4.6.7 Päätelmiä ja havaintoja

Sprint 2 aikana ilmeni samanlaisia väärinymmärryksiä ja tiedon eheyden ongelmia kuin aiemminkin projektin aikana. Meillä ei ollut missään vaiheessa täyttä ymmärrystä Tonnin toiminnasta, eivätkä käyttäjät muistaneet kertoa jokaista mahdollista tarvettaan kuin vasta sitten, kun luulimme olevamme valmiita.

Pahin ongelma oli projektin etenemisen täysin pysäyttänyt saldotietojen muutosten puuttuminen. Mutta kun saimme tämän asian otettua huomioon, itse käyttöönotto sujui lähes ongelmitta johtuen perusteellisesta suunnittelutyöstä. Asiakkaan palaute järjestelmästä on ollut pääosin positiivista, vaikka kehitettävää on edelleen.

4.7 Jatkokehitys ja arviointi

Projektin kehitystyö tulee jatkumaan käyttöönoton jälkeenkin, kun asiakas ryhtyy siirtymään vanhasta järjestelmästä uuteen. Horisontissa on suurimpana tulevana asiana myynti- ja laskutustyön siirtäminen vanhasta järjestelmästä uuteen. Keräilyjärjestelmän käyttöönotto sujui suhteellisen kivuttomasti ja tärkeimpänä tekijänä oli kaikkien kehittäjien mukaan huolellinen ja tarkka suunnittelutyö ja töiden aikataulutus.

Projektin suurimmat haasteet olivat sisään tulevan tiedon heikko laatu ja käyttäjätarpeiden tietämyksen puute. Projekti kesti sekä kalenterin että käytettyjen tuntien mukaan paljon pidempään kuin suunniteltu. Käyttöönotto kuitenkin sujui lopulta hyvin ja yllättävän kivuttomasti.

Projektin tekninen suunnittelu tehtiin järjestelmän kehittäjien eikä käyttäjien näkökulmasta. Tämä kostautui erityisesti käyttöönotoissa, kun esiin tuli yllättäen uusia tarvittavia tietoja tai olimme linkittäneet tiedot vääriin kenttiin. Näitä virheitä tuli esiin myös jo kerran hyväksytyistä tulosteista. Osa näistäkin ongelmista johtui sisään tulevan tiedon heikosta laadusta. Vaikka olimme varautuneet siihen, emme olleet ymmärtäneet kuinka olematonta tiedon eheyden tarkistus Tonnissa on.

Projektin loppuvaiheessa kävi selväksi, että Sprint-tyylinen etenemismalli soveltui projektiin erinomaisesti. Sitä tulisi käyttää jatkossa tässä ja varmasti myös muissa yrityksen projekteissa, erityisesti suuremmissa ja monimutkaisissa projekteissa.

5 Pohdintaa

Projektin tuntikirjauksista näkyy selkeästi, kuinka hajautunutta ajankäyttö ja projektin edistyminen oli projektin aikana. Tuntikirjauksista käy myös selvästi esille, että suurimmat aikasyöpöt ovat olleet Tonnista integraatiolle tuotavan tiedon ohjelmointi, kun taasen integraatiosta Odooseen tai Odoosta integraatiolle oli käytetty vain murto-osan verran aikaa. (Liite 7.) Tämä johtuu pääosin Tonnin tiedon eheyden puutteista.

Projektin jäsentely vaiheisiin ja niiden havainnointi omina osinaan oli hyvä valinta tutkimuksen esitystavaksi, sillä vastaukset tutkimuskysymyksiin alkoivat tulle esiin lähes itsestään. Vaiheistus auttoi myös nostamaan esille kuinka projektin eteneminen on vaihdellut vauhdikkaan ja hitaan välillä (Liite 7.)

Haastatteluista selvisi hyvin muidenkin tekijöiden mielikuvia ja mielipiteitä projektista. He toivat esille monia näkökulmia, joita en itse ollut tullut ajatelleeksi. Lisäksi selvisi, että meillä kaikilla oli hieman erilaiset näkemykset ja tiedot integraatioprojektista. Haastateltaville olisi mahdollisesti kannattanut kertoa, että he voivat myös antaa minun tekemistä töistä avoimesti kritiikkiä, sillä siitä olisi ollut hyötyä tutkimuksessa. (Liite 2; Liite 3.)

5.1 Mitkä olivat kriittisimmät virheet projektin aikana?

Projektin aikana esiintyi useita erilaisia virheitä, vääriä olettamuksia ja hidastuksia. Pyrin jokaisen löytämäni virheen kohdalla kuitenkin kysymään, oliko tämän virheen pohjalla joku muu laajempi tai aiempi virhe. Päädyin lopulta kolmeen kriittiseen virheeseen, jotka olivat mielestäni aiheuttaneet projektin ongelmat:

- Tietomallien yhdistämisen ja käyttäjien tarpeiden riittämätön selvitys
- Sisään tulevan tiedon heikko laatu
- Projektin etenemisen vaatimien töiden viivästykset

5.2 Miten virheisiin reagoitiin ja olisiko ne voitu estää?

Virheisiin reagoitiin ja niistä selvittiin eri tavoin ja erilaisilla menestyksillä. Menestys näytti riippuvan siitä, kuinka tekninen virhe oli kyseessä. Epäilen tämän johtuneen siitä, että projektin kehittäjillä oli enemmän teknistä kuin projektinhallinnan kokemusta. Trendinä oli kuitenkin selkeästi se, että Sprint-kehitykseen siirryttäessä toiminnasta tuli paljon tehokkaampaa ja nopeammin reagoivaa.

5.2.1 Tietomallien yhdistämisen ja käyttäjien tarpeiden riittämätön selvitys

Lasken tämän virheen alle sekä oman tietämykseni puutteet että tietojen yhdistämisdokumentoinnissa tehdyt virheet. Tiedon muunnosten ohjelmointi oli minun vastuulla, mutta koska minulla ei ollut täyttä käsitystä Tonnin tai Odoo-tietomalleista, tietomuunnos tai tallennuskohde ei ollut aina oikein. Yhdistämisdokumentti oli laadittu teknilliseltä pohjalta, eikä käyttäjien tarpeita oltu otettu kunnolla huomioon. Jokaisessa projektin käyttöönoton vaiheessa tuli vastaan tilanteita, joissa olimme laittaneet esille väärää tietoa, turhaa tietoa tai jättäneet tärkeää tietoa pois.

Virheellisiin tietoihin reagoitiin sitä mukaa, kun joku huomasi ne. Tämä saattoi tapahtua asiakkaan, muiden projektin tekijöiden tai minun toimesta. Projektin kannalta epämiellyttävintä oli, jos asiakas huomasi virheitä samalla, kun esittelimme, kuinka olimme edenneet projektissa.

Ennen kuukausittaiseen kehitykseen siirtymistä pyysimme malliesimerkkejä tulosteista, joita asiakas käytti vanhassa järjestelmässä. Esimerkkien jokaisen tiedon viereen oli kirjattu, mistä Tonnin kentästä tieto tulee. Ryhdyimme niiden pohjalta katsomaan mitkä Odoon kentät ja tiedot vastaavat niitä. Tämä oli erittäin tehokas ja toimiva tapa sekä integraation että keräilyjärjestelmän suunnittelun kannalta. Vaikka tämä tapa ei estänyt kaikkien virheiden syntymistä Sprint 2:n loppua kohden, olen silti varma että virheitä oli vähemmän kuin mitä niitä olisi ollut ilman tätä tapaa.

5.2.2 Sisään tulevan tiedon heikko laatu

Sisään tulleessa tiedossa oli kahdenlaisia laatuvirheitä. Ensinnäkin oli tiedon eheyden ongelmia, sillä minkään kentän sisällöstä ei voinut tehdä mitään olettamuksia: jopa numerokentissä saattoi tulla sisään tekstiä. Toiseksi, tieto tuotiin sisään XML-tiedostoissa, joissa saattoi olla tiedon lukua estäviä XML-standardien vastaisia virheitä, esimerkiksi kenttiä ei oltu suljettu tai & -merkkejä ei oltu merkattu oikein. Tämä virhe heijastui osittain myös edellisiin virheisiin, erityisesti kun Tonnissa oli tallennettu eri tietoja samaan kenttään.

Projektin edetessä kerroimme yhä voimakkaammin, kuinka tärkeätä XML-tiedostojen tarkistaminen on, koska virheet piti korjata käsin. Lopulta saimme XML-pohjat kuntoon ja Tonnin kehittäjän XML-kirjoittimet korjattua. Heikon tiedon kanssa yritin mahdollisuuksien mukaan automaattisesti muuntaa ja korjata sitä Odoon muotoon. Integraation koodiin piti laittaa suuri määrä try-except -lauseita jopa triviaaleihin paikkoihin, jotta integraatioprosessi ei pysähdy kehnon tiedon takia. Päätimme jo varhaisessa vaiheessa, että minun

pitää korjata vain kriittiset virheet ja pyrkiä siirtämään huonolaatuinen tieto muuttumattomana, jolloin asiakkaan tehtäväksi jäi sen muokkaaminen parempaan muotoon.

Pitkään käytössä olleen järjestelmän tietojen siirtämisessä tulee aina tiedon eheyden kanssa ongelmia. Kuten tässäkin projektissa huomattiin, aidolla testidatallakaan ei voi millään ennustaa kaikkia virheitä. Tiedon eheyden kanssa täytyy tasapainoilla sen kanssa, kuinka paljon aikaa ja resursseja kannattaa käyttää virheiden korjaamiseen. Sisään tulevien tiedostojen virheiltä olisi voinut suojautua pakottamalla kaikki tiedostot käymään läpi tarkastus ennen niiden lähettämistä integraatiolle. Tämä olisi tosin voinut hidastaa uuden tiedon saantia vielä enemmän.

5.2.3 Projektin etenemisen vaatimien töiden viivästykset

Projektin keskivaiheen töksähtelevä eteneminen johtui pääosin odottelusta. Integraatiota ei voinut lähteä luomaan ennen tietojen saantia Tonnista, eikä niiden kirjoitusta Odooseen voinut kunnolla testata ennen Odoon konfiguraation muutoksia. Eikä keräilyjärjestelmääkään voitu testata riittävästi ennen integraation ajoa. Osa viivästyksistä johtui Tonnin kehittäjän omista työmääristä, osa johtui asiakasprojektin muiden tekijöiden kiireistä ja osa omista kiireistäni.

