Osaamista
ja oivallusta
tulevaisuuden
tekemiseen

Otso Nurmi

Verifone maksupääteintegraatio Odooseen

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintätekniikan tutkinto-ohjelma

Insinöörityö

13.2.2024

Tekijä Otsikko Sivumäärä Aika	Otso Nurmi Verifone maksupääteintegraatio Odooseen 21 sivua 13.2.2024
Tutkinto	insinööri (AMK)
Tutkinto-ohjelma	Tieto- ja Viestintätekniikka
Ammatillinen pääaine	Ohjelmistotuotanto
Ohjaajat	Janne Salonen, Osaamisaluejohtaja, ICT

Työssä suunniteltiin ja toteutettiin Sprintit Oy:n käyttöön integraatio Verifone korttimaksupäätteen ja avoimeen lähdekoodiin perustuvan Odoo yritysohjelmiston kassajärjestelmän välille. Työssä myös tutustutaan Odoo yritysojelmistoon yleisellä tasolla.

Työ sisälsi tutustumista Odoo yritysohjelmistoon ja sen kassatoiminnallisuuksiin ja kassan kanssa käytettävän Raspberry Pi:n pohjalle rakennetun PosBox-laitteeseen. Työssä toteutettiin rajapintaratkaisu PosBoxin sekä Verifonen Yomani-maksupäätteen välille, sekä rajapinta PosBoxin ja selainliittymässä ajettavan Odoo POS ohjelmiston välille. Työssä myös testattiin, sekä toimitettiin järjestelmä asiakkaan omaan Odoo ympäristöön.

Työssä käytettiin pääasiassa Python ja JavaScript ohjelmointikieliä. Pythonilla toteutettiin PosBoxiin oma moduulinsa ja JavaScriptillä toteutettiin Verifone toiminnallisuuden mahdollistava moduuli selainympäristössä pyöritettävälle kassaohjelmistolle.

Tässä dokumentissa esitellään avoimen lähdekoodin Odoo yritysohjelmistoa ja käydään läpi maksupääteratkaisun kehitys ja toiminnallisuus, sekä ratkaisun toimittaminen asiakkaan tuotantoympäristöön.

Avainsanat	Integraatio, korttimaksaminen, ohjemistokehitys

Author Title Number of Pages Date	Otso Nurmi Verifone payment terminal integration to Odoo 21 pages 13 February 2024
Degree	Bachelor of Engineering
Degree Programme	Information and Communications Technology
Professional Major	Software Engineering
Instructors	Janne Salonen, Director of school, School of ICT

In this project an integration was designed and implemented between the Verifone card payment terminal and the open-source Odoo enterprise software's POS system for the use of Sprintit Oy. The project also involved getting to know the Odoo enterprise software at a general level.

The work included familiarizing with the Odoo enterprise software and its POS functionalities, as well as the PosBox device built on the Raspberry Pi platform for use with the cash register. An interface solution was developed between PosBox and the Verifone Yomani payment terminal, as well as an interface between PosBox and the Odoo POS software running in the browser interface. The system was tested and delivered to the customer's own Odoo environment.

The programming languages used in the project were mainly Python and JavaScript. Python was used to implement a module for PosBox, and JavaScript was utilized to create a module enabling Verifone functionality in the browser-based POS software.

This document presents the open-source Odoo enterprise software and discusses the development and functionality of the payment terminal solution, as well as the delivery of the solution to the customer's production environment.

Keywords	Integration, card payment, software development

Sisällys

Lyhenteet

ERP	Enterprise resource planning. Integroitu ohjelmistojärjestelmä liiketoiminnan prosessien hallintaan
SaaS	Software as a service. Toimitusmalli, jossa palveluntarjoaja palvelun ylläpidosta.
XML	Extensible Markup Language. Yleiskäyttöinen merkintäkieli.
ESC/POS	Epson Standard Code for Point of Sale. Yleisesti käytetty standardi kassalaitteiden ja tulostimien ohjaamiseen.
USB	Universal Serial Bus. Yleinen standardi tietokoneiden ja laitteiden väliseen viestintään.
LTS	Long-Term Support. Lisäliite ohjelmistoversiolle, joka saa pitkäaikasta tukea ja ylläpitoa
IP	Internet Protocol. IP osoite on numeerinen tunniste laitteeseen tai liitäntään tietoverkossa.
ECR	Electornic Cash Register. Sähköinen kassakone
HTTP	Hypertext Transfer Protocol. Protokolla jota käytetään tiedon siirtoon verkon yli.
ASCII	American Standard Code for Information Interchange. Merkistökoodausjärjestelmä kirjaimille, numeroille ja merkeille.
ENQ	Enquiry. ASCII koodissa käytetty ohjausmerkki.
ACK	Acknowledge. ASCII koodissa käytetty ohjausmerkki.
JSON	JavaScript Object Notation. Tiedon siirrossa käytetty JavaScriptiin pohjautuva tiedonvaihtomuoto.

ISO	ISO koodit ovat 3 merkkisiä koodeja joilla ilmoitetaan kieliä, maakoodeja ja valuuttoja.
PIN	Personal Identification Number. Henkilökohtainen tunnusluku
MAC	Media Access Control Address. Laitteen uniikki tunniste verkossa.

1 Johdanto

Insinöörityössä suunniteltiin ja toteutettiin Sprintit Oy alaisuudessa integraatio Verifonen Yomani korttimaksupäätteelle Odoo-yritysohjelmiston kassaohjelman välille. Työ toteutettiin Sprintit Oy:n Retail and Manufacturing osaston palveluksessa.

Sprintit Oy on vuonna 2014 perustettu yritysohjelmistoratkaisuihin erikoistunut yritys, jonka palveluksessa on yli 40 työntekijää Suomessa ja muutamia työntekijöitä Intian toimipisteellä. Sprintit Oy:n pääasiallinen tuote on Odoo-yritysohjelmiston pohjalle rakennetut yritysohjelmistoratkaisut useille toimialoille.

Verifonen Yomani korttimaksupääte on Suomen käyetyin maksupääte. Odoon kassaohjelmistoon ei ennestään ollut Suomessa yleisille maksupäätteille yhteensopivuutta, joten työnannoksi tuli suunnitella ja toteuttaa integraatio kassaohjelmiston ja Yomanin välille.

Olennainen osa integraatiota on Odoon Raspberry Pi 3 yhden piirilevyn järjestelmän pohjalle rakentama järjestelmä. PosBox toimii rajapintana kassaohjelmiston, sekä kaikkien kassalaitteiden välillä. PosBoxissa suoritetaan muokattua Odoo-ohjelmaa, joka sisältää ainoastaan moduulit kassalaitteiden ja kassaohjelman väliselle tietoliikenteelle. Osa insinöörityötä oli ohjelmoida moduli Posbox-järjestelmään, joka välittää maksukomennot kassalta Verifonen maksupäätteelle.

Insinöörityössä käydään kokonaisuudessaan läpi vaiheet toteutuksesta, testauksesta aina järjestelmän käyttöönottoon asiakkaan tuotantoon.

