PROTOTIP D'UN SISTEMA *CASHLESS* SOBRE EXPENEDORS AUTOMÀTICS EXISTENTS

Treball de Final de Grau
Presentat a

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

per

Petar Hlad Colic

En requeriment dels
estudis del grau en

Ciències i Tecnologies de la Telecomunicació

Tutor: Marcel Fernández Muñoz

Barcelona, Setembre 2016

Resum

Aquest document presenta el disseny i el desenvolupament d'un prototip per a un sistema de pagament sense contacte en expenedors automàtics ja fabricats. L'expenedor serà controlat usant un ordinador de placa reduïda per controlar els circuits que engeguen motors de l'expenedor i comunicar-se amb el servidor central, i un lector NFC per poder llegir targetes dels usuaris.

El projecte es centra en el desenvolupament del codi de les aplicacions de servidor i de client, i en el desplegament de hardware que s'integrarà dins de l'expenedor automàtic.

Resumen

Este documento presenta el diseño y el desarrollo de un prototipo para un sistema de pago sin contaco en máquinas expendedoras ya fabricadas. La máquina expendedora será controlada usando un ordenador de placa reducida para controlar los circuitos que encienden los motores de la máquina expendedora y comunicarse con el servidor central, y un lector NFC para poder leer las targetas de los usuarios.

El proyecto se centra en el desarrollo de codigo de las aplicaciones de servidor y de cliente, y en el despliegue de hardware que se integrará dentro de la máquina expendedora.

Abstract

This document presents the design and development of a prototype for a cashless system on existing vending machines. The machine will be controlled using a small-sized single-board computer to control the circuits that power the motors of the vender and to comunicate with the central server, and an NFC reader to read users' cards.

This projects focuses in the code development of the server and client applications, and also on the hardware deployment that will later integrate in the vending machine.

"Agraïments"
Autor Frase

AGRAÏMENTS

Voldria a en Marcel Fernández per creure en mi i acceptar la meva proposta de projecte.

Als companys que anaven a acompanyar-me en l'aventura de desenvolupar aquest projecte abans de decidir presentar-ho com a treball de fi de grau i en especial a l'equip vell de IT de l'associació per tota l'ajuda que m'han prestat.

I sobretot als meus pares, per tot el seu suport infalible les 24 hores del dia, cada dia.

ÍNDEX

CAPÍTOL 1

INTRODUCCIÓ

Els expenedors automàtics de begudes fredes tenen una vida útil bastant llarga però les compres amb efectiu s'estan quedant enrere.

Cada dia el nombre de transaccions amb efectiu es van reduïnt. Els diners en efectiu ocupen espai i, sobretot les monedes, pesen molt. És per això que les transaccions s'encaminen cap al *cashless* (en anglès, sense efectiu).

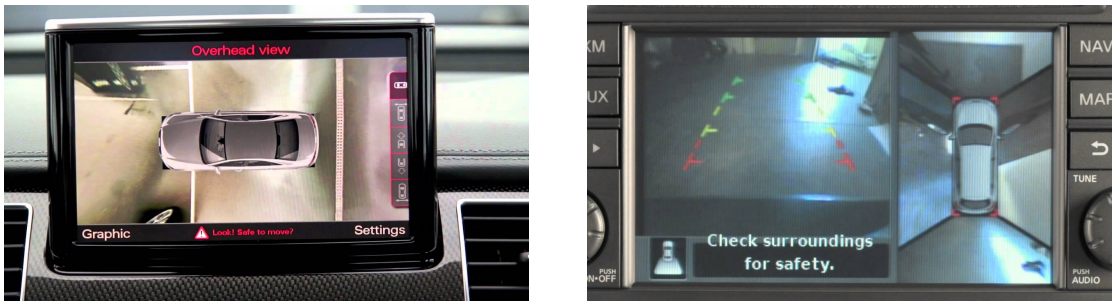
Objectius del projecte

L'objectiu d'aquest projecte és dissenyar i desenvolupar un prototip per a un sistema *cashless* que es pugui integrar dins d'expenedors automàtics existents. El projecte es centrarà en un model específic d'expenedor automàtic de begudes fredes (Dixie-Narco DNCB 386) perquè és l'expenedor que hi ha disponible per al desenvolupament.