Viivästyksiin reagoitiin aluksi hitaasti: projektilla oli vielä paljon aikataulua jäljellä, ja kaikilla painoivat muut työt ja projektit päälle. Tonnin kehittäjältä pyydettiin tietoja tai muutoksia sähköposteilla, jotka sitten unohtuivat jopa kuukaudeksi.

Minun olisi varmasti pitänyt olla aktiivisempi näiden viivästyksien kanssa, erityisesti informoimalla tehokkaammin niistä eteenpäin. Meidän olisi myös pitänyt pyytää tietoja ja muutoksia Tonnin päässä voimakkaammin. Projektin loppua kohden myös asiakkaan painostuksesta johtuen asioita saatiin tehtyä paljon nopeammin kuin projektin alkuvaiheissa.

5.3 Kehittämis- ja jatkotutkimusehdotuksia

Integraatioprojektin kehittämisehdotuksista voi helposti kirjoittaa kokonaisen toisen opinnäytetyön, joten rajaan kehittämisehdotukset koskemaan projektin jatkokehittämisen suunnittelua. Kuten aiemmin pohdin, projektin virheiden ja hidastusten taustalla oleva punainen lanka on tiedon puute ja olettamuksien teko. Etenkin tekniseen näkökulmaan perustunut suunnittelu on ilmennyt huonoksi tavaksi, ainakin käyttöönottojen aikana. Väitänkin, että kaikkea tulevaa projektin tekemistä kannattaa suunnitella käyttäjien tarpeiden perusteella, käyttäen heidän aidosti käyttämiä, tai käyttöön tulevia, tulosteita, raportteja ja näkymiä. Käyttäjillä ei välttämättä ole kiinnostusta, saati sitten tietämystä, mitä tietomalleja

ja kenttiä he tarvitsevat. Mutta kun heille annetaan käteen paperi ja kysytään voivatko he käyttää tätä työssään, palaute on innostunutta ja välitöntä.

Mielestäni hyvä jatkotutkimuskohde on tarkastella miten projektin eteneminen on sujunut kun kehitystä on jatkettu tulevaisuudessa. Erityisesti suunnittelu- ja toteutusmetodien vertailu sekä käyttäjätyytyväisyyden mittaaminen olisi hyvä selvittää. Selvitys kertoisi olivatko tämän työn väittämät tosia ja auttoiko niiden käyttöönotto projektia.

5.4 Opinnäytetyöprosessi

Opinnäytteeseen meni lopulta aika lailla tasan neljä kuukautta aikaa. Onnekseni sain tehdä opinnäytetyötäni töiden ohella, eikä kirjoitus- ja selvitysprosessi vienyt liikaa vapaa-aikaa tai voimavaroja. Kirjoitustyön aikana minulla oli tehokkaita ja tehottomia ajanjaksoja, mutta en havainnut itsessäni samoja stressitasoja kuin joillain opiskelijakavereistani heidän kirjoittaessaan omaansa. Olin miettinyt työni aihetta pitkän aikaa ja nyt vaikuttaa siltä, että aiheeni soveltui minulle täydellisesti.

Opinnäytteen tekemisessä oli kolme eri vaihetta. Aluksi oli tietoperustan selvitys ja kirjoittaminen, samanaikaisesti kirjoitin myös projektin etenemisestä käyttäen apunani tuntikirjauksia. Toisessa vaiheessa kävin läpi etenemistarinaa ja etsin havaintoja jokaisesta projektin työvaiheesta. Lopuksi pohdin havaintojani ja viimeistelin työn kirjoitus- ja ulkoasun.

Tein opinnäytetyötäni samaan aikaan projektin kanssa, joten en edes tiennyt projektin lopputulemaa aloittaessani työtäni. Tämä on antanut lisää motivaatiota analyysien tekoon, sillä olen parantanut omaa työjälkeäni samaan aikaan. Projektin käyttöönotto oli itse asiassa vain kolme viikkoa ennen työni palautusta joka aiheutti aikataulupaineita tälle työlle. Kuten aiemmin mainitsin, esimieheni antoivat minun kirjoittaa myös työajalla joka helpotti varsinkin työn loppuvaiheessa.

Tietoperustaa tutkiessani ja kirjoittaessani opin myös monia uusia asioita joista on ollut hyötyä projektin aikana. Opinnäytettäni tullaan käyttämään osana projektin onnistumisen ja etenemisen analyysia, josta olen innoissani. Mielestäni työni vastaa hyvin ja perustellusti sen esittämiin kysymyksiin.

6 Lähteet

Ryan, J. 2011. Rethinking the ESB: Building a simple, secure, scalable Service Bus with an SOA Gateway. (www.computerworld.com/article/2510364/infrastructure-management/rethinking-the-esb--building-a-simple--secure--scalable-service-bus-with-a.html). Computerworld.

Odoo. 2016. Open Source ERP and CRM | Odoo. (www.odoo.com). Luettu: 23.4.2016.

PMBOK. 2008. A guide to the Project Management Body of Knowledge (PMBOK Guide). Project Management Institute, Inc. Newtown Square.

Tech-FAQ. 2016. ERP (Enterprise Resource Planning). (www.tech-faq.com/erp.shtml). Luettu: 9.5.2016.

Turner, D. 2010. Qualitative Interview Design: A Practical Guide for Novice Investigators. Teoksessa Nova Southeastern University . The Qualitative Report Volume 15 Number 3, s.754-760. Nova Southeastern University. Fort Lauderdale.

Weckström, E. 2014. Projektinhallinta: Case oldTimerTimer. Opinnäytetyö. Hämeen ammattikorkeakoulu.

Yin, R. 2009. Case Study Research. SAGE Publications, Inc. Thousand Oaks.

Zato. 2016. Open-source ESB, SOA, REST, APIs and Cloud Integrations in Python. (https://zato.io/docs/index.html). Luettu: 7.5.2016.

7 Liitteet

Liite 1: Haastattelukysymykset

1. Mikä on mielestäsi integraatioprojektin tarkoitus?
2. Kuinka integraatioprojekti liittyy omiin tehtäviisi?
3. Oletko saanut mielestäsi riittävästi tietoa integraatioprojektin etenemisestä?
4. Onko integraation etenemisen nopeus tai hitaus haitannut omaa tekemistäsi?
5. Oletko tyytyväinen integraation tämänhetkiseen toimivuuteen?

Liite 2: Projektipäällikkö Tuomon haastattelu

Kysymys 1

P: Mikä on mielestäsi integraatioprojektin tarkoitus?

T: Integrattioprojektin tarkoitus on kytkeä kaksi tietojärjestestelmää toisiinsa sillä ajatuksella että meillä täytyy olla jonkinnäköinen master data paikka ja se vanha järjestelmä jossa on tiettyjä kyvykkyyksiä ja uusi järjestelmä josta puuttuu näitä kyvykkyyksiä joita me halutaan tällä tavoitella.

Jos nämä järjestelmät olisi yksinänsä niin niitten tietosisällöt ei vastais toisiansa eli käytännössä ne on pakko kytkeä toisiinsa

Kun ollaan tekemässä keräilyä, niin tarvitaan asiakastietoa, myyntitiedot lähetteitä varten. Se että pääsemme irti manuaalisesta syötöstä ja päästään moderniin nykyaikaan eli kytketään tietojamme toisiinsa ja pystytään hyödyntämään niitä tietoja ristiin. eli lopputulema on se että meillä on kaksi järjestelmää jotka toimii rinnan niin että toinen toimii master datana eli hallinnoi käytännössä asiakastietoja tuotetietoja ja myyntejä. Toinen järjestelmä, joka nyt sitten lähinnä nyt sitten päivittää keräilydataa eli kahdensuuntainen integraatio siinä mielessä että vaan ne keräilydatat palautetaan sinne vanhaan Tonni-järjestelmään

Kysymys 2

P: Sinun tehtävät tässä isommassa asiakasprojektissa on...?

T: Projektipäällikkö, eli yhteyshenkilönä asiakkaaseen nähden. Integraatioprojekti on yksi osa sitä. Tässä on tällä hetkellä käytännössä käynnissä, vähän riippuen laskentatavasta, kolmesta viiteen projektia asiakkaalle, joista tähän integraatioon liittyy nyt suoraan kolme, mukaanlukien integraatio. Eli varastonhallintapuoli sekä keräilytoiminnallisuus on suoraan

liitoksissa, mutta myöhemmässä vaiheessa tuotannonohjaus tulee myös oleman merkittävä osa sitä. Käytännössä ollaan siirtymävaiheessa, jossa vanhassa järjestelmässä hoidetaan niin pitkään kuin nyt on mahdollista niitä asioita mutta pikkuhiljaa niitä aletaan tiputtelemaan Odoon suuntaan ja sitä varten meillä pitää olla se data molemmissa järjestelmissä. Oikeastaan haasteen aiheuttaa sen että Tonnista pitäisi saada kaikki ne muutokset siirrettyä tonne Odoon puoleen ja tarkottaa myös sitä että Odoon päässä ei saa muutoksia edes tehdä.

T: Hankevastaavana tässä, mutta koska on kyse sen veran kuitenkin vielä pienestä projektista niin projekteilla ei ole omia projektipäälliköitä, vaan toimin näissä kaikissa projektipäällikkönä, vastaan määrittelypuolesta ja osittain laadunvarmistuspuolesta. Integraatioprojekti nyt sitten oikeastaan aika sinänsä yksinkertainen, se on nyt sitten lähinnä Pekon harteilla ollut tässä, ja saanut tarvittaessa tukea Mikulta ja Atelta.

P: Integraatioprojekti on siis ollut liimana eri projektien välissä.

T: Kyllä. Nimenomaan näin ja erityisenä haasteena on ollut se että olen pyrkinyt varmistamaan että kaikkien vaatimukset mitä keräilyllä, varastonhallinnalla ja mahdollisesti myöhemässä vaiheessa tuotannonohjausella on tietojen suhteen, että ne tulisivat huomioitua integraatioprojektissa. tosissaan vaatimuksia siihen projektiin tulee muilta projekteilta jotka sitten asettavat haasteita kieltämättä. Pystyy synkkaamaan ja keskustelemaan keskenään minkälaisia tarpeita niihin liittyy.

