INDEX

1. INTRODUCCIÓ 3
1.1 OBJECTIUS DEL PROJECTE 4

2. PSI:PLANIFICACIÓ DEL SISTEMA D'INFORMACIÓ 6
2.1 INICI DEL PLA DE SISTEMES D'INFORMACIÓ 7
2.1.1 Anàlisi de la Necessitat del Sistema 7
2.2 DEFINICIÓ I ORGANITZACIÓ DEL PSI 8
2.2.1 Especificació de l'àmbit d'abast 8
2.2.2 Organització del Pla de Treball 9
2.3 IDENTIFICACIÓ DE REQUISITS 11
2.3.1 Anàlisi de les Necessitats de informació 11
2.3.2 Requisits del sistema 11
2.4 DEFINICIÓ DE L'ARQUITECTURA TECNOLÒGICA 12
ESTRATÈGIA TÍPICA 13
ESTRATÈGIA RECOMANADA 14

3. ASI:ANÀLISI DEL SISTEMA D'INFORMACIÓ 16
3.1 MODEL DE DADES 17
3.1.1 Model Entitat/Relació 17
3.1.2 Taules del model de dades 18
3.2 ANÀLISI DE CASOS D'ÚS 22
3.2.1 Objectius i Notacions dels Casos d'ús 22
3.2.2 Escenari principal 23
3.2.3 Diagrama de cas d'ús amb detall de cuiners 25
3.2.4 Diagrama de cas d'ús amb detall de cambrers 28
3.2.5 Diagrama de cas d'ús amb detall de Cap de personal 37
3.3 ANÀLISI DE CLASSES 48
3.3.1 Vista general dels packages utilitzats 49
3.3.2 Diagrama de classes de dades 51
3.3.3 Diagrama de classes de control 54
3.3.4 Diagrama de classes de Gestor 57
3.3.5 Diagrama de classes de Aplicació 60
3.4 PROGRAMARI, TÉCNIQUES I LLENGUATGES UTILITZATS 64
3.5 ESTRUCTURA I PROTOTIPUS DE INTERFÍCIES 67
3.5.1 Interfícies principals 67
3.5.2 Alguns prototipus d'interfície 69

4. DSI:DISSENY DEL SISTEMA D'INFORMACIÓ 71

4.1 PATRONS DE DISSENY 72
4.1.1 Patrons GRASP 72
4.1.2 Patrons de disseny GOF 75
4.1.2.1 Patrons de disseny de creació utilitzats 76
4.1.2.2 Patrons de disseny estructurals utilitzats 78
4.1.2.3 Patrons de disseny de comportament utilitzats 79
4.2 INTERFÍCIES D'USUARI 81
4.2.1 Interfície de l' apartat d'administració 83
4.2.2 Interfície del frontal de vendes per pantalla tàctil 86
4.2.3 Interfície del frontal de vendes per PDA 89
4.2.4 Interfície de les terminals de recepció de comanda 91
4.3 FUTURES MILLORES 93

5. CONCLUSIONS 94

6. BIBLIOGRAFIA 98

7. ANNEX 100

ANNEX 1: NORMES DE DOCUMENTACIÓ 101

ANNEX 2: CONTINGUT DEL CD-ROM 102

1. INTRODUCCIÓ

1.1 OBJECTIUS DEL PROJECTE

Introducció:

La creixent tecnologia informàtica portàtil ha permès facilitar la vida en el mon laboral en molts àmbits (oficines, autovendes, magatzem, ...).

Existeixen, des de telèfons mòbils amb moltes funcionalitats informàtiques com agenda electrònica o possibilitat de fer conferències en directe, fins a petits ordenadors que caben en el palmell de la mà anomenats PDA. Aquests son els més utilitzats avui en dia per demanar comandes en el mon de l'hostaleria i transport de productes en general. També els podem utilitzar com si fos un bloc de notes de paper o fins i tot, els podem utilitzar com a GPS o per enviar i rebre correu, entre moltes altres funcionalitats.

Aquesta evolució tecnològica a aportat molts avenços sobretot cares als negocis i a la obtenció de la màxima comoditat i rendiment del mateix. Això ha provocat un ràpid desenvolupament de software per aquests dispositius portàtils.

Tot això pot ésser totalment aprofitable en el mon de la restauració. Sobretot en restaurants de gran envergadura amb terrasses lluny de les barres i llocs de preparació dels productes els cambrers on han de realitzar varis viatges abans de servir les taules. En aquests casos un dispositiu portàtil com una PDA seria de gran ajuda a la hora de recollir les comandes , optimitzant així el temps de resposta.

Necessitat d'informatització

Atenent al ràpid creixement d'aquestes tecnologies portàtils, creix també la necessitat de informatitzar els negocis per tal de guanyar temps, eficàcia i diners.

El fet de poder utilitzar aquests dispositius a l'hora de realitzar moltes tasques que fins ara fèiem en paper ens proporciona nombrosos avantatges, com per exemple: la realització de factures, albarans i assentaments al instant sense haver de fer la feina dues vegades, el control d'estoc dels magatzems o fins i tot, poder fer comandes automàtiques segons les ventes produïdes.

Però centrant-nos en l'àmbit de restauració, a part de poder realitzar tot el mencionat anteriorment, aquesta tecnologia ens permet donar un servei òptim als clients, garantint al mateix temps seguretat i elegància cara al públic, recollint les comandes al moment i enviant-les als respectius llocs d'elaboració al mateix instant.

A més, el fet d'informatitzar un restaurant proporciona un màxim rendiment a la economia del negoci, evitant els viatges de cambrers que van a recollir les comandes per dur-les als llocs d'elaboració, i fins i tot pot generar més quantitat d'ingressos degut a l'optimització de les feines degudes a les PDA.

Objectius

El projecte consisteix en un sistema informatitzat per restaurants centralitzat en un servidor que permet recollir comandes als clients a través de PDA's i de pantalles tàctils situades a la barra per enviar-les als corresponents llocs d' elaboració:

-Terminal de la cuina on s'elaboraran tots els menjars.
-Terminal de la bodega on es serviran els vins i caves.
-Terminal barra on es serviran els licors, refrescos (entre d'altres productes totalment configurables).
-Terminal de postres on es reben les línies de comanda relacionades amb els postres.