Els principals objectius es poden resumir en els següents punts:

Dissenyar i desenvolupar l'aplicació de servidor. El sistema de la solució serà un sistema centralitzat. Tot es gestionarà des d'un mateix lloc, i aquest serà l'aplicació de servidor.

Dissenyar i desenvolupar l'aplicació de client. Per tal de poder controlar l'expenedor automàtic i comunicar-se amb l'aplicació de servidor, és necessaria l'aplicació de client.



(a) Audi S8 – 360 View Camera (b) Nissan Rogue – Around View Monitor

Figura 1.1: Two commercial 360° vision systems display view

Dissenyar Desenvolupar el sistema de hardware. Per tal de que l'aplicació de client pugui controlar l'expenedor, és necessari el sistema de hardware que farà d'interfície entre l'aplicació i els circuits de l'expenedor.

La feina feta durant aquests mesos s'ha centrat en desenvolupar el codi de les aplicacions i en desenvolupar el desplegament de hardware que s'integrarà a l'expenedor.

Planificació del temps

The time plan followed during this project development is shown in Figure 1.2 Gantt Diagram. The developed work has been split in 8 work packages. A detailed description of each work package and the internal tasks developed can be found on Appendix **??**.

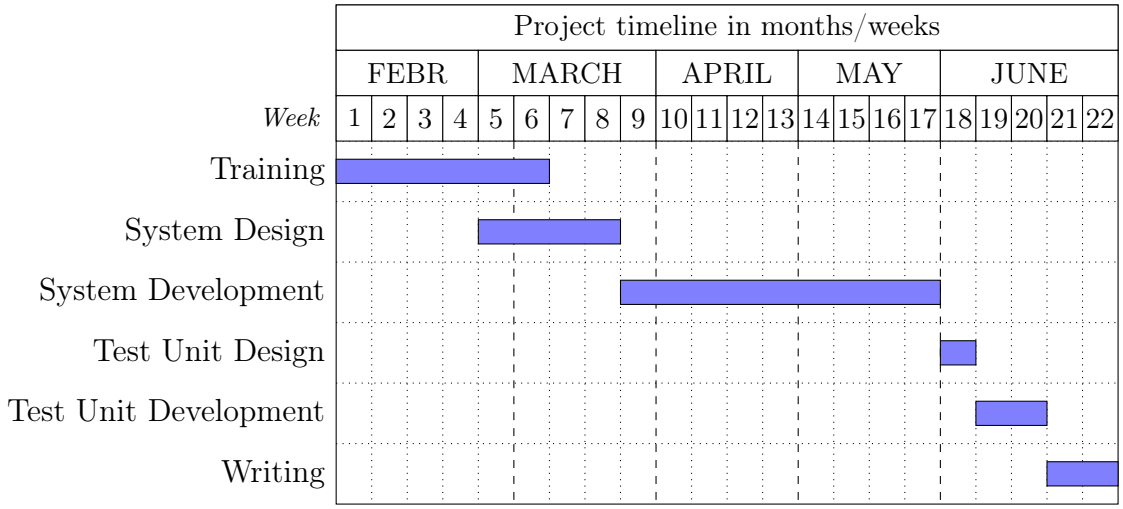


Figura 1.2: Project Gantt Chart

CAPÍTOL 2

REQUERIMENTS

L'objectiu del projecte és dissenyar un sistema que es pugui integrar en expenedors automàtics existents i permeti la compra de begudes sense necessitat de fer ús de diners en efectiu en el moment de la compra.

Requeriments generals de la solució

Millors prestacions La solució no pot empitjorar les prestacions que tenia l'expenedor automàtic en l'estat original. Només n'ha de millorar.

Reversibilitat En cas de necessitat, s'ha de poder retrocedir en la configuració de l'expenedor automàtic i deixar-lo en l'estat original.

Seguretat El sistema ha de ser prou segur per evitar frau o robatoris de diners.

Per assolir aquests requeriments el projecte es dividirà en tres parts:

Aplicació de Servidor És el sistema central de la solució. Serà la part que gestionarà les compres, transaccions monetàries, usuaris.

Aplicació de Client És el sistema que controlarà directament l'expenedor. Quan l'usuari s'identifiqui i faci una sel·lecció, l'aplicació es comunicarà amb el sistema central perquè autoritzi la compra i pugui fer que l'expenedor serveixi una beguda.