Kysymys 3

P: Oletko saanut mielestäsi riittävästi tietoa integraatioprojektin etenemisestä?

T: Sanotaanko että, kohtalaisen hyvin pysyny kartalla, mutta mitä nyt ollaan enemmän tuomassa tälläsiä säännöllisiä tapaamisia ja tilannekatsauksia eli siirrytään tälläiseen kuukaisittaiseen sprint-malliin jossa sitä keskustelua käydään enemmän myös asiakkaan kanssa. Vähän omienkin kiireiden takia näitä tapaamisia tai näitä DevDaytä ei järjestetty niin säännöllisesti alkuun, mutta jatkossa tarkoitus päästä siihen rytmiin jälleen että pystytään hyödyntämään sitä yhdessä tekemistä ja kaikki pysyy kartalla missä mennään ja seuraavaksi mitä oletamme saatavan tehtyä tai valmiiksi seuraavan kuukauden aikana sit aina katse on vähän seuraavassa kuussa mitä ollaan tekemässä sillon.

P: Eli on säilyny sellanen tieto että mikä, kuvitteellinen tieto ainakin, missä kohti on...

T: Kohtalaisen hyvin, mutta kyllä siinäkin parantamisen varaa on, omaakin syytä että pitäisi projektien tai enemmän pitää huolta siitä et kaikki on samalla viivalla.

Kysymys 4

P: Onko integraation etenemisen nopeus tai hitaus haitannut omaa tekemistäsi?

T: Ehkä siinä mielessä että tota nyt ainakin sitä integraatioprojektia on hirveesti hidastanut se Tonni-kehittäjän tekeminen eli sen toisen järjestelmän pään tekeminen, et sitä on nyt odoteltu vuosi eikä vieläkään olla, ollaanko 50%ssa. Viimeistelyihin menee aina yllättävän paljon aikaa.

P: Kyllä kyllä. Siinäkin muistaakseni oli tämmönen että se tieto, ainakin mun kohdalla, mitä tietoja tarvitaan Tonnin päästä sisään ne on vaihdellu tässä projektin edetessä: on tullu mukaan sellasta mitä ei alkuun tarvittu, ja toisittain minkä takia nämä tiedot on nyt täällä mukana.

T: Joo, ja tuosta liittyy siihen, mitä pidempiä projektit on, sitä enemmän tulee muutoksia yleensä. Me ei olla tässä ehkä muutoksien näkökulmasta, tehty vaatimusmäärittelyjä ihan tarpeeksi pitkälle, jolloin päädytty siihen tulokseen että tehdään asioita, tarkastetaan ja todetaan että meiltä puuttuu jotain. Et olis voinu sen määrittelyn tehdä tarkemmin, sit vast lähteä tekemään asioita. Tosin mä en usko että me ollaan ihan mahdottomasti turhaakaan tehty, siinä mielessä kenttien lisääminen tai poistaminen on yleensä aika triviaalia.

P: Siinä on ehkä enemmän ollu sellanen, että kun mulla itsellä ei ole ollut riittävästi tietoa Odoosta, sen sisällöstä, miten sitä käytetään, niin siinä on tullu muutamaan kerran sellanen, että on pitäny tehdä jotain uusiks mikä on jo tehty.

T: Joo. Toi on se mitä mä yritin sillä, että parittaa Attea ja sua keskustelemaan ja kattomaan kenttä kentältä, että mistä löytyy mitäkin joka on aiheuttanu vähän myös sitä hidastumista siihen että, kun sulal ei oo täyttä ymmärrystä mitä Odoo tekee, Atella ei ole täyttä ymmärrystä mitä integraatio tekee ja mitkä on sitten niitten ristileikkaukset siitä, mitä meiltä puuttuu ja mitä meiltä ei puutu ja näin päin pois. Sitä komunikaatiota mä olen yrittänyt tässä parantaa.

T: Aikatauluista muutenkin niin projektinhan on myöhässä aikataulusta. Et siinä mielessä joo, että olen joutunu käymään asiakkaan kanssa keskustelua siitä, millon saadaan asioita

käyttöön, mutta näistä ei kaikki ole kyllä meidän hanskoissa ollu. Mutta toisaalta jos Tonni-kehittäjän kaikki tekeminen ois tällä hetkellä valmista, niin ei mekään oltais kyllä valmiita. Realiteetti vaan on se.

Kysymys 5

P: Tiedätkö esimerkiksi, mitkä osat integraatiosta, kun siinä on eri tietueita, että mitkä osat niistä menee sisään täysin tai 75%sti tai ei ollenkaan tai niin edespäin? Onko siitä mitään tietoa?

T: Sanotaanko että, suunnilleen tiedän tilanteen, mutta en tiedä tarkalla tasolla et mitä on, mitä puuttuu. Ja oikeastaan huomisessa tapaamisessa (12.1 devday asiakkaalla) tarkotus sitä vähän purkaa. Kaipaisin vielä jotain semmosta määrämuotoisempaa esitystapaa myös tähän koko projektiin liittyen. Josta kattoo et nää on tän kuukauden tehtävät, näin paljon niihin on allokoitu aikaa, ja sit nähtäis ina kuukauden päätteeks et näin paljon niihin on menny aikaa. Sitä itse asiassa asiakaskin kaipasi. Tän tyyppistä lähestymistapaa otettais niinku vahvemmin. Atte sitä on tehny keräilypuolelle kohtalaisesti, mutta ei sekään ole vielä sillä tasolla mitä mä kaipaisin. Et olis ihan vaan yks paikka mistä vois kattoo suoraan et okei, nää jutut tehdään tammikuussa, nää jutut helmikuussa. Ja mitä otetaan käyttöön helmikuun alussa on tällä hetkellä vielä aika auki.

P: Ja siihenkään ei ole kuin puolitoista kuukautta.

T: Nimenomaan, nimenomaan näin.

T: Yhteenvetona tiedän sen karkean tason tilanteen, mutta en tiedä ihan kaikkia yksityiskohtia ja pienenä riskinä tässä on vielä se, että sun ja Aten tekemiset on tarpeeksi tiiviisti katottu läpi. Jatkossa nimenomaan sen tuotannonohjauksen näkökulmasta, että onhan kaikki se tieto mitä tarvitaan. Toki sitä voidaan kehittää eteenpäin, mutta että oltais mahdollisimman pitkällä sen kanssa.

Liite 3: Keräilytoiminnallisuuden kehittäjä Aten haastattelu

Kysymys 1

P: Mikä on mielestäsi integraatioprojektin tarkoitus?

A: Tarkoitus on pyrkiä mahdollisimman porrastettuun siirtymään nykyisin käytössä olevan tietojärjestelmän ja Odoon välillä eli perimmäinen tarkoitus olisi että asiakkaan ei tarvitse kertaryminällä alkaa käyttää uutta järjestelmää ja meidän ei tarvitse yrittää kertaryminällä saada kaikkea toiminnallisuutta toimimaan mitä he tarvitsevat, vaan me voidaan osittain, palasissa, alkaa siirtämään heidän toiminnallisuuttaan uuteen järjestelmään ja tämä vaatii sitä että uusi ja vanha järjestelmä koko ajan komunikoi toistensa kanssa jota sitten integraatiojärjestelmä tekee.

Kysymys 2

P: Kuinka integraatioprojekti liittyy omiin tehtäviisi?

A: Konkreettisessa mielessa se liittyy sellasena ulkoisena osatekijänä eli minä toteutan sitä ainoaa käyttäjärajapintaa mitä tällä hetkellä tehdään, ja mulla täytyy olla saatavilla joitakin tietoja joita käyttäjät tarvitsee, enkä välttämättä voi hirveän paljon edetä ennen kuin minulla on ne tiedot käsissä ja niitä sen integraation pitää tuoda käyttöön.

Kysymys 3

P: Oletko pysynyt kärryillä, mitä on tähän mennessä tehty, mitä tulee tapahtumaan, mitä on nyt valmiina?

A: No jos, ei niinku, no pointing fingers, mut tämä on ehkä firmaa laajemminkin vaivaava ilmiö, että just jouduin kysymään sinulta, etää mitä tapahtuu, eteneekö tämä? Että siellä on muutama sellanen kenttä jotka on nyt ne viimeiset puuttuvat osaset mun käyttöliittymästä ja mulla ei ole mitenkään takaraivossa tietoa siitä miten se etenee, ja mulla ei ole aavistustakaan minne mä edes menisin tarkistamaan missä vaiheessa se menee, korkeintaan joku slackin historia ois joku sellanen paikka. Mutta mulla olis lähes täysin sama ongelma missä tahansa mussakin projektissa. Vaikka jokainen kenttä olisi eriteltynä omaan odoo-täskiinsä, ja niitä olisi siirrelty shipattu/cancelled/in progress-tiloihin sitä mukaa kun niitä tehdään, niin en mä olis silti niitä jaksanu hirveän pitkään etsiä sieltä Odoon kankean haku-toiminnallisuuden kautta.

P: Eli tieto siitä, missä mikäkin kohta menee on käytännössä hihasta nykäisy?

A: Joo, mielelään se ei sitä olis, mutta...

Kysymys 4

P: Onko integraation etenemisen nopeus tai hitaus haitannut omaa tekemistäsi? Tohon on tietysti jo osittain vastattu, mutta...

A: Joo, kyllä. Se nopeus en nyt niinkään, kun me pyritään saamaan deploymentit tehtyä samanaikaisesti, eli ei se että jollakulla on jo valmista niin ei se mitään haittaa, se on ollut, ei nyt pullonkaula välttämättä, mutta varsinkin tällasissä vähän monimutkasemmissa kokonaisuuksissa: oon mä jo muutaman sellasen simppelimmän tekstikentän voinu laittaa sinne javascriptissä voin sanoa et jos sitä kenttää ei ole olemassa niin whatever, pistä siihen tyhjää tekstiä, mutta sitten tollasissa monimutkasemmissa jutuissa kuten esim toimittajien tuotekoodeissa ja asiakkaiden tuotekoodeissa niin en minä pysty tekemään mitään sellaista kikkailua vaan mun täytyy oikeasti tietää miltä se arkkitehtuurisesti näyttää siellä ennenkuin minä voin toteuttaa mitään toiminnallisuutta. Se on ilman muuta tullut eteen, mä oon aika pitkälle pystyny sanomaan että tätä mä en voi tehdä vielä, teenpä jotain muuta. Mutta nyt mä just kattelin että mulla on neljä korkeamman rioriteetin täskiä jäljellä jotka on kaikki blokattuja.