L'objectiu acadèmic del projecte serà adquirir coneixement a nivell de disseny i gestió de dades, així com la comunicació de varis dispositius en temps real. S'utilitzarà la tecnologia .net, concretament el llenguatge de programació visual c# .net. i com a gestor de bases de dades el mysql 5.0.19. A més, treballaré amb la metodologia UML, utilitzant també varis patrons.

Per fer del sistema un sistema centralitzat en un servidor s'utilitzarà la comunicació per terminal service de windows el qual ens brinda la possibilitat d'establir varies sessions d'usuaris concurrents. Cada sessió estarà destinada a un cambrer, per tant, hi haurà una sessió per PDA.

També destacar que el que farà d'aquest projecte un projecte diferent i sobretot eficient és la utilització de un motor de persistència anomenat nhibernate utilitzat per comunicar les dades del model relacional de base de dades amb els objectes de la programació orientada a objectes.

2. PSI:PLANIFICACIÓ DEL SISTEMA D'INFORMACIÓ

2.1 INICI DEL PLA DE SISTEMES D'INFORMACIÓ

2.1.1 Anàlisi de la Necessitat del Sistema

Els gran restaurants amb més de una planta i amb una o varies terrasses a fora perden molt de temps abans de servir la comanda a la taula.

Davant d'aquest problema que pot afavorir a donar mala imatge al servei de l'establiment es planteja una manera per solucionar-ho. Es pretén recórrer a la informàtica per servir als clients de manera ràpida , elegant i eficient.

Per això ja fa temps que existeixen sistemes informàtics orientats als restaurants. Molts d'aquests consten de una tpv (terminal punt de venda) en cada apartament del restaurant. Cada cambrer realitza la comanda sobre paper i posteriorment introdueix la comanda a la tpv més propera.

Segons la meva opinió, aquesta manera de treballar no és òptima. En més d'una ocasió hi ha varis cambrers esperant per introduir la comanda al a tpv. A més, estan realitzant la feina dues vegades. Una per recollir-la en paper, i l'altre per entrar-la al ordenador. En aquests dos passos es poden produir confusions o errors gramaticals i mal interpretar el que ens ha demanat el client.

Hi ha més alternatives que la anterior. Hi ha restaurants que tenen sistemes informàtics on realitzen les comandes de l'exterior amb ordenadors portàtils, les PDA. Però la majoria d'aquests establiments han d'extreure les dades de la PDA connectant-la a un ordenador. Això fa perdre temps i ben mirat és més ràpid recollir les comandes amb paper de la manera tradicional.

La idea del meu projecte sorgeix de la necessitat de millorar els sistemes comentats anteriorment. El meu projecte serà un sistema centralitzat en un servidor que permetrà recollir comandes des de les PDA's o tpv i mitjançant la tecnologia wifi enviar-la en temps real als respectius llocs d'elaboració. Així dons les terminals de recepció de les comandes son visualitzades pels cuiners. Finalment aquests tenen la possibilitat de marcar en la pantalla tàctil cada moment els productes ja preparats perquè cada cambrer sàpiga en tot moment l'estat de la comanda.

2.2 DEFINICIÓ I ORGANITZACIÓ DEL PSI

2.2.1 Especificació de l'àmbit d'abast

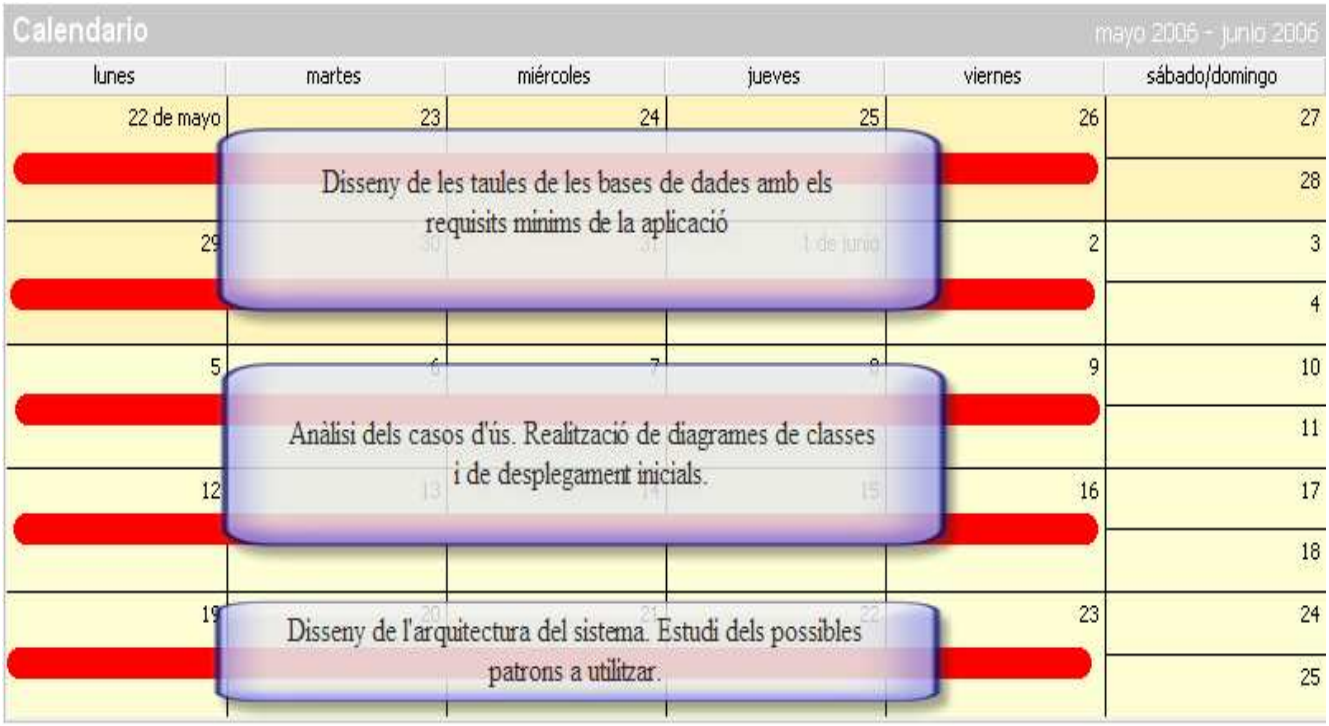
Aquest sistema és orientat a tot tipus de restaurant. Evidentment se'n pot treure molt més profit es un restaurant de gran envergadura.