Implementació i integració del hardware És tota la part del projecte que tracta la part física del projecte. En altres paraules, la integració del dispositiu que contindrà l'aplicació de client, el/s dispositiu/s que servira/n per identificar l'usuari i tots els circuits elèctrics per controlar l'expenedor i el seu estat.

Un cop dividit el projecte en tres blocs, perquè la solució sigui prou segura contra fraus, es proposa que el sistema sigui un sistema de prepagament on el saldo de cada usuari l'emmagatzema i gestiona el sistema central. També es proposa que l'usuari s'identifiqui mitjançant una etiqueta NFC.

Requeriments específics

Aplicació de Servidor

Per tal de tenir una solució que compleixi els requeriments generals, es proposen una sèrie de requisits específics per a l'aplicació de servidor:

Gestió d'expenedors Ha de tenir control sobre els expenedors. Ha de gestionar quin producte tindrà l'expenedor a cada carril i a quin preu. Ha de poder deshabilitar la venda d'un cert carril. Ha de poder visualitzar l'estat en el que es troba un expenedor (s'ha quedat sense monedes per donar canvi, s'ha avariat, s'ha quedat sense producte ...).

Gestió d'usuaris Ha de poder crear nous usuaris, eliminar-ne, modificar-ne les dades. També ha de poder identificar un usuari a partir d'algun identificador amb què aquest s'identifica quan fa ús d'un expenedor.

Gestió del saldo disponible Ha de control·lar el saldo disponible del que disposa cada usuari. L'ha de poder consultar, modificar-lo quan hi ha un pagament o una recàrrega de saldo.

Gestió de les compres Ha d'autoritzar la compra que fa un usuari tenint en compte el seu saldo disponible i la disponibilitat del producte a l'expenedor.

Seguretat en les transaccions El sistema ha de ser segur en termes de transaccions de diners i rècarregs de saldo per compra. S'han d'evitar suplantacions d'identitat, intrusions en el sistema, fraus i demés posibles accions que perjudiquin al propietari de l'expenedor o als usuaris.

Detecció i gestió d'incidencies En sistemes que involucren parts mecàniques, elèctriques, i parts que es comuniquen per xarxa, no és tan extrany que en certs moments les coses no funcionin com haurien de funcionar. Així el sistema hauria de ser capaç de detectar la gran part d'incidències produides per errors de comunicació o fallades en l'expenedor i solucionar-les en la mesura que sigui possible.

Aplicació de client

De la mateixa manera que amb l'aplicació de servidor, es proposen un seguit de requeriments específics per l'aplicació de client:

Control sobre l'expenedor Ha de poder controlar els circuits de l'expenedor per exendre llaunes. També ha de poder analitzar els paràmetres dels circuits per a poder preveure la quantitat de producte restant que conté l'expenedor i a més detectar i evitar fallades.

Comunicació amb el sistema central S'ha de comunicar amb el sistema central per a recollir la informació necessaria per funcionar i per a poder realitzar les vendes. També ha de comunicar al servidor la informació sobre l'estat de l'expenedor.

Interfície fàcil d'usar L'ús de l'aplicació de cara a l'usuari es farà a través d'una interfície gràfica d'usuari que ha de ser senzilla d'utilitzar de la mateixa manera que ho és un expenedor automàtic usual.

Seguretat en les comunicacions Les comunicacions han de tenir un mecanisme per evitar intents de frau i demés accions perjudicials per al propietari de l'expenedor i altres usuaris.

Implementació del Hardware

Tenint en compte els requisits generals de la solució i els requisits proposats per a l'aplicació de client, es proposen un seguit de requeriments específics per a la implementació del hardware dins l'expenedor:

Reversibilitat de la integració Un cop integrat el sistema, en cas de necessitat, s'ha de poder extreure el hardware i deixar l'expenedor en l'estat original. Això vol dir que no es pot alterar irreversiblement el cablejat de l'expenedor.

Volum reduït No pot ocupar molt espai. Ha de caber dins de la porta de l'expenedor.

Implementació modular Ha de ser prou facil de posar i treure de l'expenedor. Fàcil de reparar i reposar blocs defectuosos.