P: Niin, koska kaikilla on joku requirementti ja niin edespäin, ja yhdellä requirementillä on sitten taasin kaksi muuta requirementtiä ja niin edespäin.

Kysymys 5

P: Oletko tyytyväinen integraation tämänhetkiseen toimivuuteen?

A: Mitä siinä nyt noita HTTP-kutsuja tehdään aikalailla, mutta kyllä mä ymmärrän että se suurin syy siihen hitauteen on ollut toisessa päässä, eikä meidän päässä niin ...

P: Ulkoinen tekijä

A: Niin, ulkoinen tekijä, aivan. Kyllä minä olen ihan tyytyväinen siihen miten se on edennyt, ja ymmärrän kyllä jos sieltä jotain sellaisia kummallisuuksia tulee, niin ymmärrän senkin, että sä jonglööraat kahden uuden järjestelmän kanssa: zato ja Odoo. Pythoniakaan sä et ollu ymmärtääkseni käyttäny hirveesti.

P: En kovin paljoa, no, Civilization IV:sta modannut joskus.

A: Ja sit sulla on Zaton lisäksi Odoon tietomalli siinä vieressä joka on vähän koukeroinen joissakin tilanteissa. KAikki sellaset, siellä on lapsosvirheitä, mutta ne on ymmärrettäviä sellaisia, et ei siinä ole mitään mun mielestä moitittavaa.

P: Se, että ne osat jotka sieltä läpi, ne jotka on valmiita, ne tulee läpi joko ihan semi-OK tai OK.

A: Ehkä se isoin huolenaihe on se että, se ei ole ollut end-to-end -pstyssä missään vaiheessa vielä. Sä oot testannu sitä sun Zaton scripti-kokonaisuutta, että se osaa käsitellä sitä tietoa, sä oot tarkistanu et se osaa puskea sitä tietoa Odooseen, sä oot inkrementaalisesti saanut tehtyä Tonnista tulevasta tiedosta sellasta, että se menee sieltä läpi. Mutta missään vaiheessa vielä edes testipalvelimella, kukaan ei ole täällä pystynyt painamaan missään lomakkeessa nappia, ja sanoa "vie tämä". Että se on siinä se huolestuttavin osa.

P: Joo, siitä ei taida olla muuta kuin tämä yksi valtava toimintakuvaus siitä, miten se kama liikkuu siellä Zaton sisällä, mutta esimerkikse se, kuinka lähtetä jotain Zatolle, niin semmonen asia puuttuu.

A: Aivan, sekin on vielä ihan auki, miten Tonnin rajapinnasta tulee mitään ulos.

Liite 4: Sprint 1 Planning 2016/01

Kysymyksiä käyttäjille

Varastomiehille:

Missä vaiheessa päätetään kuka on Käsittelijä? Kuka kirjoittaa tiedon Tonniin (vai kirjoittaako kukaan!? Pistääkö Tonni automaattisesti tulostavan käyttäjän nimen?) Pitääkö pystyä tabletilla määrittelemään kuka on Käsittelijä (riippumatta siitä lukeeko Tonnissa tieto?) Käsittelijä on myyjä joka käsittelee tilauksen: useimmiten sama kuin myyjä (asiakkaan oletusmyyjä) mutta ei aina. Ei liity varastoon.

Asiakkaan edustajalle:

Varasto lisäilee ylimääräisiä tuotteita: ennen tehtiin kynällä, nyt tehdään tabletilla. Odoo vaatii että lisättävä rivi sisältää oikean tuotteen. Onko kaikki tällaiset tuotteet (eurolavat, paketit) Tonnissa tuotteina?

On

Mitä kaikkea rahtikirjoihin tulee? Esimerkki-rahtikirjassa josta meillä on kuva on paljon käsin täytettyä tietoa: kuka kirjoittaa kyseiset tiedot? Voiko tiedot kirjata Tonniin etukäteen (esim. tuotteen paino) vai pitäisikö varastomiesten pystyä kirjaamaan tiedot tulostuksen yhteydessä?

Erityisohjeet

Kollien lukumäärä

Paino

Rahdin maksaja saattaa muuttua, mutta se annetaan vain myyjälle muokkausoikeus

Asiakaskohtaisissa sopimuksissa on määritetty kuka maksaa rahdin:

Vap. varastolla - asiakas maksaa

Vap. asiakkaalla - Firma maksaa

Saattaa olla myös kolmas osapuoli (TODO: Selvitä Tonni-kehittäjälta miten tiedot ovat Tonnissa, rahtisopimus Tonnissa ainakin olemassa)

done

Lähetteelle pitäisi saada tiedot oikein, toimitusosoite saattaa esim. muuttua

Priorisointi: Lähete tarvitaan nyt, rahtikirja ja kollitarra myöhemmin

Uusi vaatimus: Lähetteiden etsiminen ja tietojen katselu Odoosta

Lahdessa löydettiin seuraavien koodien tuotteita:

ME21551500

ME21551600

Tonnissa kyseiset koodit on merkitty poistetuiksi, joten niitä ei ole viety integraation kautta Odooseen. Voimme tehdä asialle kaksi asiaa (jopa molemmat):

Jos Atte saa tiedot tuotteista, hän voi lisätä ne manuaalisesti järjestelmään TODO: Asiakkaan edustajalle tuotekoodit ja mitä tietoja kaivataan
done
Integraatio voidaan pistää tuomaan myös poistettuja tuotteita Odooseen. Kysymys on onko se työn arvoista: kuinka paljon tällaista saattaa tapahtua? Jos tämän lisäksi sattuu kolme tai viisi kertaa niin Atte voi hyvin napsutella sisään manuaalisesti. Mutta jos tapahtuu useammin niin pitänee tuoda nekin integraation kautta.
Kirjataanko toimittajan sopimusnumero erilliseen kenttään, vai kirjoitetaanko se vain toimittajan nimen viereen lähetteellä?
Lähetystapa:
Sopimusnumero on asiakastiedoissa, 3 erillistä kenttää, josta ne saadaan rahtikirjalle
TODO: Ei tule integraatiosta, pyydetään Tonni-kehittäjältä
Pyydetty

Kaikille:
Miten hoidetaan lähetteen tulostus? Ymmärtääkseni varastomiehet printtaavat sen kun ovat keräilleet valmiiksi.
Tuleeko siihen kaikki samat tiedot mitä on nykyisessäkin?
Toive - lähetteelle lisää: osoitekentät eivät aina mahdu Tonni-järjestelmään (ongelma Tonnin päässä, ei vaadi toimenpiteitä)
Tuleeko rivit myyntitilauksesta? Eli onko lähetteellä mukana postikulut yms? (huom. että Odoossa myynnin rivit != keräilyn rivit ja palvelut eivät siirry keräilyyn)
Toive - tehty hintataulukko paperille ja sitä haluttaisiin myös automatisoida

Lähetteen sivuun merkitään nykyään paperille painona (kg) ja sen perusteella lasketaan postikulut ja lisätään PK-rivinä Tonniin, postikulujen ei tarvitse näkyä lähetteellä => Lisätään backlogiin, ei toteuteta vielä
Missä vaiheessa uudessa järjestelmässä myyjää kirjaa riville lisätietoja (“osa näistä tuotteista jostain muualta” yms)? Ennen vanhaan kirjoitettiin käsin printattuun lähetteeseen.
Vaihtoehtoja:
Jos kenttä on jo olemassa ja muokattavissa, Myyjien pitäisi käyttää sitä: ei enää lyijykyniä
Myyjä menee varastojärjestelmään, etsii kyseisen keräilyn ja täyttää tiedot sinne
Hyvää:
Ei tarvitse tehdä uutta logiikka
Huonoa:
Myyjille pitää sallia pääsy varaston puolelle
Myyjä menee Odooseen ja kirjaa lisätiedot myynnin riveille, joista ne sitten uivat keräilyyn
Asennetaan moduuli tähän soveltuvaa moduulia ei ole

Hyvää:
Myyjällä ainoastaan oikeudet myyntiin
Huonoa:
Kyseinen kenttä pitää jotenkin saada keräilyyn, joka ei välttämättä ole triviaalia (onko siirrosta suora linkki myyntiriviin?)
Tähänkin löytyy moduuli:
https://apps.openerp.com/apps/modules/8.0/stock_sale_order_line/
Kaikki myynnin rivit menevät automaattisesti readonly-tilaan kun myynti hyväksytään, eikä niitä voi Odoo:n perustoiminnallisuudessa muokata
Kommentit menee samaan kenttään myynti ja varasto-puolella
Pakotetaan lisätieto-kenttä read/write-tilaan vaikka myynti olisi hyväksytty
Tonniin lisätään kenttä jota kautta käyttäjä lisää lisätiedon (Atte äänestää tätä)
Hyvää:
Käyttäjälle loogisinta: kaikki myyntiin liittyvä tehdään Tonnissa, ei tarvitse siirtyä järjestelmästä toiseen
Huonoa:
Pitää lisätä Tonniin kyseinen kenttä
Pitää lisätä integraatioon kyseinen kenttä

Integraatio

Tuote:
negatiivinen varastosaldomäärä estää tallennuksen
muunnetaan negatiiviset saldomäärät -> 0 ? (käsittely https://repo.web-veistamo.fi/odoo/client-terastarvike/blob/master/addons/integration_tonni/models/product.py#L30 ja jos nollaksi muuttaminen ei sovi, niin koitetaan tehdä positiivinen inventaario tonni stock -> tonni inventory)
Miinusmerkkinen saldo tulee silloin, kun laskutetaan ennen kuin on vastaanotettu määrä päivitetty Tonniin
Päätös: saa mennä nollana done

manufacture_ok -kenttä ei ole olemassa (tarvitaanko?) odoo conf
TODO:
tonnin myyks suoraan odooseen, muunnokset pois done
yn_ryhma -> ext_product_group
luodaan kenttä odoo conf

käykö categ_id (product.category)?