2 Odoo

2.1 Odoon historia

Odoo on avoimen lähdekoodin toiminnanohjausjärjestelmä, jonka on kehittänyt Belgialainen Odoo S.A, alkuperäiseltä nimeltään OpenERP S.A. Odoota käyttää nykyään maailmanlaajuisesti yli 12 miljoonaa käyttäjää. (1). Odoo S.A perustaja Fabien Pinckarens aloitti kehittämään toiminnanohjausjärjestelmää nimellä TinyERP. Vuoteen 2010 mennessä yritys toimi nimellä OpenERP yli sadan työtekijän voimin (2). Vuonna 2015 Odoo S.A siirtyi open core liiketoimintamalliin, joka tarjoaa ilmaisen Odoo Community version rinnalle maksullisen Odoo Enterprise ohjelmiston, sekä Odoo SaaS pilvipalvelut. Vuoden 2024 alussa Odoo S.A:lla työntekijöiden määrä oli yli 2800, Odoo kumppaneiden määrä yli 5000 (1).

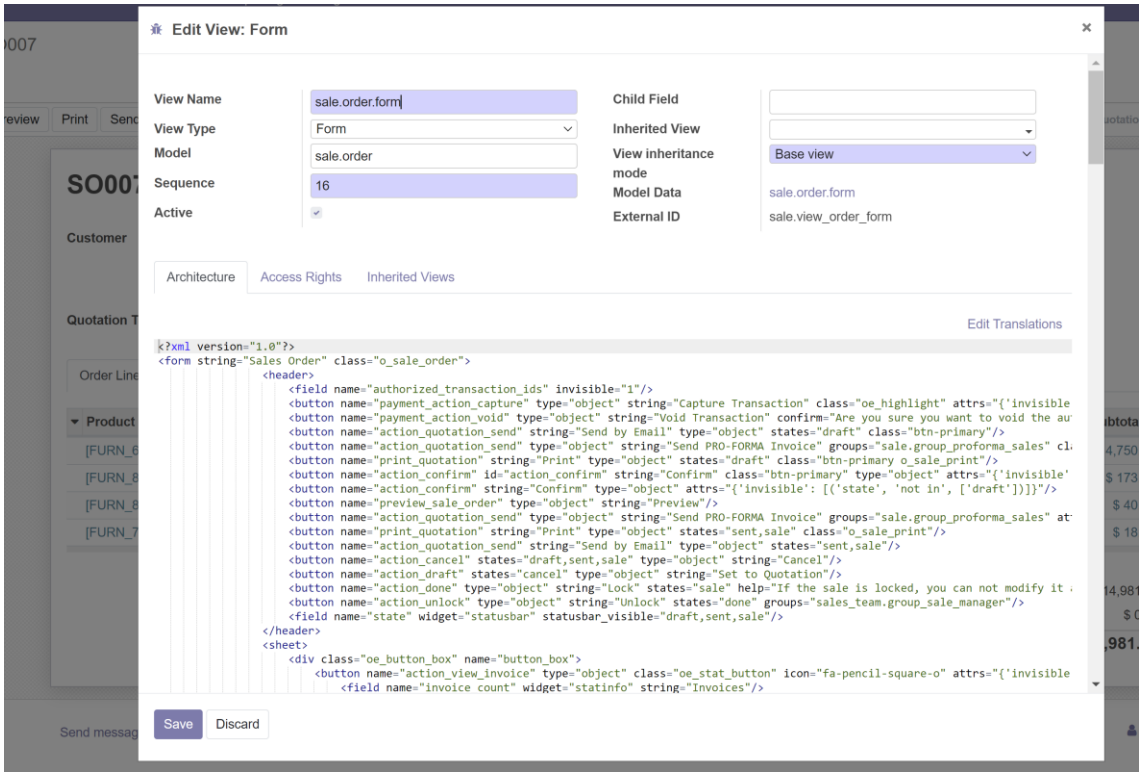
2.2 Yhteisö

Yksi merkittävä Odoon kehittäjä Odoo S.A:n lisäksi on Odoo Community Association, OCA, joka on vuonna 2014 perustettu voittoa tavoittelematon organisaatio. Organisaatio yhdistää kehittäjiä ja Odoo kumppaneita tarkoituksenaan kehittää avoimen lähdekoodin moduleita ja rajapintoja Odoo järjestelmään (3).

2.3 Modulaarisuus

Odoo perustuu modulaariseen sovelluspohjaiseen arkkitehtuuriin joka mahdollistaa järjestelmän räätälöinnin vastaamaan tarkasti asiakkaan tarpeita. Tässä etuna on joustavuus toteuttaa ainoastaan toiminnallisuudet, jotka ovat tarpeellisia liiketoiminnassa ja välttää tarpeettomat toiminnallisuudet jotka monimutkaistavat käyttökokemusta. Joustavuudesta on myös apua, sillä uusien toiminnalisuuksien aktivoiminen käyttöjärjestelmään on mahdollista tarpeiden muuttuessa.

Odoo tarjoaa myös laajat käyttäjäystävälliset tavat kustomoida käyttöjärjestelmää suoraan käyttöjärjestelmästä ilman laajempaa ohjelmistokehitystä. Odoon käyttöliittymät ovat rakennettu vahvasti XML-kuvauskielellä ja näkymien koodirakenteen saa auki suoraan käyttöliittymästä, joten pienten muutosten tekeminen on vähäisellä teknisellä osaamisella helppoa. Esimerkkeinä pienistä muutoksista on tietokenttien tuominen näkyviin käyttöliittymään tai niiden piilottaminen, uudelleen sijoittaminen ja järjestely(Kuva 1).



Kuva 1 Kuvakaappaus Odoo käyttöliittymän näkymäeditorista

Odoo on myös julkaissut versiossa 10 Odoo Studio työkalun, joka on graafinen käyttäjäystävällinen tapa tehdä muutoksia järjestelmään. Sillä esimerkiksi voi kehittää yksinkertaisia sovelluksia, tehdä parannuksia tulosteisiin sekä kustomoidoida näkymiä.(6). Tämä tekee Odoosta helpommin lähestyttävän pienille yrityksille, joilla ei ole suuria resursseja ostaa ulkopuolista kehitystyötä.