No interferir amb el funcionament mecànic de l'expenedor La implementació no ha d'interferir en el funcionament mecànic usual que té un expenedor com són l'obertura de l'expenedor per recollir les monedes o reposar producte.

Evitar creuament amb una compra amb monedes Es pot donar el cas que es vulgui efectuar una compra sense efectiu i s'hagin introduït monedes a l'expenedor al mateix temps. Ha d'evitar que es produeixin problemes per aquest creuament.

Aïllament físic amb l'exterior Ha d'estar físicament de l'exterior per protegir el sistema i evitar possibles intents de manipulació.

Interfície d'usuari Ha de tenir una interfície de manera que l'usuari pugui visualitzar la informació que dóna l'aplicació de client i també alguna manera d'introduir la seva selecció.

Lector NFC Ha d'incloure un lector NFC per a la identificació dels usuaris.

CAPÍTOL 3

FUNCIONAMENT D'UN EXPENEDOR AUTOMÀTIC DE BEGUDES FREDES

Per a poder dissenyar la solució i poder entendre les millores, primer s'ha d'esbrinar i entendre el funcionament original d'un expenedor automàtic.

Descripció de les parts

Descripció de les parts.

(Imatge de la màquina).

Cas d'ús

Presentem el cas d'ús usual:

1. Un usuari s'acosta fins l'expenedor i observa quins productes estan disponibles i quin preu tenen. Cada botó de selecció té una etiqueta al costat que indica el producte i el seu preu. La disponibilitat del producte la indica un indicador lluminós sota el botó (si emet llum, no està disponible.
2. Un cop l'usuari ha fet la seva elecció, introdueix monedes amb un valor igual o superior a l'import del producte.
3. L'usuari apreta el botó de selecció del producte que ha escollit.
4. L'expenedor retorna el canvi (si n'hi ha) i serveix la beguda que l'usuari ha seleccionat.

Exposem els casos d'ús quan alguna cosa va com no hauria d'anar:

L'usuari apreta el botó d'un producte que no està disponible: No passa res. És com si no hagués apretat cap botó.

L'usuari apreta el botó d'un producte pel qual no té saldo suficient: No passa res. És com si no hagués apretat cap botó.

L'indicador de ïmport exacte"està encès: L'expenedor només accepta monedes del tipus que pot retornar i no retorna canvi.

Circuit elèctric

L'expenedor automàtic està dissenyat per funcionar amb un sistema electromecànic sense necessitat de circuits electrònics.

El mecanisme consistia en un petit interruptor mecànic amb una palanqueta allargada que quedava premuda quan s'introduïa un tipus de moneda concret i activava la resta del circuit que era controlat per un relé de tres commutadors, els botons de selecció i els interruptors que es troben on els carrils per controlar la posició dels motors.

Avui dia, amb expenedors que retornen el canvi, aquest relé és substituït per la màquina de canvi, que és l'aparell que classifica les monedes segons el seu valor, i després retorna canvi si cal. A més, dona la possibilitat de poder definir diversos preus.

Per tal de tenir una visió més clara del funcionament per tal de poder dissenyar el nostre sistema, presentem un circuit simplificat de l'expenedor juntament amb la màquina de canvi que s'ha extret del manual d'instruccions (llegint les explicacions i analitzant l'esquema elèctric complet).

(Esquema elèctric simplificat)

Quan s'introdueixen les monedes, la màquina de canvi dona corrent als botons pels quals s'ha introduit un import suficient. Un cop premut el botó de selecció, es tanca el circuit del motor i el motor s'acciona. Hi ha mes factors a tenir en compte, però per a dissenyar el nostre sistema ja en tenim suficient.

Cabria comentar que els botons de selecció estan connectats en serie d'aquesta manera en concret per evitar que el circuit el tanqui més d'un botó al mateix temps.

CAPÍTOL 4

DISSENY DE SOFTWARE

En aquesta part es presenta el disseny de l'aplicació amb les decisions de disseny explicades.

Per al projecte sencer s'ha escollit programar en llenguatge *Python* perquè és molt senzill de programar i és molt més llegible.

Aplicació de servidor

Per a l'aplicació de servidor s'ha fet servir un *framework* per a desenvolupar webs en Python anomenat Django.