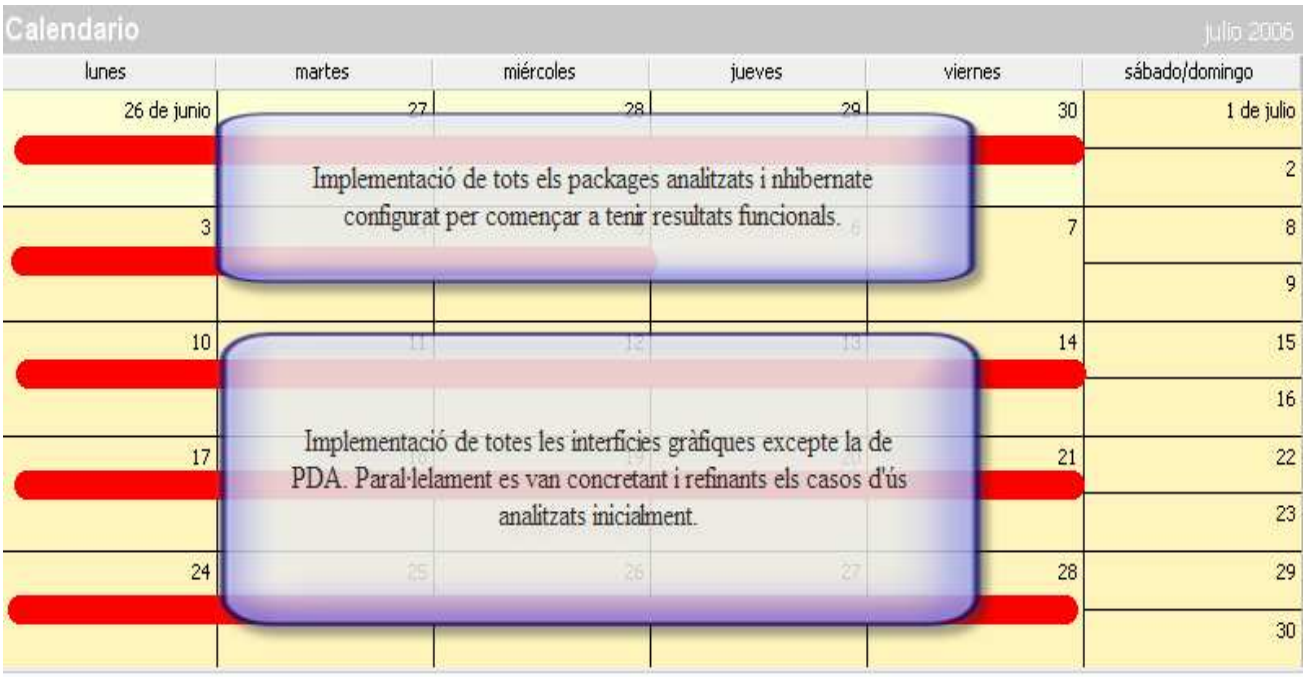
Gràcies a les opcions de configuració de terminals podem decidir quantes utilitzar. Per tant es un sistema totalment adaptable a tots els pressupostos.

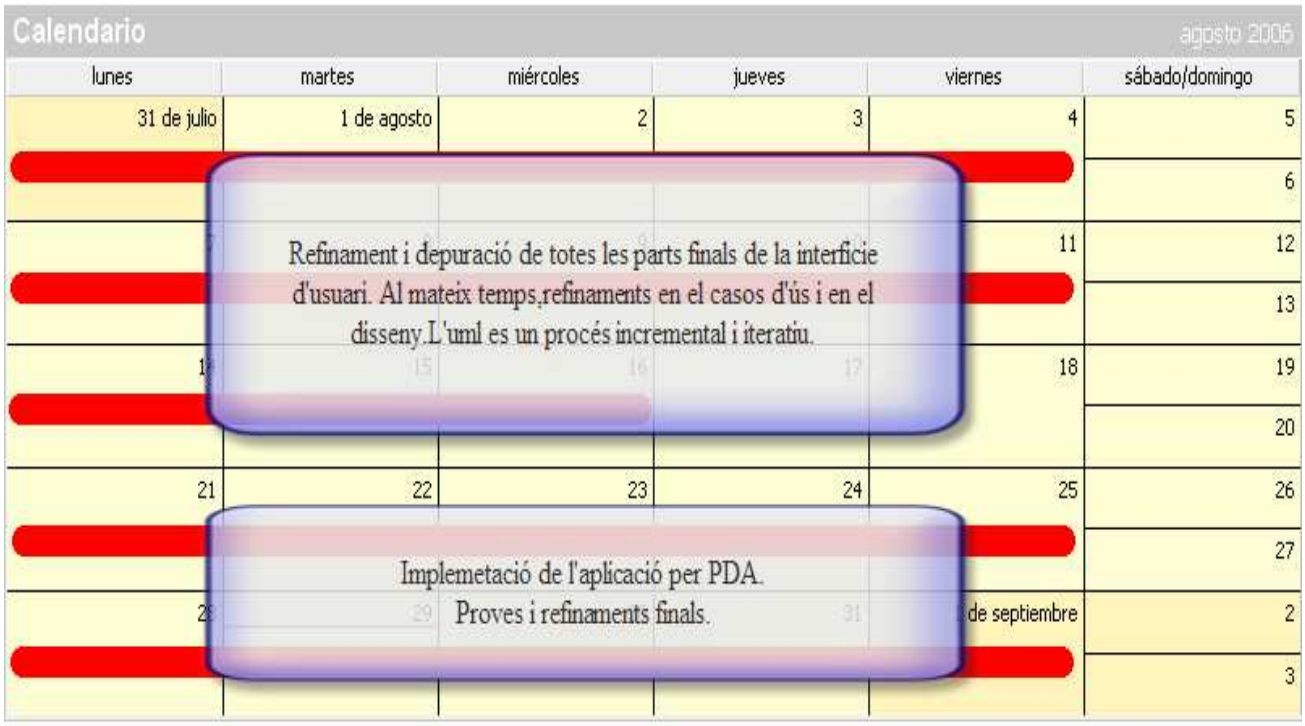
També permet la opció de treballar amb PDA opcionalment en casos en que el pressupost sigui més elevat.