3 Integraation toteutus

3.1 Laitteisto ja materiaalit

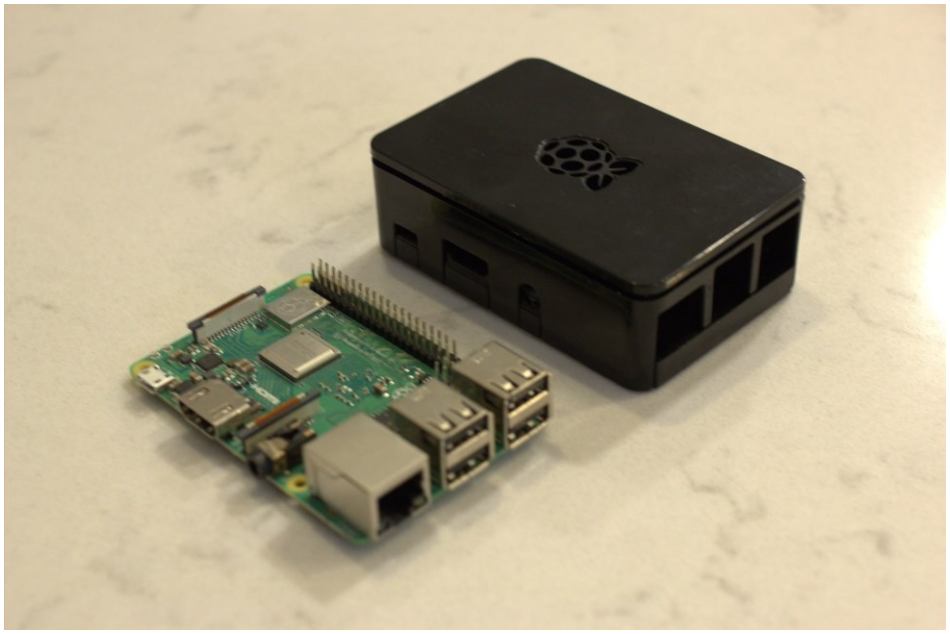
Integraation toteuttamiseksi tilattiin Verifonelta kehitys- ja testikäyttöön tarkoitettu maksupääte. Maksupääte on konfiguroitu käyttämään Verifonen maksutapaoperaattorin testijärjestelmää, joka kommunikoi maksupäätteelle kuin todellisissa maksutapahtumissa. Nordea pankilta tilattiin valikoima testipankkikortteja eri ominaisuuksin, kuten lähimaksuominaisuudella ja luottokorttiominaisuuksin. Yksi korteista oli sellainen, jolla ei ole katetta. Katteeton kortti mahdollistaa tilanteissa, joissa asiakkaalla ei ole katetta maksaa veloitettavaa summaa testaamisen. Korteilla suoritettavat maksut eivät ole oikeita, eikä niiden käyttö aiheuta oikeaa maksuliikennettä tai kuluja. Verifonelta tilattiin myös integraatiomanuaali, josta käy ilmi maksuprosessi ja siihen liittyvä viestiliikenne, sekä viestien syntaksi.

Kuva 2. Kehitystyössä käytetty testimaksupääte

Kehitystyöhön vaadittiin myös kuittitulostin, ja laitteeksi valikoitui Citizen s651 kuittitulostin. Laitteen valinnassa ainoa valintaperuste oli sen yhteensopivuus

ESC/POS komentokielen kanssa (7). Yhteensopivuus on tärkeää, sillä Odoon PosBox ohjelmistoon kuuluu esiasennettu hw_escpos moduli joka huolehtii integraatiosta kuittitulostimen osalta. Integraatio tukee siis kaikkia ESC/POS komentokieltä tukevia USB-liitännällä varustettuja kuittitulostimia.

Työhön hankittiin myös Raspberry Pi 3 tietokone suojakotelolla, joka toimii alustana POSBox ohjelmistolle. Tallennustilaksi Raspberry Pi:lle tuli 16gb microSD muistikortti, jolle asennetaan Odoon levykuva Ubuntu -käyttöjärjestelmästä josta löytyy Odoon ohjelmisto valmiina.



Kuva 3. Kehitystyössä käytetty Raspberry Pi 3 tietokone ja musta suojakotelo.

Kehitysympäristönä toimi Raspberry Pi:n lisäksi myös henkilökohtaiselle Windows käyttöliittymäiselle kannettavalle asennettu Odoo 12 enterprise, sillä se oli kehitystyön aikaan tuorein versio Odoosta. Odoo ympäristöön asennettiin Odoon kassaohjelmisto ja modulit joista se on riippuvainen, kuten account, sale ja inventory. Lisäksi kannettavalle asennettiin PostgreSQL tietokanta, joka on vaatimus Odoo ohjelmistolle.

Työssä tehtäviä python, javascript, ja xml koodeja varten käytin Notepad++

ohjelmistoa. Tämä on henkilökohtainen preferenssi ohjelmiston yksinkertaisuuden takia ja työn olisi voinut suorittaa myös muilla koodieditoreilla, joissa on enemmän ominaisuuksia kuten vaikkapa integroitu versionhallinta. Versionhallintaan käytettiin Git:in Windows versiota.

3.2 Kehitysympäristön pystytys

Kehitystyö aloitettiin valmistelemalla tarvittava ympäristö kehitystyötä varten. Kehitystyöhön tarkoitetulle kannettavalle ladattiin internetistä PostgreSQL tietokannan hallintajärjestelmä, jonka jälkeen se asennettiin perusasetuksin. Asennuksen yhteydessä luotiin tietokannalle käyttäjä ja salasana, sekä määriteltiin mistä verkkoportista tietokantaan pääsee kiinni.

Seuraava vaihe oli asentaa Odoo 12 enterprise. Windows ympäristöön asentamiseen löytyy asennusohjelma Odoon kotisivuilta. Asennus tekee asennuksen automaattisesti ja se myös huomaa automaattisesti järjestelmään asennetun PostgreSQL ohjelmiston. Asennuksen sijainti määritellään asennuksen yhteydessä, tässä tapauksessa C-levyn juureen. Asennuksen jälkeen, ohjelma käynnistää Odoo palvelun. Odoo palvelu oletuksena pyörii portissa 8069, joten sitä pääsee käyttämään nettiselaimella syöttämällä osoitekenttään localhost:8069.

Ensiasennuksen jälkeen selaimeen aukeaa Odoon tietokannan luontinäkymä. Näkymässä tietokannalle annetaan Odoon tietokannan pääavain, jota käytetään hallinnollisiin tarkoituksiin. Pääavain antaa oikeudet luoda, varmuuskopioida, palauttaa ja poistaa tietokantoja Odoon tietokantojen hallintajärjestelmässä. Muita syötettäviä tietoja pääavaimen lisäksi ovat luotavan kannan nimi, pääkäyttäjätunnus ja salasana, Odoon pääkieli, valtio automaattisia lokalisaatioasetuksia varten. Lisäksi asennukseen voi valita valmiin demodatan, jolloin tietokannan luonnin yhteydessä Odoo luo kantaan valmiiksi kehitystyössä hyödyllistä valmista dataa. Näin esimerkiksi varastosta löytyy valmiita tuotteita, joilla voi tehdä myyntitapahtumia kassaohjelmassa. Lopuksi hyväksytään syötetyt tiedot ja Odoo luo tietokannan.

Odoon luotua tietokannan aukeaa Odoon käyttöliittymä. Odoon kassaohjelmisto ei ole

oletuksena asennettu, joten se asennettiin odoon Apps sovelluksesta. Lisäksi asensin Sale modulin, sekä Inventory modulin. Tässä vaiheessa päätelaitteen kehitysympäristö on valmis.

3.3 Testilaitteiston pystytys

Kehitysympäristön pystytyksen jälkeen oli vuorossa testilaitteiston saattaminen käyttökuntoon. Laitteisto sisälsi Verifone Yomani testimaksupäätteen, Citizen kuittitulostimen, kassalippaan, testipankkikortit Nordealta sekä, Raspberry Pi keskusyksikön. Muu laitteisto ei vaatinut erityisempää konfiguraatiota, mutta Raspberry Pi keskusyksikköön piti asentaa käyttöliittymä.

Käyttöliittymän asennus tapahtui polttamalla MicroSD muistikortille Odoon sivuilta ladattava POSBox levykuva. Levykuva sisältää Ubuntu 16.04 LTS käyttöjärjestelmän, sekä sen päälle asennetun POSBox version Odoosta. Arkkitehtuuriltaan tämä versio Odoosta on identtinen tavallisen Odoo asennuksen kanssa, mutta se on ominaisuuksista riisuttu versio. Ohjelmiston riisuminen mahdollistaa ohjelmiston suorittamisen pienitehoisemmalla laitteistolla. Versiosta on riisuttu muun muassa kaikki tavallisen Odoon moduulit ja tilalle on tuotu kassalaiteintegraatioihin tarkoitetut hardware moduulit. Näistä yksi tähän kassaratkaisuun merkityksellinen oli hw_escpos moduuli, joka mahdollistaa kuittitulostimen liittämisen järjestelmään, sillä käyttämämme kuittitulostin käyttää ESC/POS standardia. Myöhemmin maksupäätteelle viedään toinen työssä kehitettävistä moduuleista.