S'ha seguit el patró MVC (Model-Vista-Controlador).

Model de dades

Per a desenvolupar una aplicació web mínimament complexa, primer s'ha de començar dissenyant el model de dades.

Cada *model* en Django representa una taula a la base de dades.

User Cada usuari està associat a un `User` i contindrà les dades que l'identificaran com el seu nom, correu electrònic, nom d'usuari i contrassenya, DNI ...

Credit Conté el saldo disponible de l'usuari i, per tant, està associat a un `User`, i cada `User` té associat com a molt un `Credit`. Està en una taula separada perquè està pensat perquè l'aplicació pugui ser integrada en una base de dades d'usuaris existent.

Card Representa la targeta amb la què s'identificarà l'usuari. Per tant està associada a un usuari. Igual que les targetes de crèdit, un usuari pot tenir més d'una targeta associada.

Vender Representa un expenedor. Conté la informació bàsica de l'expenedor (model, número de sèrie, localització geogràfica), i també la informació sobre el seu estat.

ApiKeyToken Per a garantir un mínim de seguretat en les comunicacions, l'aplicació de client usarà un codi per a identificar l'expenedor des del qual s'està comunicant. Aquest identificador serà únic per a cada expenedor. `Vender` i `ApiKeyToken` estan en taules separades per raons d'implementació de codi d'autenticació en Django.

Column Representa un carril d'un expenedor. Cada carril té un preu de venda i un tipus de producte. També pot estar actiu o inactiu, i també pot estar buit o ple.

Purchase Cada cop que l'usuari fa una selecció i tot va bé (la informació proporcionada per realitzar la compra és correcta i l'usuari té suficient saldo disponbible) es crea una instància de `Purchase` que representa la compra que ha fet l'usuari amb la informació corresponent (targeta amb què s'ha comprat, producte, carril, preu, data i hora, identificador de la compra, i si ha estat pagada). L'identificador de la compra és un identificador aleatori que assigna l'aplicació de client per raons que comentarem més endavant. Tot i que `Column` ja conté la informació de carril, producte i preu, es guarda per separat ja que al llarg del temps aquests paràmetres podrien canviar.

Payment Quan un `Purchase` ha estat creat, s'ha de procedir al pagament de la compra, per tant un pagament està associat directament a una compra. `Payment` representa l'acció de pagar la compra. Inclou el preu de la compra, la quantitat descomptada (si es fa descompte) i el preu final a pagar. També inclou l'usuari a qui se li ha de cobrar, el dia i hora i la informació per determinar si el pagament està pagat i és valid, o si l'han hagut de reemborsar.

RefundPetition Quan un procés de compra es queda a mitges, o l'expenedor pateix una fallada tècnica, o per qualsevol altre motiu és necessari reemborsar l'import d'una compra a l'usuari, primer s'ha de crear una petició de reemborsament. És així perquè d'aquesta manera les peticions es poden generar de manera automàtica mentre que l'autorització de la petició es realitza amb més cura (un sistema més curós o bé amb supervisió humana). Conté l'identificador del pagament que es vol reemborsar, el sol·licitant (ja sigui un empleat o bé un expenedor de manera automàtica), la raó de la petició, l'hora i la data, i la informació per determinar si ha estat denegada o finalment s'ha realitzat el reemborsament.

Refund Representa el reemborsament d'un pagament un cop la petició és acceptada. Així doncs conté l'entitat que ha autoritzat el reemborsament, la petició que ha estat autoritzada i l'hora i la data de l'autorització.

Controlador

El controlador és on es troba tota la lògica en si del sistema.

En Django, quan es desenvolupa una aplicació de servidor que respon a peticions, la funció de controllador ho desenvolupen les classes que de l'API.

Les següents són les classes que fan de control·lador i cadascuna respon a un tipus diferent de petició:

GetUserDataFromCard Retorna la informació de l'usuari (nom i saldo disponible) a partir de l'identificador de la seva targeta.

GetVenderColumns Retorna la informació de cada carril de l'expenedor que l'ha demanat (tots els paràmetres que conté el model `Column`).