Asiakas/Toimittaja:

TODO:

yt_lvvtun (esim 1234567-0) -> vat done

yla_toimtapa (esim VIEDÄÄN) -> propert_delivery_carrier

Rakenne:

rivi jossa vaadittu tuotemäärä = 0 estää tallennuksen

tiputetaan nämä rivit? kysymys asiakkaalle

Aineistomoka, WIP

asennetaan odoo moduuli joka hyväksyy 0-rivit? odoo conf

Myyntitilaus:

T-rivin tekstikenttä/kentät puuttuu ei xml:ssä

TK & PK-rivi

lisätään automaattisesti uusi rivi jossa 1kpl tiettyä tuotetta?

tarvitaanko mitään muuta?

TODO:

mt_tila -> state ei xml:ssä

mt_totatks (esim VIEDÄÄN) -> carrier_id (delivery.carrier)

mtr_posno -> sequence

https://apps.openerp.com/apps/modules/8.0/stock_sale_order_line/

myyntirivi <-> siirtorivi

Toimitustavat (delivery.carrier):

luodaan nämä valmiiksi? -> manuaalisesti, osa poistetaan

Toimitustapa: name (esim NOUTO)

Normaalihinta: normal_price

mistä tulee? aina 0?

Kuljetusyritys: partner_id

aina Firma? (entä esim jos jos tapa = KIITOLINJA ?)

luodaan valmiiksi?

Toimitustuote: product_id

aina sama?

luodaan valmiiksi?

TODO Pekko: lähetä lista Asiakkaan edustajalle ja varmista mitkä ovat tarpeellisia done

Automatisoidut toiminnot

Mukaan production conffiin:

Autoconfirm new sale.orders created by zato odoo conf

Tekemistä integraatioon

Integraation testaus ja varmistus, kun materiaali saatu Tonni-kehittäjälta

Integraatiopalvelimen tuotantokelpoisuuden varmistus

Tuoteintegraation testaus

Xml-mallien viimeistely

Odoo-kenttien varmistus

Lähetteen tiedot varmistettu? että kaikki kentät tulevat integraation mukana

ext_code=PAR in product.uom not found in Odoo

Keräily ja varastonhallinta

Tekemistä

Yksikkömuutos

Muunnetaan yksiköt takaisin WIP

Keräilyn visualisointi: SATit erotettaisiin muista yksiköistä (vihreäksi tai siniseksi SATin määrä, jolloin

50% ylikeräily heittää huomion

Käytettävyystestaus

UI viimeistely

Taustalogiikka kuntoon

Tulostustestaus varastossa

Raportit

Lähetteen raporttipohjan muutokset?

Raportit (helmikuulle?)

Rahtikirja

Kollitarrat

Tuotannonohjaus

Tuotannonohjauksen integraatio:

Miten hoidetaan valmistuksessa kulutetut tuotteet (vähennetäänkö saldoja ja pitääkö sen vaikuttaa myös Tonnin tuotesaldoihin integraation kautta - entä poikkeustilanteet kuten tuotepuutteet tällöin?) ja mihin valmiit tuotteet siirretään (Tonni-integraatiolla Tonni-saldoihin vai omaan tuotanto-varastoon vai jotenkin muuten?)

Työmääräarvio integraatiosta:

Valmistuneiden ja tuotettujen tuotteiden määrämuutosten siirtäminen Odoosta Tonniin

Tuotantohinnan laskeminen: materiaalit ja työmäärät

TODO: tarkastetaan kumpaa hintaa halutaan käyttää viimhhintaa vai pkeskhin Odoon kustannuslaskennassa (Asiakkaan edustaja tarkistaa Ristolta)

Tekemistä

Vaatimuslistan läpikäynti -> mitä voitaisiin vielä pudottaa pois?

Suunnittelutyökalut

Tuotannonohjaus ilman BoMia

Muuta

Aikataulu

Tammikuun käyttöönottopäivä?

Helmikuun suunnittelutapaaminen?

Sis: Seuraava dev day?

Koulutus salasanojen muuttamiseen

Dippatyön esittelytilaisuus - päivä?

Myynti

NFC-koulutus - päivä?

Liite 5: Sprint 2 Planning 2016/02

Kysymyksiä käyttäjille

Asiakkaan edustaja

Jos tuotetta ei ole tilattu alunperin, mikä sen tilattu määrä pitäisi olla lähetteellä (0 oletettavasti?)

Kun keräily tulee käyttöön, myyjien pitäisi kirjata mahdolliset rivien lisähuomiot Odoossa. Näin ollen heillä pitäisi olla kaikilla tunnukset kunnossa Odooseen.

Pyytäisimme listaa jossa näkyy seuraavat tiedot jokaiselle käyttäjälle:

Toivottu käyttäjänimi (täysin vapaasti valittavissa)

Sähköpostiosoite

Pyydetty spostitse

Myyjät

Jos myyjä merkkaa lisätietoja Odoo:ssa, onko riski sille että varasto ottaa keräilyn tehtäväksi heti? Silloin myyjä saattaa lisäillä riveille lisätietoja samanaikaisesti keräilyn kanssa, jolloin varasto ei näe lisätietoja. Jos tämä saattaa olla ongelma, mitä asialle voisi tehdä? Jos tämä on riski ainoastaan kiireellisissä tapauksissa, myyjä voisi olla suoraan varastoon yhteydessä ja varoittaa etteivät aloita keräilyä ennen kuin on kirjannut lisätiedot

Tonniin voitaisiin lisätä toiminto/kenttä jonka perusteella Zato näkee että luotua myyntitilausta ei pitäisi hyväksyä integraation jälkeen. Myyjä menee sitten Odooseen, merkkaa lisätiedot ja merkitsee myynnin sitten hyväksytyksi

Tonniin lisätään myyntiriville lisätieto-kenttä johon myyjä voi merkitä lisätietoja, jotka sitten menevät Odooseen integraation kautta (vaatii paljon duunia Tonnin puolella, tuskin tehdään).

Integraatio

Salaus:

Tonni-kehittäjä: pistetään tietoa Tonnista Windows-työasemalle ja hoidetaan tiedonsiirto siitä/siihen

Ulos: curl (SSL)

https://curl.haxx.se/download.html

http://stackoverflow.com/questions/9507353/how-do-i-install-set-up-and-use-curl-on-a-windows

Sisään: SFTP? FTPS?

https://filezilla-project.org/download.php?type=server

https://wiki.filezilla-project.org/FTPS_using_Explicit_SSL/TLS_howto_(Server)

Voiko myyks vaihtua? (eihän, pliis)

Tuotekohtaiset asiakaskoodit

Muutamalla tuotteella 1 tai useampi asiakaskoodi, jossa nimike1-kenttä on tyhjä. Mitä tehdä?

Valtaosalla MTR:stä toimitettu & laskutettu on 0.

Esitellään xml-data Tonni-kehittäjälle, selvitetään homma tarkasti

Onko niin, että to- & lamaara näkyy vain jatkotoimituksissa?

SYY: ei oltu vielä keräilty

MTR T-riveistä puuttuu tekstikenttä/kentät

mtr_nimitys1 XML:ään

Tonnissa voi kirjata tyhjiä rivejä. Zaton täytyy jotenkin käsitellä tällainen:

Tehdään uusi rivi-tyyppi Tonniin, merkataan Odoo:n erikoistuote eri lailla jos se on alitilausrivi (pitää korostaa)

MTR TK & PK-rivit, miten tehdään

Myyntitilaukselle uusi kenttä mt_kerpaik -> sale.order.warehouse_id

Varastolle (stock.warehouse) uusi kenttä ext_identifier josta Zato tunnistaa varaston kirjaimeen perustuen

Timeline

Pe 26.02 Kaikki asiakaskoodit sisään, ensin testiin sitten tuotantoon

Pe 04.03:

Kaikki Tonni-kehittäjän hommat tehty

Myyntitilausdumppi 01.01.2014-nyt

Mielellään 6 Mt kerralla (200 000 riviä kerrallaan)

Yhteyden salaus: joko HTTPS tai (S)FTP(S) (jos HTTPS, niin Zaton toiminta muuttuu: ei enää pollata FTP:llä.)

Tonni-kehittäjä tutkii FTPS ja SFTP (eli SSH-yhteyden) mahdollisuudet

Miku lähetti Tonni-kehittäjälle SCO Unixin FTP:n urlin, josta löytyy OpenSSH ja GNU Tools (SFTP- ja FTPS-tuki mahdollisesti)

FTP kuntoon:

Muutetaan tiedostonimi aikaleimaksi (unix timestamp + muutama generoitu kirjain)

Myyntitilaus riveineen kirjoitetaan WV-OUT-kansioon kun myyjä on valmis

MTO xml -pohjaa käytetään
Keräilyn paikan valitseminen
mt_kerpaik: M (tapanila), T, L, J
Nykyään printtaus: PR-komento, P1, P2, P3
Uusi toiminnallisuus: uusi PR-komento (Print-komento) eli TM ja TL
Tila näkyviin Tonniin: “Lähetetty tablet-järjestelmään”
Jos lähetystä yritetään tehdä uudelleen niin kysytään käyttäjältä: “haluatko varmasti lähettää uudelleen?”
Toimintatapa: muutoksia myyjä pystyy tekemään suoraan Odoon toimitustilaukselle
Integraation kirjoittamat keräily-xml:t luetaan Tonniin WV-IN -kansiosta
Kun keräily on hyväksytty (ei enää muutoksia), niin silloin tiedosto tulee Tonnin päähän
Uudet tuotteet: pos=0 ja uusia tuotteita voi tulla useita
Tilakoodi myynnille: “Tablet-keräily on tehty”
Samalla myyjälle lähetetään sähköpostia keräilyn onnistumisesta, myyjä printtaa lähetteen, kirjaa siihen huomiot ja jälkitoimitukseen/ei-jälkitoimitukseen ja toimitetaan paperisena laskutukseen
Alitilausrivin korostus
Tehdään TA-rivi, joka korostetaan Odoossa lähetteellä
TA-rivi tulostuu kuten T-rivi Tonnista nykyään
Jälkitoimitus: merkitään tilaan “toimitettavaa” ja putki käynnistyy alusta uudelleen
Keräilijä ei tiedä jääkö jälkitoimitukseen vai ei vaan myyjä tekee päätöksen
Kaikki Pekon hommat tehty
Kaikki Aten hommat tehty
Ma 07.03. Kenraaliharjoitus niin tuotantomaisena kuin mahdollista. Simuloidaan keräilyn taival myyjältä varastomiehelle ja takaisin.