Kun käyttöliittymän asennus oli valmis, tuli aika kytkeä kaikki laitteet yhteen. POSBox saa virtansa micro USB-liittimen kautta ja sen mukana tullut verkkovirta-adapteri huolehtii sen virransaannista. POSBox toimii koko järjestelmän sydämenä ja kaikki laitteet kassalipasta lukuunottamatta ovat siihen joko fyysisesti tai sisäverkon kautta suoraan yhteydessä.

Kuittitulostin, sekä maksupääte yhdistettiin POSBoxiin USB-kaapeleilla. POSBox:issa on neljä USB-A naarasporttia ja sekä maksupäättessä, että kuittitulostimessa on molemmissa USB-B naarasportit, joten yhteydet molempiin muodostettiin USB A-USB

B kaapelein. POSBoxissa on yleisesti käytössä oleva RJ45 liitin, jonka avulla kytkettiin RJ45 liitimisellä CAT5 parikaapelilla sisäverkkoon. Maksupääte ottaa virtansa USB portista johon se on yhdistettynä, kuittitulostin kytketään suoraan verkkovirtapistokkeeseen. Kuittitulostimeen liitetään RJ11 liittimellä kassalipas. Yhteys kuittitulostimen ja kassalippaan mahdollistaa kassalaatikon automaattisen aukeamisen maksutapahtuman onnistuessa, kuitin tulostumisen yhteydessä. Päätelaite kytketään samaan sisäverkkoon, johon PosBox on yhdistettynä.

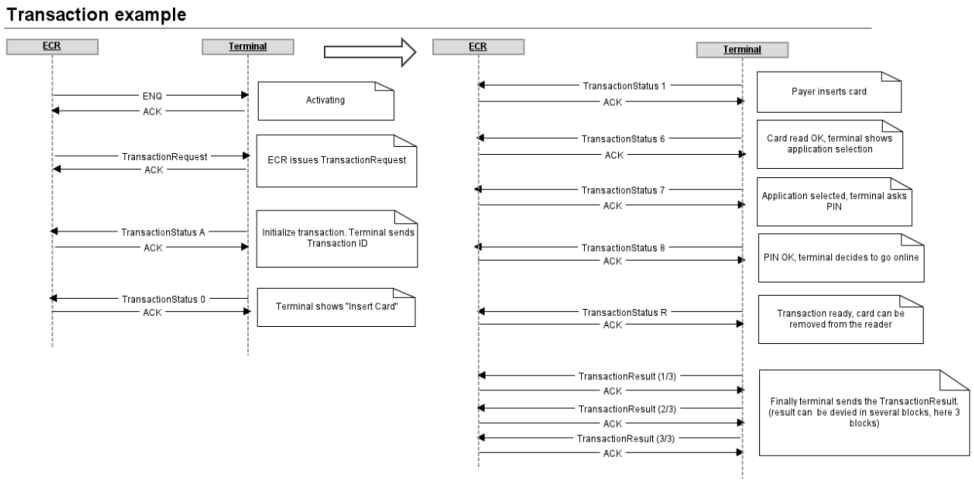
Päätelaite yhdistetään POSBoxiin päätelaitteen Odoon selainkäyttöliittymästä. Kun POSBox käynnistyy se tulostaa kuittitulostimella kuitin, jossa se ilmoittaa sisäverkon IP-osoitteen. Tämä sisäverkon IP-osoite syötetään Odoon Point of Sale moduulin asetusvalikossa yhteysosoitteeksi PosBoxiin. Nyt testiympäristö on toimintakuntoinen ja valmis kehitystyölle.

3.4 Hardware moduuli

Integraatio koostuu kahdesta moduulista. Odoo-palvelimelle asennettavasta moduulista, joka huolehtii integraatiosta Odoon ja POSBoxin välillä, sekä hardware -moduulista, joka huolehtii integraatiosta POSBoxin ja maksupäätteen välillä. Tässä osassa työtä kuvaamme kahdesta jälkimmäistä.

Käytin modulin runkona Odoo appstoresta ladattua Odoo OCA:n kehittämää hw_telium_payment_terminal modulia. Tämä moduli eroaa suuresti lopullisen ratkaisun vaatimuksista, sillä se noudattaa aivan erilaista toimintamallia, kuin lopulliseen ratkaisuun tarvittava. Kyseinen moduuli on kehitetty toimimaan Telium protokollaa käyttävillä maksupäätteillä(8) ja toimintaperiaate on huomattavasti simppelimpi, mutta siitä sai runkoa esimerkiksi tarvittaviin python kirjastoihin ja maksupäätteelle lähetettäviin viesteihin. Kuitenkin Verifonen maksupäätteelle vaadittava koodi oli rivimäärällisesti huomattavasti Telium moduulia suurempi. Teliumin moduuli koostui 300 rivistä koodia, kun työn Verifone maksupääteratkaisu vaati liki 1000 riviä koodia.

Kehitystyössä tärkeä työkalu oli Verifonelta saatu integraatio-opas. Integraatio-oppaassa kuvataan maksutapahtumaa prosessikaavioissa ja viestien syntaksia. Alla olevassa kuvassa ote Integraatio-oppaasta, jossa havainnollistetaan maksutapahtumaa ja sen aikaista keskustelua ECR:än ja Maksupäätteen välissä. Tässä tapauksessa ECR on POSBox. Vuokaavion mukaisesti ECR lähettää maksupäätteelle määrämuotoisen viestin, jolla maksutapahtuma aktivoituu. Maksupääte kuittaa jokaisen komennon vastaanotetuksi. Nämä kuittaukset tulee myös erikseen ECR:än lukea maksupäätteen sarjaportista. Eli kuittaus ei automaattisesti perille, vaan koodiin tuli kirjoittaa funktio joka aina viestien välillä käy tasaisin väliajoin lukemassa sarjaporttia. Transaktion aikana maksupääte myös lähettää status viestejä, jotka antavat maksutapahtuman vaiheesta tietoa. Näitä tulee ECR:rän myös lukea jatkuvasti ja viestin luettuaan myös kuitata se maksupäätteen suuntaan. Mikäli jokin vaiheista venyy liian pitkäksi, tulee aikakatkaisu ja maksutapahtuma keskeytyy. Eli koodissa portin lukuväli tuli asettaa sopivan lyhyeksi, jotta viestit tulevat kuitatuiksi ajoissa. Myöskin kun odotetaan maksupäätteeltä kuittausta, tulee viestin lähetyksen jälkeen olla pieni viive, ennen kuin koitetaan lukea kuittausta. Lisäksi kuittausta koitetaan hakea kolmesti, ennen kuin maksutapahtuma keskeytetään. Näin saadaan eliminoiduksi pienistä viiveistä johtuvat virheet. Kuva 1 havainnollistaa tavanomaisen maksutavan maksuliikenteen vaiheita.



Kuva 4. Sekvenssikaavio integraatio-oppaasta (3).