2.2.2 Organització del Pla de Treball

A continuació, el calendari mostrant amb detall tota la organització.







La documentació s'ha anat fent a mida que el projecte ha anat evolucionant.

2.3 IDENTIFICACIÓ DE REQUISITS

2.3.1 Anàlisi de les Necessitats de informació

Es presenten les principals entitats utilitzades en l' aplicació. És veuran les restants i molt més detallades en l'apartat d'anàlisi:

Entitat Producte: Aquí es guarda la informació de cada producte, així com el preu, disponibilitat..i el més important, de quin grup de producte és.

Entitat Cambrer: Aquesta guarda les dades del personal i també el seu loggin i password per accedir a l'aplicació

Entitat Pagament: Aquí tenim la informació de l'estat de les comandes. Si han estat pagades apareixeran en aquesta entitat, altrament seran cobros pendents.

Entitat comanda: Informació de l'estat de la comanda, preu,etc. L'estat de la comanda és la base d'aquest projecte, ja que aquesta informació es pot canviar en les pantalles tàctils per informar en cada moment de la situació de cada plat.

Aquestes entitats són les principals, però n'hi ha d'altres que veurem més endavant en l'anàlisi del sistema.

2.3.2 Requisits del sistema

A continuació es presenten els requisits fonamentals per poder arrancar l'aplicació en el servidor.

- Per preparar el servidor principal es necessita tenir instal·lat el sistema gestor de base de dades mysql 5.0.

- Es necessari tenir instal·lada la última versió del framework .net v2.0, ja que com he comentat en els objectius, aquest projecte està realitzat integrament amb tecnologia .net.

- Serà necessari tenir el connector entre mysql i el .net anomenat mysql.connector.net. Es pot descarregar gratuïtament de l'apartat de descarregues de la pàgina oficial de mysql, http://www.mysql.com .

- Al treballar amb sessions concurrents al servidor en un windows XP, és necessari tenir comprades tantes llicencies de terminal service com PDA's s'hi connectin. Altrament existeix un patch per aquest sistema operatiu que aprofita una funcionalitat que ens oferia la versió beta del win XP que permet tenir tantes sessions

de terminal com vulguem, legalment i el millor, gratuïtament.

2.4 DEFINICIÓ DE L'ARQUITECTURA TECNOLÒGICA

A continuació es mostra el desplegament del hardware i l'arquitectura utilitzada respectivament. Es mostra també la importància de definir una estratègia de persistència en la nostra arquitectura basada amb nhibernate. En una aplicació estàndard més del 50% del codi generat esta relacionat amb aquesta lògica .