ProductPurchase Quan l'usuari ha fet la selecció del producte, es comprova que el producte estigui disponible, que el producte del carril seleccionat coincideixi amb el producte que conté, que el preu també coincideixi i que el saldo disponible de l'usuari sigui suficient. Si tot és correcte, es crea una instància de `Purchase` i el seu `Payment` associat i se li descompta l'import de la compra al saldo de l'usuari. Finalment se li retorna una resposta de que tot ha anat bé a l'expenedor perquè serveixi una beguda o bé se li retorna una resposta d'error informant de què és el que ha anat malament.

RequestCancelPayment Quan alguna cosa ha anat malament a la banda de l'expenedor (s'ha encallat el mecanisme, ha fallat la xarxa, ...) pot ser que s'hagi cobrat la compra a l'usuari i no s'hagi servit la beguda. És per això que cal un mecanisme per demanar reemborsament sota certes circumstàncies. Quan es rep una sol·licitud de reemborsament aquesta s'emmagatzema si fins el moment no existia cap altra sol·licitud associada al mateix pagament.

Vista

La vista en si és la informació que es pot visualitzar de la resposta d'una petició. En la nostra aplicació hi ha dos possibilitats d'origen de petició: des de l'aplicació de client, i des de la interfície d'administració.

Django té per defecte implementada una interfície d'administració per gestionar manualment les dades de l'aplicació. Amb això ens estalviem haver de desenvolupar aplicacions d'intefície grafica d'usuari per tal de poder gestionar nous usuaris, incidencies, etcètera.

Aleshores, la interfície d'administració és un tipus de vista, i la informació que retorna el controlador a l'aplicació de client és l'altre tipus de vista de l'aplicació.

Aplicació de client

Per a l'aplicació de client, com ha de tenir una interfície gràfica d'usuari, s'ha decidit utilitzar `PyQt4` que és una llibreria gràfica. Té integrada la part gràfica i també la comunicació entre fils d'execució.

L'aplicació de client es divideix en 6 grans blocs diferenciables:

comms S'encarrega d'implementar la comunicació amb el servidor.

card_reader S'encarrega de controlar el lector de targetes. Cada cop que detecta una targeta avisa al fil d'execució principal juntament amb l'identificador de la targeta.

column_controller S'encarrega de controlar els mecanismes dels carrils de l'expenedor i de detectar les possibles fallades mecàniques.

db_controller S'encarrega de controlar la base de dades local. En aquesta base de dades s'emmagatzema la informació dels carrils i també la informació de compres fallides per les quals el servidor encara no ha confirmat la creació d'una petició de reemborsament.

graphic_ui : S'encarrega de crear la interfície gràfica d'usuari i de gestionar totes les accions de l'usuari i derivar-les cap el fil d'execució principal.

cocacolero : S'encarrega d'instanciar, controlar i coordinar la resta de blocs. Aquest bloc es troba en el fil d'execució principal

Cas d'ús

A continuació s'exposa l'explicació del cas d'ús usual de l'aplicació per part de l'usuari.

1. L'usuari arriba i apropa la seva targeta al lector.
2. L'usuari és correctament identificat i el seu nom i el seu saldo disponible apareixen per pantalla. Els botons de selecció dels productes disponibles queden habilitats.
3. L'usuari fa la selecció pitjant amb el dit sobre el botó del producte que ha escollit
4. L'usuari disposa de suficient saldo disponible i la compra s'efectua amb èxit. L'expenedor serveix la beguda, es deshabiliten els botons i finalitza el procés.

Exposem el mateix cas d'ús des del punt de vista de l'aplicació.

1. `card_reader` detecta una targeta i comunica l'identificador a `cocacolero`.
2. `cocacolero` demana la informació de l'usuari associat a la targeta detectada a l'aplicació del servidor. El servidor l'identifica amb èxit i retorna la informació demanada (nom i saldo). `cocacolero` habilita els botons de `graphic_ui` dels productes disponibles.
3. Quan l'usuari fa la selecció, `graphic_ui` comunica a `cocacolero` la selecció.
4. `cocacolero` genera un identificador aleatori per a la compra, envia al servidor la petició de la compra. El servidor comprova que la informació és correcta i que l'usuari té suficient saldo. El servidor respon amb èxit. `cocacolero` comunica a `column_controller` el carril que s'ha d'activar i aquest li respon amb èxit quan la beguda s'hagi servit. `cocacolero` deshabilita els botons de `graphic_ui` i finalitza el procés.