Komennot kassapäätteeltä tulee käyttäen HTTP-reittejä. Ne ovat osotteita joihin on sidottu ohjelman funktioita jotka ajetaan kun osoitetta kutsutaan. Yhteys kassapäätteen ja POSBox:in välille on rakennettu näillä reiteillä. Ratkaisuun ohjelmoitiin maksutapahtuman käynnistykselle, keskeytykselle, uudelleenkäynnistykselle komennot. Ratkaisuun kehitettiin myös HTTP-reitit, joista maksupääte hakee tiedot maksutapahtuman tilasta, päättymisestä ja kuittille tulostettavat maksutiedot.

Maksutapahtuma POSBox:in ja Maksupäätteen välille käynnistyy, kun Odoon kassaohjelmasta lähetetään ensimmäinen sanoma HTTP-reittiä pitkin. Sanoma sisältää maksutapahtumaan tarvittavat tiedot JSON objektina. Sanoma myös alustaa koodissa käytettäviä tilamuuttujia edellisen maksutapahtuman jäljiltä. Sanoma käynnistää transaction_start funktion, jolle välitetään parametrina vastaanotettu JSON-objekti.

Start_transaction funktio käyttää simplejson python kirjastoa, joka purkaa JSON-objektin python sanakirjaksi. Funktio myös käyttää initialize_msg funktiota, joka lähettää ensimmäisen ENQ komennon joka toimii herätteenä maksupäätteelle ja varmistuu ACK vastauksen vastaanottamisesta. Vastauksen vastaanottamisen jälkeen se palauttaa boolean muuttujana toden transaction_start funktioon jonka suoritus jatkuu.

Start_transaction välittää aiemmin JSON-objektista puretun sanakirjan prepare_data_to_send funktioon, joka poimii datasta lopulliseen sanomaan viestissä välitetyt tiedot ja lisää myös muita parametreja. Lopuksi funktio kokoaa muuttujista tiedot data nimiseen sanakirjaan joka on motoiltu integraatiooppaan mukaiseksi sanomaksi ja palauttaa sen takaisin transaction_start funktioon, joka lähettää sen send_message funktioon.

Field	Length	Content	Description	Type
0	1	MessageID	transactionRequestEx = 'X' or 'x' for single block result message	(A)
1	1	Transaction type	0 = normal purchase, 1 = cash withdrawal, 2 = reversal, 3 = refund 4 = retrieve, 5 = quasi cash, 6 = cashback, P = Initial Preauthorization	(A)
2	7	Amount, transaction currency	Total transaction amount, always an absolute value.	N
3	7	Amount, Cashback	The cashback part of transaction amount, must be less than amount in field 2	N
4	5	TransactionID	TransactionID (terminal's receipt number) for reversal and retrieve (and for refund in some countries)	N
5	1	Force Authorization	1 = Force online authorization regardless of the amount, 0 = Terminal resolves the need of authorization.	(N)
6	1	Manual card number	3 = Telephone order 2 = Mail order 1 = Terminal requests card number to be keyed in, instead of reading card 0 = Card should be read (chip or mg-stripe)	(N)
7	1	Bonus handled	1 = yes, 0 = no	(N)
8	7	Authorization code	Received from manual authorization (voice referral) Note! The code is variable length 4..6 characters and should always be terminated by a FS (0x1C). FS in the first character position means 'no code given'.	A
9	12	Timestamp	YYMMDDhhmmss of the original Transaction for reversal and retrieve	(N)
10	9	SerialNo	Reader serial number for reversal (Optional; the terminal uses it's own number, if this field is zeroes).	A
11	1	Payment method restriction	Single ASCII hex character bitmap, explained on page 10	(A)
12	1	Surcharge handled	unused	(A)
13	1	LookForDOB	1=Send Cardholder DateOfBirth (special status message), 0=don't send	(N)
14	2	ECR number	ECR number used in transaction reference. Alphanumeric characters accepted. Other values: Terminal's own number is used.	(A)
15	1	Flags	Used in retrieval.	(A)
16	6	RFU	Reserved for Future Use	A

Total length 64 bytes.

Kuva 5. Integraatio-oppaan kuvaus lähetettävästä viestistä (3).

Send_message funktio purkaa datan string muuttujaan ja tarkistaa sen olenvan integraatio-oppaan määrämän mittainen. Seuraavaksi funktio lisää viestin aloitus STX ja ETX merkit viestiin merkitsemään viestin alkua ja loppua. Funktio myös laskee generate_lrc funktiota käyttäen tarkistussumman sanomalle ja se liitetään sanoman loppuun. LRC lasketaan XOR operaatiolla. LRC tarkistussummaa käytetään datan eheyden varmistamiseksi vastaanottavassa päässä. Viimeiseksi funktio muuttaa viestin tavujonoksi ja lähettää sen maksupäätteelle serial_write funktiolla. Lähetyksen jälkeen koodin suoritus palaa vielä transaction_start funktioon ja se tarkistaa get_onge_byte_answer funktiolla, että maksupääte vahvistaa sanoman vastaanottamisen ACK sanomalla. Nyt maksutapahtuma on käynnissä ja maksupääte antaa summan asiakkaalle ja jää odottamaan maksukorttia.

3.5 Kassapäätemoduuli

Kassapäätteen moduuli luotiin teknisellä nimellä sprintit_payment_terminal. Tähän otettiin rungoksi pos_payment_terminal, joka on Odoo OCA:n moduli Teliumin maksupääteintegraatiota varten. Tästä moduulista ei suurta osaa pystynyt käyttämään, mutta se toi pari laajennusta Odoon kassa-asetusten tietomalliin ja pos_config, sekä

kirjanpidon tietomalliin account_journal. Laajennukset pitivät mukanaan kassaan boolean valintaruudun, jolla voi valita maksupäätteen aktiiviseksi kassaan. Kirjanpitopuolelle laajennus loi monivalinnan, jossa voi asettaa kassaan luotuun maksutapaan tyypiksi käteisen ja lisäksi kortin.

Liikenteeseen kassapäätteen ja maksupäätteen välille OCA:n modulin toteutuksesta ei saanut runkoa, sillä lopullinen toteutus on mutkikkaampi. OCA:n toteutuksessa lähetetään vain yksi komento maksutavan käynnistämiseksi ja vastaanotetaan vahvistus onnistuneesta maksutavasta. Näin maksutapahtuma tapahtuu maksupäätteellä kokonaisuudessaan ja kassalaitteelle ei tule tietoa maksutavan vaiheista maksutapahtuman aikana. Myöskin maksun korttitiedot sisältävä kuitti tulostuu suoraan maksupäätteeltä, eikä odoon kautta. Tässä tavassa on se heikko puoli, että Suomessa on totuttu saman kuitin sisältävän sekä tuote-erittelyn, sekä palveluntarjoajalta tulevat korttimaksutulosteet. OCA:n toteutuksessa nuo kuitit olisivat kaksi erillistä.

Tämän työn toteutuksen tavoite oli kehittää ratkaisu, jossa kassalaitteelle tulee reaaliaikaisesti tiedot maksutapahtuman kulusta ja virhetilanteista. Lisäksi maksupääte ei itsessään tue kuitin tulostusta, vaan onnistuneen maksutapahtuman päätteeksi se palauttaa palveluntarjoajalta tuleet korttimaksutulosteet, jotka järjestelmän tulee pystyä liittämään Odoon vakiomalliseen kuittiin ja lopuksi tulostamaan kuitin automaattisesti POSBox-järjestelmään liitetyllä kuittitulostimella.