Ke 09.03. Koulutukset paikan päällä: Myynti ja varasto

Pe 12.03. 16:00 alkaen huoltokatko ja käyttöönotto:
Päivitetään tuotannon koodi:
config_client: 2.0.0 -> 2.1.0
rousku: 1.1.3 -> 2.0.0
integration_tonni: 1.1.0 -> 1.2.0
Tonni-kehittäjältä uusi myyntidumppi, 01.01.2014 - nykyhetki.
Importataan myyntidumppi. Myyntihistoriasta ei luoda keräilyjä, vaan ne ilmestyvät vasta uusista myynneistä.

Ma 15.03. Atte, Tuomo, Pekko paikan päällä tukena, varmistetaan toimivuus

Liite 6: Sprint 1 Retro 2016-02-11

Hyvää

Tuotantoon mentiin ajallaan ja silloin shipattiin
Ei tullut käyttöönotossa yllätyksiä
Hyvä draivi
Roolijako: integraatio, keräily & varastonhallinta, tuotannonohjaus
Devdayt & paikallaolo Ttarvikkeella
Kuukausisprintit & aikataulujen lyöminen lukkoon
Määrämuotoinen käyttöönottosuunnitelma
Myös plan B, jos kaikki ei mee niin kuin sovittu
Ennalta koottu kyssärilista Asiakkaan edustajalle

Huonoa

Ongelmia kenttien kartoituksen kanssa
Saadaanko kenttä? Minne menossa? Mistä tulee?
Mappauskaavio on hyvä, mutta vanhentunut ja ei palvellut
Atte sai kuvan, että Pekko tajusi että sai jotain tietoa ja alkoi etsimään mistä se löytyy
Odoosta - olisi voinut heittää tehtävän Atte/Miku hommaksi “tee mulle kenttä”
=> Saa tehdä muille tehtäviä, jos tarvitsee jotain

Zato-docker
Miku sai sen itse tehtynä johonkin pisteeseen ja tyytyi omaan Zaton omaan imageen

Tonni-kehittäjän paimentaminen
Tonni-kehittäjä venkoili vastaan, unohti asioita ja hukkasi muistiinpanoja
Ei voi luottaa siihen, että Tonni-kehittäjä hoitaisi lupaamansa hommat
Curlia ei edes selvitelty ja ratkaisuna Windows-välikone
Viivästykset pitkälti Tonni-kehittäjän syytä
=> Jatkossa alkuviikko ja loppuviikko peräänkyselyä

Liian isot tehtävät -> Pilkotaan pienempiin
Työmääräarviointi
Sprint-raportointi edelleen hankalaa
Asiakkaan edustaja ja sähköposti

Käyttöönottojen lykkäytyminen
Ideoita
Selkeämmät roolijaot
Esim. mappaustaulukon vastuunjako

Jos tehtäisiin laskuintegraatio niin Tonni-kehittäjälta voisi pyytää printin laskusta sekä kaikki kentät, jotka liittyvät laskuihin
Aloituspaketti, jossa tietorakenne

Jos mahdollista niin esim. myynti olisi ollut helpompi ottaa Odoosta käyttöön ja vähentänyt tarvetta rajapinnoille
Esim. laskujen luonnissa voitaisiin ottaa koko laskutusrutiini käyttöön, pelkkien laskujen siirron sijaan

Kevätsiivous tai sprint-siivous Drive-kansiolle
Osa hackpadiin?

Tehtävien tägitys (lisätty idea-kanavalle)
Tagi on hyödytön, jos muut eivät tiedä sen olemassa olosta

Käyttäjätarinat Odooseen kaikilta projekteilta

Lisää flowchartteja ja havainnollistavia kaavioita

Sprint-raportointiin helpompi työkalu

Kysymyksiä

Miten saadaan jatkossa pysymään porukka kartalla missä mennään?
Ei tulisi uusina asioita juttuja vaan kaikki tietäisivät suunnilleen tuoreimmat kuulumiset
Tuntien käyttö on yksi kuvaava
Tehtävien tilanne on toinen

Miten Asiakkaan edustajalle tulevaa kysymystulvaa saataisiin pilkottua/pienennettyä?

Liite 7: Tuntikirjaukset

Vaihe	Pvm	Kuun tunnit	Täski	Täskin tunnit	Työkuvaus	Tunnit
Vaihe 1: Opette-lu	2015-02-16	08:15	/	00:45		00:45
	2015-02-26	08:15	Kenttien vastineiden kartoittaminen	02:00	XML-schemojen luonti	02:00
	2015-02-26	08:15	Synkronointikäytännön määrittely	02:00	Integraatiokokonaisuuden prosessikaavio	01:00
	2015-02-26	08:15	Synkronointikäytännön määrittely	02:00	Integraatiopalvelimen tarkempi prosessikaavio	01:00
	2015-02-26	08:15	Tonnin XML-muotoisten viestien määritys	07:15	Miitti asiakkaalla + matka-aika 40min	03:30
	2015-03-10	09:15	Zato-palvelimen opiskelu	17:00	Zato ensiasennus ja testailu	04:30
	2015-03-11	09:15	Tonnin XML-muotoisten viestien määritys	07:15	määrittelyä	01:45
	2015-03-17	09:15	Zato-palvelimen opiskelu	17:00	odoo-integraatio opiskelu	01:00
	2015-03-18	09:15	Tonnin XML-muotoisten viestien määritys	07:15	xsd & xml-pohjat	02:00
	2015-04-22	08:45	Zato-palvelimen opiskelu	17:00	zato käyntiin jälleen, dokumentointi	00:45
Vaihe 2: Ohjel-mointi	2015-04-23	08:45	Zato-palvelimen opiskelu	17:00	testidatan ensitestit, dokumentointi	02:00
	2015-04-24	08:45	Zato-palvelimen opiskelu	17:00	nuuskii sisääntulevan XML:n, hakee odoon taulujen nimet joka riville (joille se onnistuu 1:1)	01:30
	2015-04-28	08:45	Zato-palvelimen opiskelu	17:00	kuinka saada hajonnut zato toimimaan, lisää xml-to-odoo logiikkaa	01:30
	2015-04-29	08:45	CO: Tonni -> Integ / asiakkaat	09:00	Zaton luku ja muunto-logiikkaa	01:00
	2015-04-29	08:45	CO: Tonni -> Integ / myyntitilaukset	28:30	Zaton luku ja muunto-logiikkaa	01:00
	2015-04-29	08:45	CO: Tonni -> Integ / tuotteet	32:45	Zaton luku ja muunto-logiikkaa	01:00
	2015-05-04	39:00	CO: Tonni -> Integ / asiakkaat	09:00	Asiakas & tuote-xml prosessointi	02:45
	2015-05-07	39:00	Integ -> Odoo / Opiskelu ja testaus	05:00	Yritetty dockerin kanssa	02:00
	2015-05-07	39:00	Integ -> Odoo / Opiskelu ja testaus	05:00	Yhteyden muodostus & troubleshooting	01:00

	2015-05-07	39:00	Integ -> Odoo / Opiskelu ja testaus	05:00	trial & error, yhteys ja komennot toimivat	01:30
	2015-05-07	39:00	Integ -> Odoo / Opiskelu ja testaus	05:00	opiskelua	00:30
	2015-05-08	39:00	Integraatioprosessin ja datatyyppien määrittely	14:30	Tonni -> Zato -> Odoo prosessi	08:00
	2015-05-11	39:00	Zato-palvelimen opiskelu	17:00	zato kaatunut, load balancer ei käynnisty	02:00
	2015-05-18	39:00	CO: Tonni -> Integ / myyntitilaukset	28:30	XML määrittelyn muutos	00:30
	2015-05-18	39:00	CO: Tonni -> Integ / myyntitilaukset	28:30	XML:t sisään ja testaus	00:30
	2015-05-18	39:00	Zato-palvelimen opiskelu	17:00	load balancer takas pystyyn, git pystyyn, opetus ja funtsailu	03:45
	2015-05-19	39:00	CO: Tonni -> Integ / asiakkaat	09:00	Tonni & TT-Tonni eriyttäminen	01:30
	2015-05-19	39:00	CO: Tonni -> Integ / myyntitilaukset	28:30	Tonni & TT-Tonni eriyttäminen	01:30
	2015-05-19	39:00	CO: Tonni -> Integ / tuotteet	32:45	Tonni & TT-Tonni eriyttäminen	01:30
	2015-05-19	39:00	Integraatioprosessin ja datatyyppien määrittely	14:30	TT-Tonni & Tonni eriyttäminen	01:00
	2015-05-20	39:00	Integ -> Odoo moduuli	27:00	Alkuversion koodausta	02:00
	2015-05-25	39:00	Integ -> Odoo moduuli	27:00	zaton haproxy korjaus	01:15
	2015-05-25	39:00	Integ -> Odoo moduuli	27:00	odoo pyörimään dev2:lle	00:45
	2015-05-25	39:00	Integ -> Odoo moduuli	27:00	koodailu	02:30
	2015-05-28	39:00	Zato-alustan vaatimusten täyttäminen	02:30	tutkittu ja kirjailtu	01:00
	2015-05-29	39:00	Integ -> Odoo moduuli	27:00	koodailua	02:00
	2015-05-29	39:00	Zato-alustan vaatimusten täyttäminen	02:30	lisätutkintoja ja kirjauksia	01:30
	2015-06-01	31:15	Integ -> Odoo moduuli	27:00	kirjoitus onnistuu!	01:30
	2015-06-02	31:15	Integ -> Odoo moduuli	27:00	toiminnan funtsimista	04:00
	2015-06-02	31:15	Integ -> Odoo moduuli	27:00	asiakaskirjoitus ok	04:15
	2015-06-03	31:15	Integ -> Odoo moduuli	27:00	failed to send mail ihmettely ja korjaus	01:00
	2015-06-03	31:15	Integ -> Odoo moduuli	27:00	odoo-docker takas toimintakuntoon	01:30
	2015-06-04	31:15	Integ -> Odoo moduuli	27:00	asiakkaat & niiden yhteyshenkilöt, työstyä	03:30
	2015-06-04	31:15	Integraatioprosessin ja datatyyppien määrittely	14:30	asiakas XML -> asiakkaan & yht hlö:den res.partnerit	02:00
	2015-06-22	31:15	CO: Tonni -> Integ / asiakkaat	09:00	asiakkaat & osoitteet muunto-logiikkaa	02:00
	2015-06-22	31:15	Integ -> Odoo moduuli	27:00	asiakkaiden yhteyshlö:t asiakkaiden lapsiksi	01:15
	2015-06-22	31:15	Tapaamiset muiden kehittäjien kanssa	08:00	Aikataulut & missä mennään	01:30