Ara exposem els casos d'ús quan alguna cosa no funciona com està previst:

L'usuari no vol realitzar cap compra després d'identificar-se: L'usuari pitja el botó `Log Out` i finalitza el procés.

L'usuari no s'ha identificat amb èxit: S'ignora

Es detecta una targeta durant un procés de compra: S'ignora

L'usuari fa una selecció per la qual no disposa de saldo suficient: el servidor ho detecta i ho comunica a l'aplicació de client. L'aplicació descarta la compra i finalitza el procés.

La informació de la compra és incorrecta: El mateix cas que el de saldo insuficient.

El servidor no pot processar la compra per qualsevol altre motiu: El mateix cas que el de saldo insuficient.

El servidor no respon dins del termini establert: S'anul·la el procés. S'emmagatzema la informació de la compra i s'envien peticions al servidor periodicament fins que el servidor crea la petició de reemborsament amb èxit o la descarta per qualsevol motiu.

L'aplicació detecta una fallada tècnica i no s'ha servit el producte: El mateix cas que quan es perd la resposta del servidor.

CAPÍTOL 5

DISSENY DE HARDWARE

En aquest capítol comentarem tot el disseny del hardware que s'ha seguit per poder controlar l'expenedor automàtic assolint els requisits definits anteriorment.

Disseny

Com a controlador central de l'expenedor es farà servir un ordinador de placa petita *RaspberryPi 2 B* (a partir d'ara, RPi), ja que disposa de bastants pins d'entrada/sortida, se li pot instal·lar un sistema operatiu basat en linux (que fa molt més fàcil el disseny de l'aplicació de client), té connexió a internet, possibilitat de connectar-hi una pantalla per HDMI o bé una pantalla tàctil.

Per a la interfície d'usuari usarem la pantalla tàctil *Raspberry Pi Touch 7″*.

Per a poder accionar els circuits que accionen els motors de l'expenedor des de la RPi farem servir una placa de relés activats per nivell baix.

RPi necessitarà una alimentació de 5V i 1A. La pantallà tàctil necessita una alimentació de 5V i 1A. La placa de relés necessita una alimentació de 12V.

CAPÍTOL 6

RESULTATS

En aquest capítol exposem els resultats del disseny i el desenvolupament del sistema sencer.

Aplicació de servidor

L'aplicació de servidor és la que més s'ha trigat en dissenyar i desenvolupar perquè el disseny de l'aplicació de client depen directament de la de servidor.

L'aplicació funciona com s'esperava.

A través de la interfície d'administració es poden fer fàcilment les gestions necessàries com són: afegir usuaris nous, afegir expenedors nous, gestionar els carrils dels expenedors ..., tot el que es requeria de poder gestionar en remot.

A la intefície d'administració només hi poden accedir els usuaris que tinguín permís d'administració, i això es pot gestionar des de la mateixa interfície.

(Imatges de la interfície d'administració)

Aplicació de client

L'aplicació de client compleix els requeriments que s'han especificat.

No obstant, té un problema amb el lector de targetes: si l'aplicació es tanca de manera inesperada (per exemple, degut a un error d'execució inesperat), es perd la comunicació amb el lector, i no es pot recuperar fins que no se li fa un reset de l'alimentació al lector. Això és un problema de la llibreria que utilitzem per controlar-lo (nfcpy), ja que amb altres llibreries que s'han provat això no passava.

La interfície gràfica d'usuari no ha estat dissenyada per un dissenyador

Sistema de hardware

El hardware en conjunt està desenvolupat. No obstant, per temes de calendari de l'associació (durant el mes de març l'expenedor està inaccessible, durant l'agost i part de setembre el despatx està tancat), no s'han pogut fer totes les proves que es volia fer.

Durant el quadrimestre s'han fet petites proves de concepte bàsiques amb èxit:

1. Provar d'expendre una llauna amb un relé activant-lo manualment.
2. Provar d'expendre una llauna amb els relés activats per l'aplicació de client.

CAPÍTOL 7

PRESSUPOST

This TFG project has been developed jointly with the GPI team. The budget presented is divided in Activities. Each Activity corresponds to an specific task, and the division has been done according to the equipment and staff needed to develop each tasks. In this budget are considered not only the activities developed in this project scope, but also the activities that are required to perform this project development.