Ensimmäinen vaihe oli luoda funktiot, joilla käynnistetään maksutapahtuma ja aloitetaan säännöllisesti lukemaan POSBox:in HTTP-rajapinnasta tietoa maksutapahtuman tilasta.

Käynnistysfunkio saa parametreissaan line_cid, currency _iso, sekä currency_decimal muuttujat. Funktiossa ensimmäisenä luetaan käynnissä olevan kassaistunnon maksurivit listaan. Lista maksuriveistä syötetään for-lauseeseen, jossa niistä yksitellen tarkastetaan if-lauseessa, millä niistä on maksurivi joka on käsittelyssä. Tämä tehdään siksi, että mikäli maksutapahtumassa on jo toinen maksurivi se ei tule tapahtumaan mukaan. Mikäli ehto täyttyy, tallennetaan valikoitu rivi line muuttujaan.

Maksurivioliolla on funktio get_amount(), jolla haetaan amount muuttujaan maksurivin summa. Seuraavaksi tarkastetaan, onko amount pienempi kuin 0, siltä varalta onko kyseessä rahan palautus. Mikäli kyseessä olisi palautus, muutettaisiin type muuttuja muotoon 3, joka ilmoittaa sanomassa olevan kyseessä rahan palautus asiakkaalle. Summa pyöristetään lopuksi kahteen desimaaliin toFixed() funktiolla.

Seuraavaksi funktiossa kootaan JSON-objekti, johon syötetään POSBoxille välitettävät parametrit. Parametreissa lähetetään funktion parametreina saamat currency_decimal joka välittää käytössä olevien desimaalien määrän, sekä rahayksikön määrittävän ISO 4217-standardin mukaisella kolmikirjaimisella koodilla käytettävän rahayksikön. Parametrit sisältävät myös muodostetun summan sekä type muuttujan.

Koottu sanoma lähetetään POSBoxin HTTP-reittiin, joka käynnistää edellisessä alaluvussa kuvaillun transaction_start funktion. Lopuksi käynnistyy get_status() funktion säännöllinen suorittaminen, jolla haetaan tieto maksutavan tilasta ja lopussa maksutapahtuman päättymisestä. Tämä on toteutettu javascriptin setInterval() funktiolla joka ottaa parametrina suoritettavan funktion, sekä millisekunteina ajan jonka välein annettua funktiota suoritetaan.

Käynnistynyt get_status() funktio saa nyt jatkuvasti tarvittavat tiedot maksutapahtuman tilasta. Kassahenkilön kuvaruudulle tulee punainen ponnahdusikkuna ja kassanäkymän muut elementit ovat lukossa estäen kassan käytön maksutapahtuman ollessa käynnissä. Ponnahdusikkuna on peritty kassankäyttöliittymän pos.gui.show_popup() funktio. Funktiolle annetaan parametreina ponnahdusikkunan tekstisisältö sekä painikkeiden tekstit ja painikkeisiin sidotut funktiot. POSBox lukee tasaisesti maksupäätteen antamaa dataa, ja POSBoxin koodi muuttaa luetun datan selkokieliseksi. Nämä selkokieliset merkitykset välitetään kassapäätteelle ja ne syötetään parametreina ponnahdusikkunan viestiin.

Kassamoduliin kehitettiin myös käsittely kassahenkilön toimenpidettä vaativiin käyttötapauksiin. POSBoxista saadaan myös muuttuja joka ilmaisee onko kyseessä jokin kassahenkilön toimenpidettä vaativa tapahtuma. POSBoxin modulin koodiin on määritelty kaikki Verifonen integraatiomanuaalin mukaiset maksupäätteen paluuviestit. POSBoxin maksupäätteeltä lukema statussanoma sisältää nelinumeroisen result

coden. Manuaalissa on annettu jokaiselle result codelle merikitys. Merkitykset on myös jaettu eri luokkiin. Result codet väliltä 000-999 ovat informatiivisia ja maksutapahtuma jatkaa kulkuaan seuraavaan vaiheeseen. 1000-1999 välillä oleva result code tarkoittaa virheeseen tai rajoitteeseen keskeytyvää maksutapahtumaa. Result codet väliltä 2000-2999 tarkoittaa tilannetta, jossa maksutapahtuma pysäytetään kassahenkilön toimenpidettä varten. Näissä tilanteissa maksutapahtuma jatkuu, kun toimenpide on suoritettu ja kassahenkilö kuittaa sen tehdyksi.

Kassahenkilön toimenpidettä vaativassa tilanteessa punaiseen ruutuun ilmeistyy valintapainikkeet confirm ja cancel, jossa kassan käyttäjä voi kuitata ikkunan viestissä määrätyn toimenpiteen suoritetuksi. Esimerkki toimenpidettä vaativasta tapahtumasta on, mikäli asiakas valitsee maksupäätteellä PIN-koodin ohituksen, tarvitsee kassahenkilön tarkistaa hänen henkilöllisyytensä ja kuitata ok. Mikäli henkilöllisyyden todentaminen ei onnistuisi, kassahenkilö valitsee cancel jolloin maksutapahtuma peruuntuu.

Kun maksutapahtuma on mennyt onnistuneesti läpi, POSBox lukee maksupäätteen TransactionResult sanoman jonka maksupääte lähettää ainoastaan onnistuneen maksutavan jälkeen. Tuo sanoma aiheuttaa POSBoxissa muutoksia tilamuuttujissa. Käynnissä olevaa maksutapahtumaa ilmaiseva transaction_ongoing muuttuja muutetaan ’not active’ tilaan. Myöskin payment_ok, muutetaan todeksi ilmaisemaan maksutapahtuman päättyneen onnistuneeseen maksuun. TransactionResult sanoma on kolmiosainen ja se sisältää myös kuittitekstit jotka manuaalin kuvauksen mukaisesti luetaan python sanakirjaan. Sanoma sisältää myös tiedon, mikäli maksutapa vaatii kauppiaan kuitin tulostamisen esimerkiksi luottokorttimaksuissa. Tällöin tulostetaan kauppiaan kuittiin myös allekirjoituskenttä, johon asiakkaan allekirjoitus kirjoitetaan.

```
answer_data = {
    'message_id': real_msg[1],
    'transaction_type': real_msg[2],
    'payment_method': real_msg[3],
    'card_type': real_msg[4],
    'transaction_usage': real_msg[5],
    'settlement_id': real_msg[6:8],
    'card_number': real_msg[8:27],
    'aid': real_msg[27:59],
    'tc': real_msg[59:75],
    'tvr': real_msg[75:85],
    'tsi': real_msg[85:89],
    'transaction_id': real_msg[89:94],
    'filling_code': real_msg[94:106],
    'dtt': real_msg[106:118],
    'amount': real_msg[118:125],
    'currency': real_msg[125:128],
    'reader_serial_number': real_msg[128:137],
    'print_payee_receipt': real_msg[137],
    'flags': real_msg[138],
    'payers_receipt': payer_r,
    'payee_receipt': (real_msg[i:])[:-1]
}
```

Kuva 6. Kuittidatan purku maksupäätteen sanomasta python sanakirjaan hardware moduulissa.