	2015-06-23	31:15	CO: Tonni -> Integ / asiakkaat	09:00	osoitteet asiakkaiden lapsiksi	01:45
	2015-06-23	31:15	Tapaamiset muiden kehittäjien kanssa	08:00	Kenttä revyy Aten kanssa	00:45
	2015-06-24	31:15	CO: Tonni -> Integ / tuotteet	32:45	Toimittajat odooseen	01:30
	2015-06-24	31:15	CO: Tonni -> Integ / tuotteet	32:45	Tuotteet odooseen, supplier puuttuu	04:00
	2015-06-26	31:15	Odoo -> Integ -> Tonni sunnittelu	05:45	alkusuunnittelu	00:45
	2015-07-01	76:45	CO: Integ -> Tonni	07:15	runko koodattu	02:00
	2015-07-01	76:45	Odoo -> Integ	09:00	runko koodattu	01:00
	2015-07-01	76:45	Odoo -> Integ -> Tonni sunnittelu	05:45	Odoo -> Zato -> Tonni prosessikaavio	04:30
	2015-07-02	76:45	CO: Integ -> Tonni	07:15	läpikäynti tuomon kanssa & preppaus miittiä varten	01:30
	2015-07-06	76:45	CO: Tonni -> Integ / tuotteet	32:45	Zaton sisäinen invoker-service & virheiden läpikäynti	02:00
	2015-07-06	76:45	Tapaamiset muiden kehittäjien kanssa	08:00	ma with tuomo & Tonni-kehittäjä	03:45
	2015-07-07	76:45	Odoo -> Integ	09:00	mock-up testidata, prosessin koodailua	02:30
	2015-07-08	76:45	CO: Integ -> Tonni	07:15	testi ftp-servu pystyyn, prosessin koodailua	03:45
	2015-07-08	76:45	Odoo -> Integ	09:00	prosessin koodailua	03:30
	2015-07-09	76:45	CO: Tonni -> Integ / myyntitilaukset	28:30	tilaukset & tilausrivit sisään & linkittymään oikein	04:00
	2015-07-10	76:45	Aidon datan siirron seuranta	08:30	13000 ekaa tuotetta + korjauksia	01:30
	2015-07-10	76:45	CO: Tonni -> Integ / tuotteet	32:45	13000 ekan tuotteen tekemien virheiden korjaus	02:00
	2015-07-13	76:45	Aidon datan siirron seuranta	08:30	tuotteiden import jatkuu	02:30
	2015-07-13	76:45	CO: Tonni -> Integ / tuotteet	32:45	calc-servicen tuote-osan korjaukset	02:30
	2015-07-14	76:45	Aidon datan siirron seuranta	08:30	tuotteet.xml korjattu, raportoitu, kaikkien (paitsi 1) import ok	02:30
	2015-07-14	76:45	CO: Tonni -> Integ / tuotteet	32:45	calc-servicen tuote-osan korjaukset jatkui	02:30
	2015-07-22	76:45	CO: Tonni -> Integ / myyntitilaukset	28:30	tilausrivien hinnat, double-ihmettely	02:30
	2015-07-24	76:45	CO: Tonni -> Integ / asiakkaat & toimittajat	08:15	Kaikki asiakkaat, ongelmien korjaukset ja datan siirron seuranta	05:15
	2015-07-27	76:45	Aidon datan siirron seuranta	08:30	asiakas & tuotesiirtoja	02:00
	2015-07-27	76:45	Integraatioprosessin ja datatyyppien määrittely	14:30	asiakas, toimittaja, tuote: kentät tonni<>odoo	03:30
	2015-07-27	76:45	Rikkinäisen datan käsittely	04:45	aidon datan kanssa setvimistä, paljon parempi loggaus ja virhepalaute	03:00
	2015-07-28	76:45	CO: Tonni -> Integ / asiakkaat & toimittajat	08:15	asiakkaat & toimittajat uudelle testipalvelimelle	03:00
	2015-07-28	76:45	CO: Tonni -> Integ / tuotteet	32:45	tuotteet uudelle testipalvelimelle	04:00

	2015-07-28	76:45	Integ -> Odoo moduuli	27:00	virheilmoitukset & niiden läpikäynti paljon paremmaksi	01:30
	2015-07-29	76:45	CO: Tonni -> Integ / myyntitilaukset	28:30	kenttien selvitystä, asiakas+toimitus+laskutus-osoitteiden tarkastus ja händlaus	04:30
	2015-07-29	76:45	Tapaamiset muiden kehittäjien kanssa	08:00	ttarvike-client projektin käyttöönotto, kenttien läpikäyntiä	02:00
	2015-07-31	76:45	CO: Tonni -> Integ / myyntitilaukset	28:30	koodin siivousta, virheiden kitkemistä	03:30
Vaihe 3: Pitkit-tyminen	2015-08-03	11:15	Rikkinäisen datan käsittely	04:45	myyntitilauksien sisääntuonnin seuraamista & korjailuja	01:00
	2015-08-04	11:15	Rikkinäisen datan käsittely	04:45	tyhjien kenttien tarkastus, kaupunki+postikoodi tarkastus	00:45
	2015-08-10	11:15	CO: Tonni -> Integ / tuotteet	32:45	tilannekatsaus	00:30
	2015-08-24	11:15	CO: Tonni -> Integ / tuotteet	32:45	product type päivitykset, rikkinäisen datan käsittely	04:00
	2015-08-25	11:15	CO: Tonni -> Integ / tuotteet	32:45	devday: virhekäsittelyn uudistus, integraatioprosessin läpikäynti	04:30
	2015-08-28	11:15	CO: Tonni -> Integ / tuotteet	32:45	tuotteilla oikea kategoria (jatkuu)	00:30
	2015-09-11	00:30	CO: Tonni -> Integ / tuotteet	32:45	oudot tuotteet & asiakkaat keräys	00:30
	2015-10-19	20:00	Tuotantopalvelimeen siirtyminen	14:30	bugfixes	02:00
	2015-10-21	20:00	Tuotantopalvelimeen siirtyminen	14:30	zato-koodin korjailua & testausta	03:30
	2015-10-22	20:00	Tuotantopalvelimeen siirtyminen	14:30	tuotantopalvelimeen siirtyminen	07:15
	2015-10-23	20:00	Tuotantopalvelimeen siirtyminen	14:30	zaton käyttö-ohjeistus & gittailu	00:45
	2015-10-23	20:00	Tuotantopalvelimeen siirtyminen	14:30	tulosten tarkastaminen & bugien korjailu	01:00
	2015-10-26	20:00	Lokitus (Sentry)	00:00	Sentrytus ja testailu	02:00
	2015-10-27	20:00	CO: Integ -> Odoo / tuotteet	09:45	tuoterakenteet (BoM)	02:00
	2015-10-27	20:00	Lokitus (Sentry)	00:00	sentry-palvelun opettelu & lokituksen tarkastus	01:00
	2015-10-27	20:00	Odoo -> Integ -> Tonni sunnittelu	05:45	virheraportointi kuntoon	00:30
	2015-11-03	02:15	FIX Odoossa ei pitäisi käyttää KPL-yksikön monikkoja tuotteen oletusyksikkönä	01:15	Mistä on kyse-höpinä + muunnoksen koodaus & testaus	01:15
	2015-11-03	02:15	CO: Tonni -> Integ / tuotteet	32:45	sat/tuh konversiot yksiköiksi	01:00

Vaihe 4: Sprint 1	2015-12-01	29:45	SitRep miitti	01:15		01:15
	2015-12-04	29:45	Käyttöönotto	20:15	Vaihe 1: Testiskripti	01:30
	2015-12-04	29:45	Käyttöönotto	20:15	Vaihe 1: API-avain	03:00
	2015-12-11	29:45	Käyttöönotto	20:15	Vaihe 2: tuotexml malli	00:30
	2015-12-11	29:45	Käyttöönotto	20:15	Vaihe 2: rakenteen tonni -> odoo koodaus	01:45
	2015-12-11	29:45	Käyttöönotto	20:15	Vaihe 1: lisäselvitysmaili	00:30
	2015-12-16	29:45	Käyttöönotto	20:15	Vaihe 2: dokumentointi, rakenne valmis	01:00
	2015-12-18	29:45	-tapaaminen	02:45		02:45
	2015-12-21	29:45	CO: Integ -> Odoo / myyntitilaukset	14:30	mto & mtr	04:30
	2015-12-22	29:45	CO: Integ -> Odoo / myyntitilaukset	14:30	mto & mtr	01:30
	2015-12-23	29:45	CO: Integ -> Odoo / myyntitilaukset	14:30	mto & mtr	06:00
	2015-12-29	29:45	CO: Integ -> Odoo / myyntitilaukset	14:30	muiden tietojen tuonnin testaus & koodin korjaukset mto & mtr vaatimien muutosten takia	02:00
	2015-12-30	29:45	CO: Integ -> Odoo / tuotteet	09:45	uuden tuotexml: haku, tuoteinteraation ajo ja seuranta	01:00
	2015-12-30	29:45	CO: Integ -> Odoo / tuotteet	09:45	uusien kenttien lisäys integraatioon	00:30
	2015-12-30	29:45	CO: Integ -> Odoo / tuotteet	09:45	uuden tuote xml:n korjaus	00:30
	2015-12-31	29:45	CO: Integ -> Odoo / tuotteet	09:45	selvitys miksi 130 tuotetta eivät menneet läpi. syy löytyi	01:30
	2016-01-04	38:15	CO: Integ -> Odoo / tuotteet	09:45	uusi tuoterakennedata	01:45
	2016-01-04	38:15	Käyttöönotto	20:15	Vaihe 2: odooseen lisättyjen kenttien tuonti	02:30
	2016-01-05	38:15	Käyttöönotto	20:15	Vaihe 2: to & as -> terrpitestiin, devday asiakkaan tiloissa	04:30
	2016-01-12	38:15	Käyttöönotto	20:15	Vaihe 2: muutokset integraatioon keskustelujen pohjalta	02:00
	2016-01-12	38:15	Käyttöönotto	20:15	Vaihe 2: devday miitti	02:15
	2016-01-14	38:15	Käyttöönotto	20:15	Vaihe 2: tuotetuonnin tietojen tarkistus	00:15
	2016-01-18	38:15	Käyttöönotto	20:15	Vaihe 2: sitrep	00:30
	2016-01-18	38:15	Odoo -> Integ	09:00	prosessikehitystä	01:30
	2016-01-22	38:15	CO: Integ -> Odoo / tuotteet	09:45	uusi tuotexml	01:00
	2016-01-22	38:15	Odoo -> Integ	09:00	koodausta odoo-datan mukaan	00:30