The Activities that had been taken into account are:

Activity 1: Recording I. Captures done using cones as a pattern.

Activity 2: Recording II. Captures done using a chessboard as a pattern.

Activity 3: Undistortion. Test the different undistortion algorithms.

Activity 4: Registration, warping and blending. This Activity includes all the algorithms and methods tested in this TFG project scope and are directly related to the Arcol project.

Activity 5: Documentation. Additional activity that includes all the tasks done outside the Arcol project scope -TFG writing, additional tests, etc.-.

Each Activity summarized cost is the following:

	Activity	Equipment	Workers	Subtotal
1	Cone recording	197.06 €	290.00 €	487.06 €
2	Chessboard recording	205.83 €	290.00 €	495.83 €
3	Undistortion	46.70 €	1,030.00 €	1,076.70 €
4	Registration. warping and blending	442.00 €	21,000.00 €	21,442.00 €
5	Documentation	66.00 €	3,800.00 €	3,866.00 €
			TOTAL	**27.367,59€**

Each activity detailed costs can be found in Appendix **??**. This Appendix also contains each equipment detailed amortization and each worker salary.

CAPÍTOL 8

CONCLUSIONS I TASQUES FUTURES

This project has described the procedure followed to obtain an specific image stitching algorithm. This algorithm had to merge the image from four cameras located around a bus in a single 360°view image. The work described in this project has been done in a multidisciplinary UPC team working in a project commissioned by the Arcol company.

First of all, all the requirements and specifications inherited from the Arcol project have been stated. In basis on these requirements, a research in the current stitching methods has been done, including both commercial systems and high-level algorithms.

The next part included in this project have shown the used methods in the steps inside this TFG scope. From the four different stitching steps –undistortion, registration, warping and blending–, only warping have been fully developed in this project. Table 8.1 shows each step internal tasks and its final status.

Stitching step	*Internal task*	*Status*
Undistortion	Search and test undistortion algorithms	Done by another team member
	Apply the final algorithm	Done by the project author
Registering	Automatic registering	Not done
	Manual registering	Done by the project author
Warping	Search and test warping algorithms	Done by the project author
	Apply the final algorithm	Done by the project author
Blending	Apply provisional algorithm	Done by the project author
	Search and test blending algorithm	Not done
	Apply the final algorithm	Not done
Unghosting	Segment above ground objects	Not done
	Define an algorithm	Not done
	Apply the algorithm to the stitching	Not done

Taula 8.1: Internal tasks status

Finally, the last part of this project shows the results obtained in the warping process. It can be extracted from the results that the developed algorithm beats the state-of-the-art algorithms for this specific application. As it can be seen in Chapter **??**, the *Homography merging* algorithm developed in this project obtains better results than simple homography in this specific application.

Regarding to the initial goals of this project, the results obtained in each point can be summed up as follows:

Representative points automatic detection. The automatic key points detection has been left outside this project scope, and has been left to a future development.

Estimate the warping parameters. The warping parameters and warping algorithm had been estimated successfully.

Blending the results on the final stitching. Blending algorithms have been tested and the feathering selected offers an acceptable performance for the project.

Follow the requirements stated by the Arcol project. Regarding the Arcol specifications, this project is fully adapted to be used with the specific hardware and software. Moreover, the land lines discontinuity has been fully fulfilled. Exposure compensation and avoiding host images has been left to the Arcol project future development.

The overall project assessment is that, although not completely fulfilling all the goals stated at this project beginning, the results obtained in the developed parts are fully satisfactory. At the project start, the warping process has stated to be a trivial part, by simply applying the homography method. However, the development has arose many issues a priori not taken into account. These difficulties had recommended a change in the project direction, focusing on obtaining a more precise warping algorithm. This development sprung good results –and many discarded algorithms– and establish a base-line for the future development.

This project had left the development in a solid stage, with several work strands to be followed. First of all, an automatic registration algorithm can be implemented on the final product. This algorithm should automatically detect the chessboard corners and automatically do the key points registration. Another opened work strand is the definitive blending algorithm, directly bounded up with the above ground object treatment. In this part, an algorithm has to be defined to manage this objects and obtain the minimum image distortion.