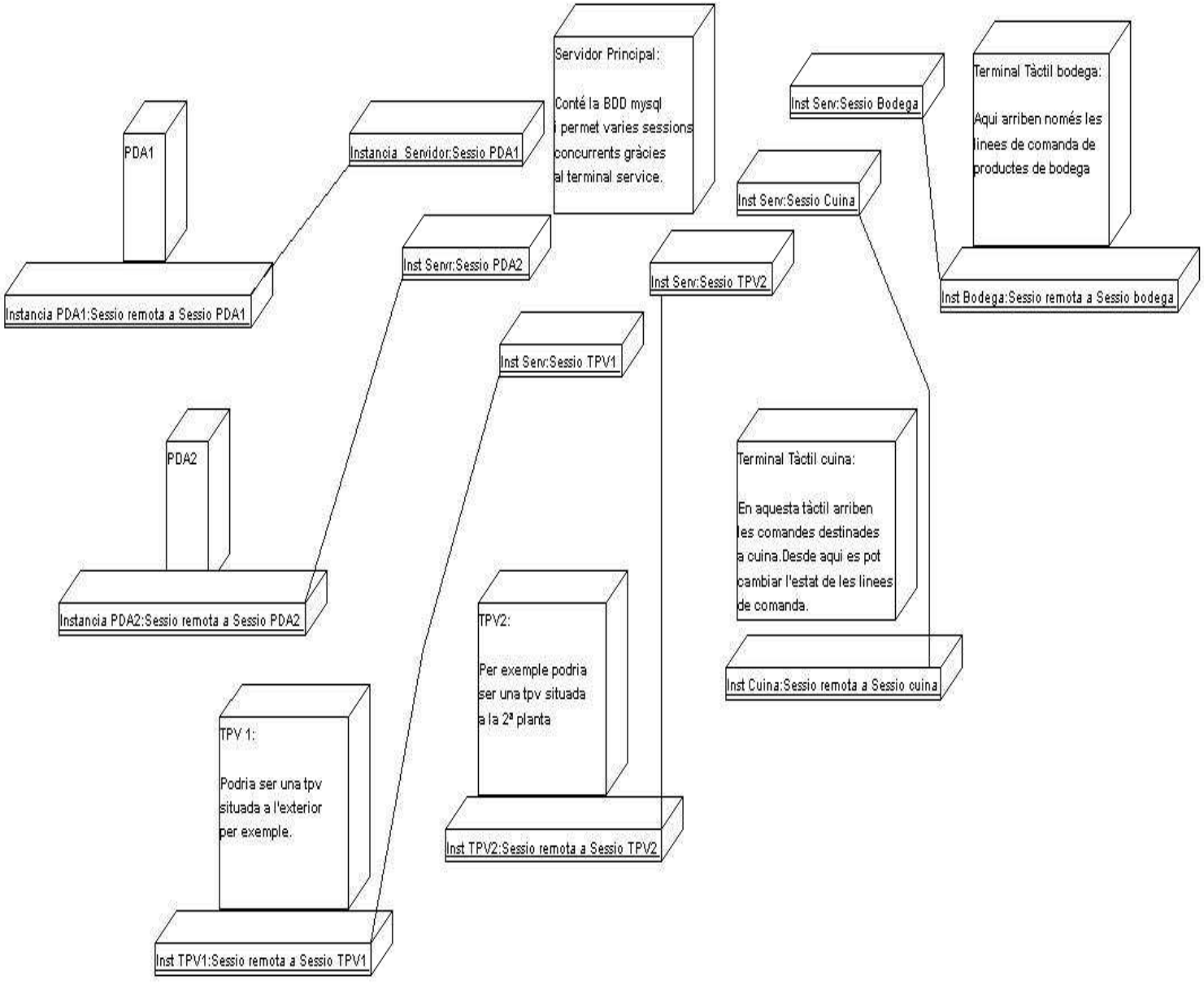


Fig 1: Desplegament del hardware

Com podem comprovar el servidor serveix totes les sessions per cada client. També podríem descarregar-li la feina utilitzant altres màquines que facilitin sessions de terminal.

NHibernate, un framework Object-Relational-Mapping

NHibernate resol de forma automàtica la persistència dels objectes de domini .NET.

La lógica de persistència avarca tot el codi que meva aplicació requereix per poder gravar i recuperar la informació inherent al seu domini i hi ha varies estratègies per dur a terme aquestes accions.

Estratègia típica

Per el desenvolupador de .NET, la estratègia més típica es utilitzar directament ADO.NET. Amb aquesta estratègia, la major part del codi escrit se centra en recuperar un snapshot de la base de dades en un DataSet, modificar eventualment el DataSet en memòria, per posteriorment, a través del DataAdapter corresponent, aplicar els canvis contra la base de dades subjacent. Tot i que els DataSets funcionen correctament, des de el punt de vista de la orientació a objectes evidencien les següents desavantatges:

- Els DataSets representen informació tabular, no representen objectes del meu domini.
- Els DataSets representen relacions entre taules, no representen els diferents tipus de associacions que sorgeixen entre els objectes del meu domini.
- Els DataSets son extremadament sensibles als canvis que puguin sorgir en el esquema de la base de dades.
- El tipus de codi generat per manipular els DataSets tendeix a ser repetitiu i relativament difícil de mantenir.

Estratègia recomanada

Una estratègia més elegant i compatible amb las bones pràctiques de disseny en els últims 10 anys, es la de dissenyar un model de objectes del domini que representi el 100% de la informació que manipula l'aplicació i utilitzar un framework de Object Relational Mapping (ORM) que resolgui de forma transparent la persistència de aquests objectes contra una base de dades relacional. Utilitzar un framework ORM ofereix entre altres els següents avantatges:

- Persistència transparent: Els objectes del domini no saben res sobre la base de dades on son persistits, el framework ho resolt de forma automàtica utilitzant fitxers de mapping expressats en XML.

- Suport de polimorfisme: Es poden carregar jerarquies de objectes ,de forma polimòrfica.

- Suport dels 3 nivells de mapeig de herència: Es pot mapejar tota una jerarquia de classes a una sola taula, crear una taula per cada classe concreta o crear una taula per cada escaló de la jerarquia.

- Suport complet d' associacions: Els frameworks de ORM suporten el mapeig de tots els tipus de relacions que poden existir en un model d'objectes del domini (associacions 1..1, 1...N, N..M, unidireccionals i bidireccionals).

- Suport de carrega de objectes Proxy: Es poden carregar objectes que només continguin la clau del objecte complet.

- Suport de caching: En el context de una transacció, puc disminuir la quantitat de vegades que vaig contra la base de dades cachejant en memòria els objectes que son accedits vàries vegades.

- Suport de múltiples dialectes SQL: Puc independitzar-me completament del tipus de base de dades utilitzada. La meva aplicació pot persistir les seves dades en SQL Server, en Oracle, en MySQL, etc. simplement canviant la configuració corresponent en un XML.