	2016-01-25	38:15	Tuoteintegraation livetys	22:00	zato setup & ohjeet	01:45
	2016-01-26	38:15	Tuoteintegraation livetys	22:00	kenraaliharjoitus 1 valmistelu	04:45
	2016-01-27	38:15	Tuoteintegraation livetys	22:00	kenraaliharjoitus 1	01:30
	2016-01-28	38:15	Tuoteintegraation livetys	22:00	ftp service	04:30
	2016-01-28	38:15	Tuoteintegraation livetys	22:00	kenraaliharjoitus 2 valmistelu	02:00
	2016-01-29	38:15	Tuoteintegraation livetys	22:00	tuoteinteg livetys	02:30
	2016-01-29	38:15	Tuoteintegraation livetys	22:00	kenraaliharjoitus 2, livetyksen valmistelu	04:00
	2016-01-30	38:15	Tuoteintegraation livetys	22:00	tuoteinteg livetyksen onnistumisen tarkistus	00:30
Vaihe 4: Sprint 2	2016-02-02	31:30	CO: Tonni -> Integ / tuotteet	32:45	ftp tester service, yhteyden testaus	00:45
	2016-02-03	31:30	Helmikuu sprintin suunnittelupaltsu	01:15		01:15
	2016-02-05	31:30	Sprint 02/16 täskien luonti & tarkennuskeskuste-lut	01:30		01:30
	2016-02-08	31:30	Tonni FTP-listaus kuntoon	00:30	ftp test service	00:30
	2016-02-08	31:30	dokumentointi	01:00		01:00
	2016-02-08	31:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	testiajon suunnittelu	00:30
	2016-02-10	31:30	devday miitti	02:15		02:15
	2016-02-12	31:30	Zato->Tonni FTP-kirjoituksen testi service	01:00	Testien koodaus, ahman testien testaus	01:00
	2016-02-12	31:30	Zato FTP-tarkistus service lähettää virheistä mailin	00:30	Mailin lähetys luotu, testattu, ja toimii	00:30
	2016-02-17	31:30	Zato->Odoo Myyntitilausintegraation test.xml & fail.xml	01:00	xml:n suunnittelu	01:00
	2016-02-17	31:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	testi1: keräily valmis->xml:n kirjoitus	00:30
	2016-02-18	31:30	CO: Integ -> Odoo / tuotteet	09:45	tuoteintegraation käyttöönotto	01:30
	2016-02-19	31:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	sprint2 testi 1: testimyynti, keräily ja siirto takaisin tonniin	03:30
	2016-02-22	31:30	FIX Hae käyttäjää ext_username-kentän perus-teella	00:15	hakukentät vaihdettu	00:15
	2016-02-22	31:30	FIX Pistä myynnin käsittelijän tieto kenttään	00:15	yritin löytää tiedon xml:stä, ei ollut, viesti Tonni-kehittäjälle	00:15

			sale.order.user_id			
	2016-02-23	31:30	Zato->Odoo odoo-out service	01:30	Multi Search Terms: koodaus ja testaus	01:30
	2016-02-23	31:30	Zato->Odoo Tuotteiden asiakaskoodit	01:30	asiakaskoodien koodaus	00:45
	2016-02-24	31:30	Tonni->Zat->Odoo mt_totatks->delivery.carrier	02:00	selvitys, koodaus, testaus	02:00
	2016-02-24	31:30	Tonni->Zato->Odoo Myyntitilauksen T-rivit	00:30	testattu ja toimii	00:30
	2016-02-24	31:30	Zato->Odoo Myyntitilauksen PK & TK-rivit: koo-dattu ja testattu	01:00		01:00
	2016-02-24	31:30	Zato->Odoo Tuotteiden asiakaskoodit	01:30	asiakaskoodien kirjoituksen testaus+debug	00:45
	2016-02-25	31:30	Tuoteintegraation livetys	22:00	ftp virheiden parempi hallinta	00:30
	2016-02-26	31:30	Zato->Odoo aseta myyntitilaus oikeaan tilaan luonnin jälkeen: koodattu ja testattu	00:30		00:30
	2016-02-26	31:30	CO: Integ -> Odoo / myyntitilaukset	14:30	myynti tilassa 1-8 -> Odoossa status on done: koodattu ja testattu	00:30
	2016-02-26	31:30	Jos tilaus poistetaan tai perutaan Tonnissa, merkitse se perutuksi Odoo:ssa	01:00	koodattu ja testattu	01:00
	2016-02-26	31:30	Tuotteiden asiakaskoodit tuotantoon	04:15	asiakkaat uudestaan tuotantoon, yritetty asiakaskoodeja, ei onnistu nykyisellä konfiguraatiolla	03:00
	2016-02-29	31:30	CO: Tonni -> Integ / myyntitilaukset	28:30	Tonni-kehittäjän MTIL-TESTI -mallin läpikäynti ja sisäänkoodaus, omien testi-xml:ien muutokset	01:00
	2016-02-29	31:30	Integraatio mock-up scriptit koulutusta varten	01:00	scriptin koodaus ja läpikäynti Aten kanssa	01:00
	2016-02-29	31:30	Tuotteiden asiakaskoodit tuotantoon	04:15	virheiden selvitys ja korjaus Aten kanssa, testaus testipalvelimella	00:45
	2016-02-29	31:30	Tuotteiden asiakaskoodit tuotantoon	04:15	tuotanto-odooseen sisäänajo	00:30
	2016-03-02	17:45	CO: Tonni -> Integ / myyntitilaukset	28:30	testi-ftp yhteys kuntoon, asiakaskoodien bugifiksi	01:00
	2016-03-03	17:45	CO: Tonni -> Integ / myyntitilaukset	28:30	asiakasdumpin sisääntuonti	02:30
	2016-03-03	17:45	CO: Tonni -> Integ / myyntitilaukset	28:30	myyntidumppien sisääntuontia	01:00
	2016-03-04	17:45	CO: Tonni -> Integ / myyntitilaukset	28:30	myyntidumppien sisääntuontia jatkuu, virheiden keräys ja korjaus	05:00
	2016-03-04	17:45	Myyntitilausintgraation Kenraaliharjoitukset	13:45	kenraaliharjoituksen suunnittelu ja valmistelu	01:15
	2016-03-07	17:45	Myyntitilausintgraation Kenraaliharjoitukset	13:45	1. kenraaliharjoitus	04:00
	2016-03-11	17:45	Zato schedulerin ongelmien selvitystä	00:30		00:30

	2016-03-17	17:45	Uusien XML-tiedostojen testaus ja muunnoksen koodaus	02:30	Uudet XML:t	01:00
	2016-03-17	17:45	Uusien XML-tiedostojen testaus ja muunnoksen koodaus	02:30	Vanhojen virheiden korjaus	01:30
	2016-04-06	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	väliajalla luotujen xml:ien kopiointi	00:30
	2016-04-06	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	väliajalla luotujen xml:ien ajo testipalvelimelle (asiakkaat), virheiden kirjaus & korjaus	00:45
	2016-04-07	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	havaintojen kirjaukset käyttöönottosuunnitelmaan	00:15
	2016-04-07	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	väliajalla luotujen xml:ien ajo testipalvelimelle (loput), virheiden kirjaus & korjaus	01:15
	2016-04-11	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	Muutokset gittiin	00:15
	2016-04-13	35:30	Keräilytoiminnallisuuden koulutus	04:45	koulutuspäivä	04:45
	2016-04-14	35:30	[FEAT] Laske Tonnin "Vapaana" oleva tuote-määrä	00:30	Luetun ymmärtäminen & tekemisen suunnittelu	00:15
	2016-04-14	35:30	[FEAT] Laske Tonnin "Vapaana" oleva tuote-määrä	00:30	Luetun ymmärtäminen & tekemisen suunnittelu	00:15
	2016-04-14	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	myynntien tulostamisen testaus	00:15
	2016-04-14	35:30	Myyntitilausintegraation Livetus	26:15	Uusien dumppien pyyntö Tonni-kehittäjälta	00:15
	2016-04-15	35:30	Myyntitilausintegraation Livetus	26:15	livetys (klo 16-18)	02:00
	2016-04-15	35:30	Myyntitilausintegraation Livetus	26:15	livetys (klo 18-21:30, 3:30h * 1,5)	05:15
	2016-04-15	35:30	Myyntitilausintgraation Kenraaliharjoitukset	13:45	myyntien tulostuksen testaus jatkuu	00:45
	2016-04-16	35:30	Myyntitilausintegraation Livetus	26:15	dumppien seuranta	03:00
	2016-04-16	35:30	Myyntitilausintegraation Livetus	26:15	virheiden korjaus & koodimuutokset	02:30
	2016-04-17	35:30	Myyntitilausintegraation Livetus	26:15	dumppien seuranta	03:00
	2016-04-17	35:30	Myyntitilausintegraation Livetus	26:15	virheiden korjaus & viimeistely	01:00
	2016-04-18	35:30	Myyntitilausintegraation Livetus	26:15	lähitukihnkilöinti	09:15