Kun kassapääte lukee maksutapahtuneen päättyneen onnistuneeseen maksutapahtumaan, se kutsuu viimeiseksi HTTP-reittiä payment_terminal_transaction_receipt. Reitistä se lukee kolmea dataa, asiakkaan kuittiin liitettävät tekstit, tiedon tulostetaanko myös kauppiaan kuitti, sekä kauppiaan kuittiin liitettävät tekstit.

Kun kassapäätteellä on tarvittava data suoritetaan koodissa maksutapahtuman vahvistus ja kuitin tulostus. Vakiossa Odoossa maksutapahtuma vahvistettaisiin käsin painikkeesta, josta siirryttäisiin kuitin tulostusnäkymään ja siitä edelleen painikkeella seuraavaan kassaistuntoon. Koodissamme tarkastellaan onko maksutapahtuma maksettu kokonaisuudessaan korttimaksutapahtuman päätyttyä. Tässä tapauksessa edellä mainitut vakio Odoon manuaaliset painallukset hoidetaan koodillisesti

kutsumalla painikkeiden funktiot järjestyksessä. Näin kuitti tulostuu automaattisesti ja ikkuna päivittyy suoraan valmiiksi uuteen kassataphtumaan. Mikäli maksutapahtuma ei olisi kokonaan maksettu, vaan asiakas esimerkiksi maksaisi kahdella kortilla tai osittain käteisellä, kassatapahtuma vaihtuisi uuteen vasta kun maksettava summa olisi kokonaisuudessaan suoritettu.

3.6 Testaus

Kun integraatio vaikutti toimivalta oli aika testata sen toiminta. Verifone toimitti meille testausohjeen, jolla suoritimme siinä luetellut testitapaukset. Testaus ohje käsitti kaksi kohtaa, jotka kummatkin sisälsivät testitapauksia. Kohta yksi käsitti perustoiminnat, Maksutapahtuma sirulla tai lähimaksulla sekä hyvitystapahtuman sirulla.(9.

Maksutapahtuma testattiin valitsemalla kassasta myytävät tuotteet ja siirryttiin maksuikkunaan. Maksuikkunassa valittiin maksutavaksi korttimaksu ja käynnistettiin maksutapahtuma. Korttimaksupääte kehotti syöttämään kortin ja syötettyäni kortin maksupääte kysyi PIN-koodia. Syötin pinkoodin ja maksu meni onnistuneesti läpi. Lopuksi kassaohjelma tulosti kuitin korttimaksuteksteineen.

Hyvitystapaus hoidettiin myymällä negatiivinen määrä hyvitettäviä tuotteita. Valitsin kassasta tuotteen ja valitsin sille negatiivisen määrän. Odoon kassa automaattisesti kääntää tapahtuman maksettavan summan negatiiviseksi. Siirryin maksuikkunaan, jossa valitsin maksutavaksi korttimaksun ja käynnistin maksutapahtuman. Maksupääte kysyi sirukorttia ja sen syötettyäni maksu meni automaattisesti läpi ja maksupääte tulosti kuitin.

Testausohjeen toinen osa sisälsi poikkeustapaukset. Poikkeustapaukset sisälsivät seuraavat käyttötapaukset, joissa tarvitaan erillistä henkilöllisyyden varmennusta. Tapauksia oli kaksi, toinen siirryttäessä magneettijuovaan sirun luvun epäonnistuessa, sekä PIN-koodin ohituksessa.(9). Sirun luvun epäonnistumista pystyi testata laittamalla kortin väärinpäin lukijaan. PIN-koodin ohitusta laitteen saa pyytämällä painamalla vihreää enter painiketta PIN-koodin syöttämisen sijasta. Molemmat käyttötapaukset

toimivat ilman huomauttamista ja kassapäätteelle ilmestyi valinta ikkuna, joka mahdollisti kassan käyttäjälle maksutapahtuman jatkumisen tai maksutapahtuman keskeyttämisen mikäli henkilöllisyyden varmentaminen ei olisi mahdollista.

Poikkeustapauksiin kuului myös testata kassan toimintaa asiakaslähtöisestä keskeytyksestä. Tätä testattiin sekä poistamalla kortti kesken tapahtuman, että painamalla punaista stop näppäintä. Maksutapahtuman keskeytyessä kassan näytölle ilmestyi viesti, joka ilmoitti asiakkaan keskeyttäneen maksutapahtuman. Lopuksi ikkuna palautui maksuikkunaan, josta voi käynnistää maksutapahtuman uudestaan.

Viimeinen testattava poikkeustapaus oli ”Ei myyntilupaa” käsittely. Testipäätteen ollessa kytkettynä Verifonen testipalveluun, se saa kieltäviä vastauksia tietyillä tapahtumasummilla 90,01€-99,99€. Varmennusvastauksen koodi 116 tarkoittaa, ettei kortinhaltijan tilillä ole riittävästi katetta. Tällöin kassalaitteen tulee ilmoittaa myyjälle ”Ei myyntilupaa” Toiminnallisuus varmistui toimivaksi ja testaukset päättyivät.

4 Tuotantoon vienti

Kun kehitystyö oli tullut päätökseen, oli aika viedä ratkaisu asiakkaan toimipisteeseen tuotantokäyttöön. Asiakas oli etukäteen ohjeistettu tilaamaan oikea maksupääte Verifonelta ja kirjoittamaan asiakassopimus maksupääteyhteydelle.

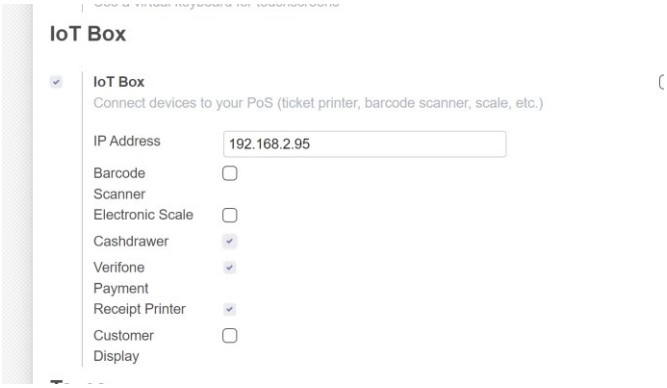
POSBox laitteet konfiguroitiin valmiiksi etukäteen. Ratkaisun vientiä monelle Raspberry Pi:lle helpottamaan luotiin levykuva alkuperäisen kehityskäytössä rakennetun POSBox:in muistikortista. Näin Rufus nimisellä ohjelmistolla levykuva on helppo polttaa monille muistikorteille, jotka toimenpiteen jälkeen toimivat suoraan sisältäen minun maksupäätteelle räätälöimän modulin.

Ennen asiakkaan toimipisteelle lähtöä varmistin, että heillä on sisäverkkonsa puolesta kaikki tarvittava ratkaisun asentamista varten. Kassalle tulee ratkaisua varten olla kolme verkkokaapelia. Yksi verkkokaapeli kassalaitteelle, jonka selaimessa kassaohjelmistoa käytetään, toinen maksupäätettä varten ja kolmas POSBoxia varten. Tarvittaessa POSBox tukee myös langatonta yhteyttä, mutta ratkaisun

toimintavarmuuden kannalta fyysinen yhteys on optimaalinen. Lisäksi heillä tuli olla verkon reitittimen tunnukset minua varten, jolla asetin POSBoxille staattisen sisäverkon IP-osoitteen. Tämä on tärkeää, sillä jos laitteisto irroitetaan sähköverkosta voi laitteen IP-osoite vaihtua ja kassajärjestelmään konfiguroitu POSBoxin osoite ei enään olisi oikea. Ratkaisussani POSBox saa aina saman MAC-osoitteeseen sidotun IP-osoitteen.