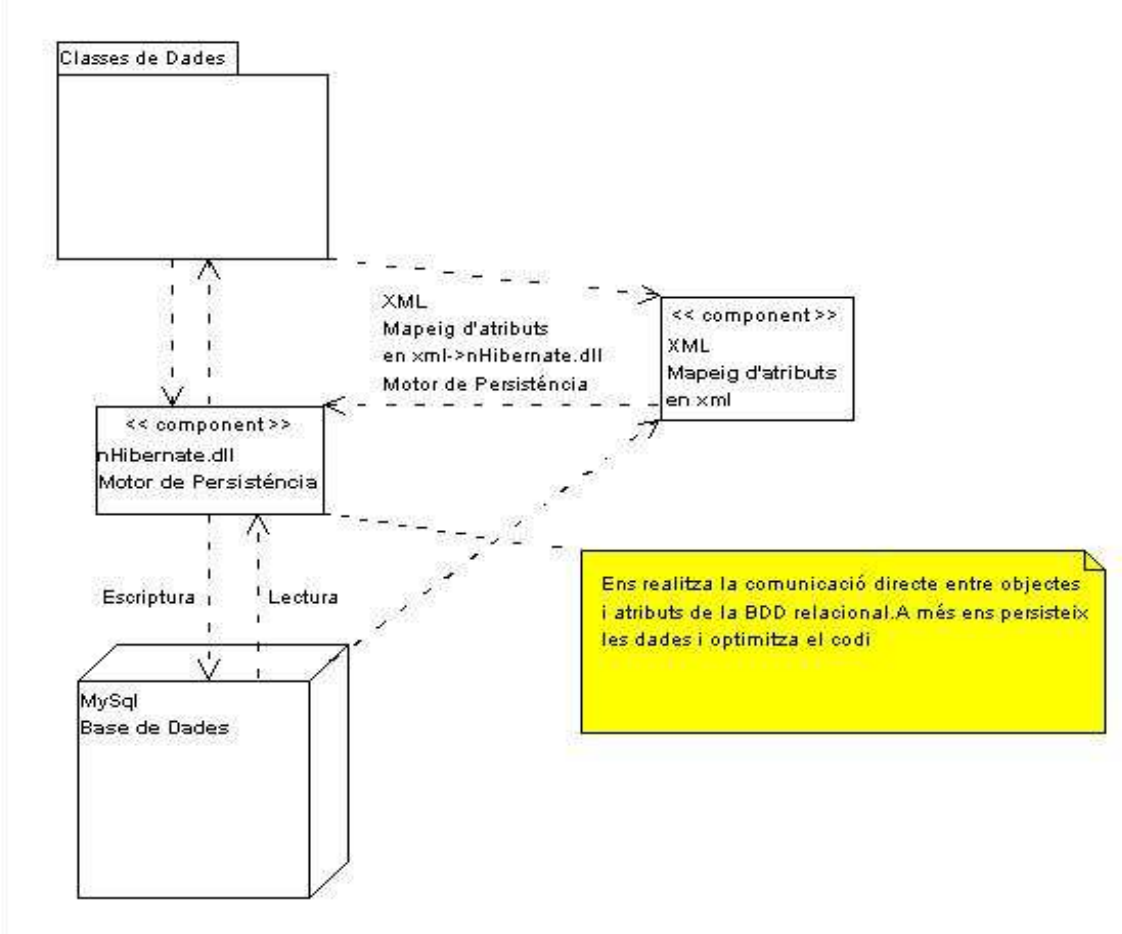


Fig 2:Diagrama de components

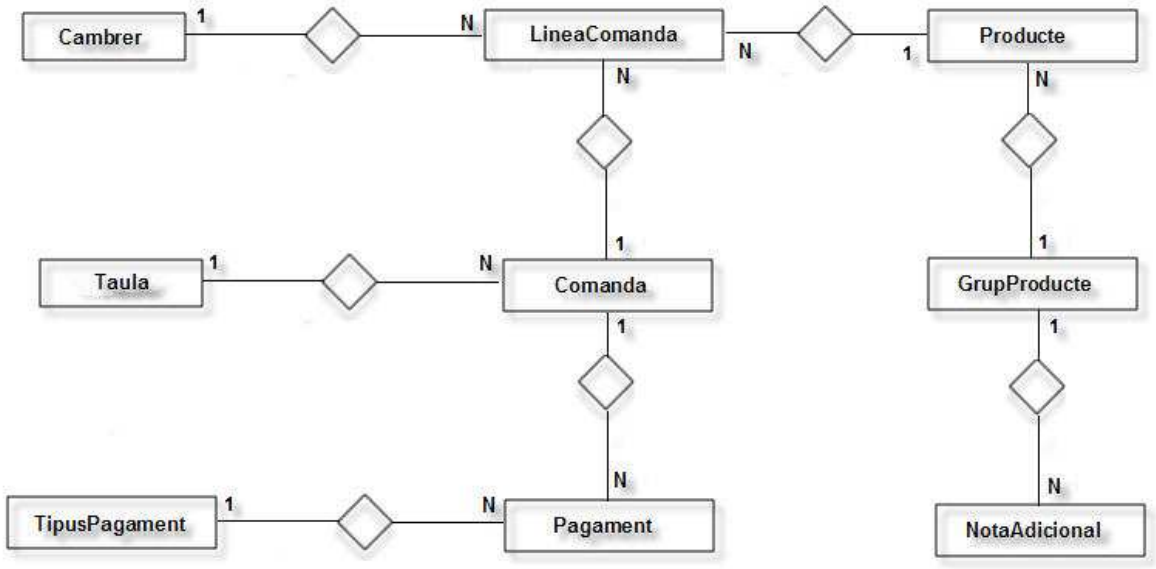
En aquest diagrama podem veure una part de l'arquitectura de comunicació entre els objectes de les classes de dades i el model relacional de la base de dades.

3. ASI:ANÀLISI DEL SISTEMA D'INFORMACIÓ

3.1 MODEL DE DADES

3.1.1 Model Entitat/Relació

Model entitat-relació de la base de dades utilitzada en el projecte.



3.1.2 Taules del model de dades

A continuació es descriu la informació de cada atribut de cada taula de tota la base de dades, així com el tipus, clau primària i claus foranes. Només seran comentats aquells camps que es cregui oportú.

Taula cambrer:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdCambrer	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Nom	VARCHAR(50)			☐ BINARY	NULL	
Cognoms	VARCHAR(50)			☐ BINARY	NULL	
Password	VARCHAR(45)			☐ BINARY	NULL	
Actiu	TINYINT(1)	✓		☐ UNSIGNED ☐ ZEROFILL	1	

Clau principal:IdCambrer
L'atribut *password* serveix per fer el loggin al sistema amb el nom de cambrer i aquesta paraula clau.

Taula taula:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdTaula	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Nom	VARCHAR(45)	✓		☐ BINARY		
Activa	TINYINT(1)	✓		☐ UNSIGNED ☐ ZEROFILL	1	

Clau principal: IdTaula

Taula comanda:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdComanda	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Data	DATETIME				NULL	
IdTaula	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	
Estat	VARCHAR(20)			☐ BINARY	NULL	
Preu	FLOAT	✓		☐ UNSIGNED ☐ ZEROFILL	0	
PreuModificat	FLOAT	✓		☐ UNSIGNED ☐ ZEROFILL	0	

Clau Principal: IdComanda
Clau Forana1 : IdTaula , relacionat amb la Taula taula.

Estat en l'aplicació agafarà els següents valors:Inicial,elaborant,pendent cobrar,i cobrada o no cobrada.
PreuModificat serveix per si el cambrer ha fet una moficació al preu final, normalment s'utilitza per fer un descompte.

Taula GrupProducte:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdGrup	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
NomGrup	VARCHAR(50)			☐ BINARY	NULL	
Disponible	TINYINT(1)	✓		☐ UNSIGNED ☐ ZEROFILL	1	
Enviar_a	VARCHAR(45)	✓		☐ BINARY		

Clau principal:IdGrup
L'atribut Enviar_a, serveix per saber a quina terminal s'enviaran els productes que formin part d'aquest grup. Per exemple un bistec amb patates s'ha d'enviar a la terminal de la cuina.

Taula Producte:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdProducte	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Nom	VARCHAR(50)			☐ BINARY	NULL	
Preu	FLOAT	✓		☐ UNSIGNED ☐ ZEROFILL	0	
IdGrup	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	
Disponible	TINYINT(1)	✓		☐ UNSIGNED ☐ ZEROFILL	1	

Clau principal:IdProducte.
Clau Forana 1:IdGrup,relacionat amb la Taula GrupProducte.

Taula notesAdicionals:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdNota	INTEGER	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Descripcio	TEXT	✓				
IdGrup	INTEGER	✓		☐ UNSIGNED ☐ ZEROFILL	0	

Clau principal:IdNota
Clau forana 1:IdGrup, relacionat amb GrupProducte.

Cada grup de productes pot tenir associades unes notes addicionals. Son notes sobre situacions no previstes que ens poden passar davant la petició de la comanda de un client. Pot passar que el client vulgui per exemple un tall de carn molt cru. Això seria una nota prevista i ens estalviaríem d'haver d'entrar amb la pda aquesta nota.

Taula lineaComanda:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdComanda	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	
IdProducte	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	
Quantitat	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	1	
Estat	VARCHAR(20)			☐ BINARY	NULL	
IdCambrer	INT(10)			☐ UNSIGNED ☐ ZEROFILL	NULL	
IdLineaComanda	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
QuantsFinalitzats	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	
NotaAdicional	TEXT				NULL	
Preferencia	VARCHAR(45)	✓		☐ BINARY		

Clau principal:IdLineaComanda.
Clau forana 1: IdComanda, relacionat amb la Taula comanda.
Clau forana 2:IdProducte, relacionat amb la Taula Producte.
Clau forana 3:IdCambrer, relacionat amb la Taula Cambrer.

Normalment en una línea comanda en tenim prou utilitzant el id de comanda i de producte. Però poden existir línees amb mateix id de comanda i producte ,però amb alguna nota addicional diferent.

L'atribut QuantsFinalitzats servirà als cuiners per saber quants en porten fins al moment. Un cop QuantsFinalitzats==Quantitat la línea passarà a estat finalitzada.

Finalment, l'atribut preferència serveix per indicar l'ordre dels plats. Primer,segon,postres o cap preferència per les begudes,etc.

Taula tipusPagament:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdTipus	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
Descripcio	VARCHAR(45)	✓		☐ BINARY		

Clau principal: IdTipus

Son les diferents maneres amb les que es podrà pagar en el local.Pot ser Visa,metàl·lic, Mastercar,etc.

Taula Pagament:

Column Name	Datatype	NOT NULL	AUTO INC	Flags	Default Value	Comment
IdPagament	INT(10)	✓	✓	☐ UNSIGNED ☐ ZEROFILL	NULL	
IdTipus	INTEGER	✓		☐ UNSIGNED ☐ ZEROFILL	0	
Data	DATETIME	✓			0000-00-00 00...	
Import	FLOAT	✓		☐ UNSIGNED ☐ ZEROFILL	0	
IdComanda	INT(10)	✓		☐ UNSIGNED ☐ ZEROFILL	0	

Clau principal: IdPagament
Clau forana 1:IdTipus,relacionat amb la Taula tipusPagament
Clau forana 2: IdComanda, relacionat amb la Taula comanda.

Ens serveix per saber si una comanda ha estat pagada o no. Si no ho ha estat,es podrà administrar des de l'apartat d'administració a pagaments pendents si ho paguen més endavant.

3.2 ANÀLISI DE CASOS D'ÚS

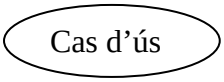
3.2.1 Objectius i Notacions dels Casos d'ús

Els objectius dels Casos d' Ús son:

- Representar els requisits funcionals del sistema gràficament i expressar-los des del punt de vista de l'usuari.
- Explicar el que fa l'aplicació, no com ho fa. Es detallen els passos a seguir per cada cas d'ús.

Notacions dels Casos d'Ús:

Cas d'ús : té forma d'el·lipse amb el nom del cas a dins.



Actor : Un actor es representa amb una figura amb el nom del actor a sota del dibuix.