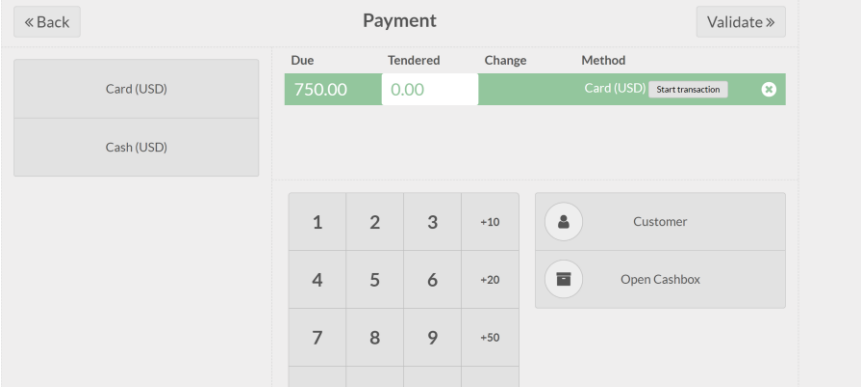
Asentaminen oli suhteellisen yksinkertaista, kun kaikki ennakkovaatimukset täyttyivät. Kuittitulostin kytkettiin verkkovirtapistokkeeseen ja USB A-USB B kaapelilla POSBoxiin. Maksupääte kytkettiin sisäverkkoon verkkokaapelilla ja USB A-USB B kaapelilla POSBoxiin. Maksupääte ei vaadi erillistä virtalähdettä, vaan se saa virtansa USB yhteyden välityksellä ja on aina käynnissä kun POSBox on käynnissä. Lopuksi yhdistettiin POSBox verkkokaapelilla sisäverkkoon ja yhdistettiin se omalla micro USB-virtalähteellä virtapistokkeeseen. Laitteiston ja verkon osalta asennus oli nyt valmis.

Lopuksi tehtiin konfigurointi asiakkaan jo muuten valmiiksi toimitettuun Odoo järjestelmään. Asetuksista valittiin POSBox aktiiviseksi, jolloin aukeaa valintaruudut jossa voi valita POSBoxiin kytketyt laitteet ja asettaa PosBoxin sisäverkon IP-osoitteen. Valitsin aktiiviseksi kuittitulostimen, kassalippaan, sekä moduulini luoman Verifone Payment valinnan. Kassalipas ei itsessään ole kytketty POSBoxiin, vaan sen avausta tukevaan kuittitulostimeen. Näin kuittitulostin toimiessaan avaa kassalippaan automaattisesti.



Kuva 7. POSBoxin konfigurointivalinnat. Myöhemmissä versioissa POSBox sai uuden nimen IoT Box.

Lopuksi konfigurointiin korttimaksutapa. Muuten korttimaksutapa vastaa esimerkiksi käteismaksutapaa, mutta sen Payment Mode valinnaksi valitaan moduulissamme luotu Card maksutapa, jolloin kassan maksuikkunassa se tarjoaa maksutapahtuman käynnistyspainikkeen maksuriville.



Kuva 8. Kuva Odoo kassajärjestelmän maksuikkunasta. Korttimaksutavalle lisätty painike Start Transaction painike maksurivillä.

Kun asennukset ja konfiguroinnit olivat valmiita, kokeilimme vielä yhdessä asiakkaan kanssa maksuliikenteen toimivuutta. Teimme pienen maksutapahtuman oikealla maksukortilla varmistuaksemme maksuliikenteen ja kuittitulosteiden toimivuudesta. Maksun suorittamisessa tai kuitin tulostuksessa ei havaittu ongelmia. Maksu oli kuitenkin myös tarkoitus palauttaa kortille, mutta tässä törmäsimme maksutapahtumaa käynnistäessä ongelmiin. Maksutapahtuma keskeytyi muutamasta yrityksestä huolimatta. Otin asiakkaan kanssa yhteyden Verfionen asiakaspalveluun ja syyksi paljastui, että maksupäätteen hyvitysmahdollisuus piti kytkeä erikseen päälle. Laitoin tämän muistiin tulevaisuuden kassalaitetoimituksia varten, jotta voin ohjeistaa asiakasta aktivoimaan ominaisuuden liittymäänsä etukäteen. Totesimme järjestelmän toimivaksi ja asiakas aloitti kassajärjestelmän käytön toimipisteessään.

Työ oli ensimmäinen varsinainen ohjelmistokehitystyöni työelämässä ja hetkittäin se haastoi osaamiseni. Tunsin kehitystyön mielekkääksi ja se oli mukava ensikosketus uravalintaan, johon opintoni tähtäsivät. Pidin toteutuksen monipuolisuudesta, sillä se vaati sekä ohjelmistokehityksen, että laitteisto ja verkkopuolen osaamista.

Varsinaisesta kehitystyöstä on jo vierähtänyt muutama vuosi ja tätä työtä kirjoittaessa

kävin koodia läpi, huomasin muutamia koodin osia jotka osaisin kirjoittaa nykyään tehokkaammin. Ratkaisussa haasteita toi epäsynkroniset yhteydet kassapäätteen, POSBox:in ja maksupäätteen välillä. Tämä on ratkaisun heikko kohta, sillä pienet sisäverkon pätkimiset on välillä saaneet kassatapahtuman keskeytymään. Tällöin asiakas on joutunut käynnistämään PosBoxin uudelleen. Tämän työn kirjoitushetkellä ratkaisu on edelleen eräällä asiakkaalla käytössä. SprintIT Oy ei kuitenkaan enään ylläpidä ratkaisua, vaan tarjoaa ensisijaisesti toisen maksuoperaattorin pilvipohjaista ratkaisua. Tämä ratkaisu on kehitysmielessä huomattavasti kevyempi ratkaisu sillä se ei vaadi POSBox:ia, vaan maksukomennot lähetetään suoraan pilveen mistä ne välittyvät maksupäätteelle.

Lähteet

1 Odoo. Verkkoaineisto. <https://www.odoo.com/page/about-us> Luettu 14.01.2024.

2 The Odoo story Verkkoaineisto. <https://www.odoo.com/blog/odoo-news-5/the-odoo-story-56> Luettu 14.1.2024.

3 Point ECR Standard Finland v4.5.1. PDF-dokumentti.

4. Rufus. Verkkoaineisto.<https://rufus.ie/en> Luettu 15.01.2024

5. Who is the OCA. Verkkoaineisto. <https://odoo-community.org/about>. Luettu 15.01.2024

6. Odoo Studio Overview. Verkkoaineisto. <https://www.odoo.com/app/studio-features>. Luettu 15.01.2024

7. Citizen CT-S651II. Verkkoaineisto. https://www.citizen-systems.com/en/products/printer/pos/ct-s651ii> Luettu 1.2.2024

8. hw_telium_payment_terminal. Verkkoaineisto. <https://odoo-community.org/shop/hardware-telium-payment-terminal-3507> luettu 1.2.2024

9. VF-Integraatio-testaus. PDF-Dokumentti.