Actor

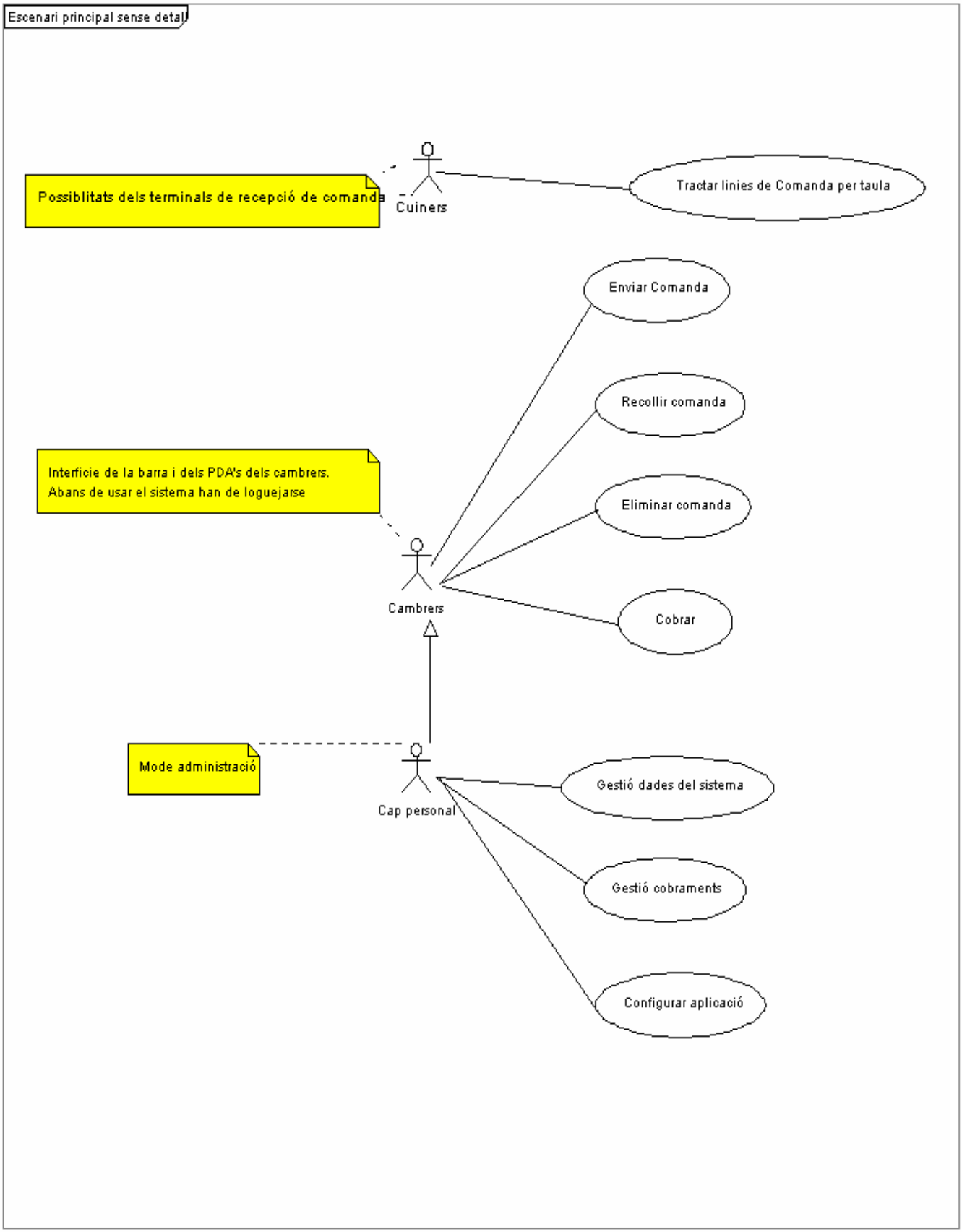
Relacions:

- **Relació Actor-Cas d'ús**: Representació amb una línia contínua:

- **Relació Cas d'ús – Cas d'ús**: Representació amb una línia discontínua. Si porta el missatge <<extend>> vol dir que en un cas d'ús recull el comportament addicional,optatiu, d'un altre cas, i en canvi, si porta el missatge <<include>> significa que un cas d'ús recull el comportament comú d'un altre cas d'ús.

3.2.2 Escenari principal

A continuació es mostra el diagrama de l'escenari principal sense cap detall, més endavant anirem detallant més aquest diagrama.



Observem tres grans distincions sobre l'ús del software.

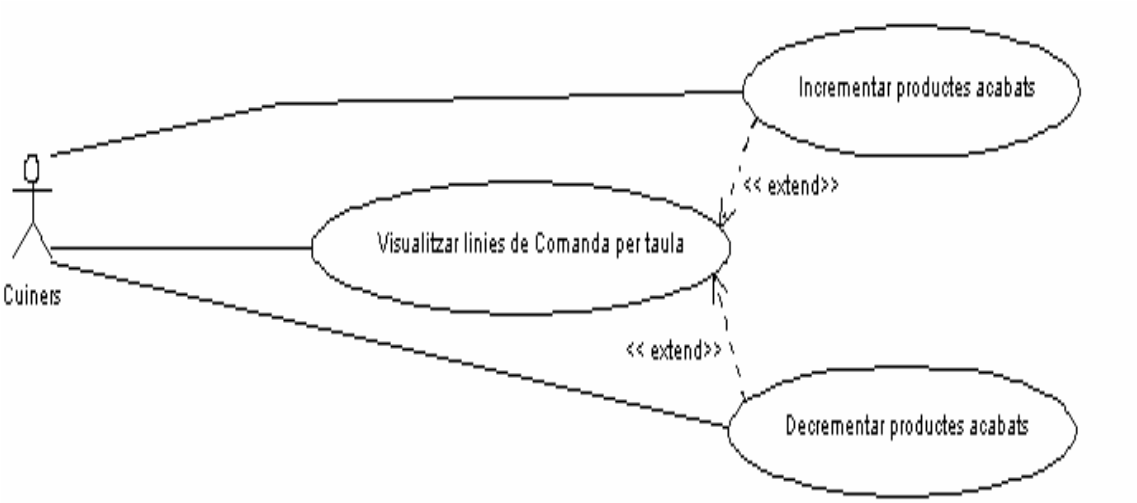
- En primer lloc tenim l'apartat que utilitzaran els treballadors de la cuina, bodega o allà on hi hagi un terminal de recepció de comandes.

- Els cambrers tenen unes funcionalitats destinades a la recepció de comandes cares al públic amb pda's i o tpv.

- Finalment hi haurà aquell usuari amb privilegi per administrar les dades del sistema. Aquest hereta els casos d'ús d'un cambrer, ja que aquesta mateixa persona que administra pot ésser un cambrer. Recordem que estem mirant des de el punt de vista de funcionalitats pels diferents usuaris.

3.2.3 Diagrama de cas d'ús amb detall de cuiners

Aquests casos d'ús seran els que podran utilitzar tot aquell personal que se'n cuidi de una terminal de recepció de comandes.



Fitxes dels casos d'ús de cuiners

1) Visualitzar línees de Comanda per taula

CAS D'ÚS	Visualitzar línees de Comanda per taula
Descripció	Visualitza de manera detallada totes les línees de comanda d'aquella taula.
Actors	Cuiner
Precondició	Existeix alguna taula.
Flux principal	1.- Seleccionar la taula de la que vulguem obtenir la comanda. Punt d'extensió 1:Cas d'ús incrementar productes acabats. Punt d'extensió 2:Cas d'ús decrementar incrementar productes acabats.
Fluxos alternatius	
Postcondició	Tenim una llista amb totes les línees detallades per començar-hi a treballar.
Comentaris	

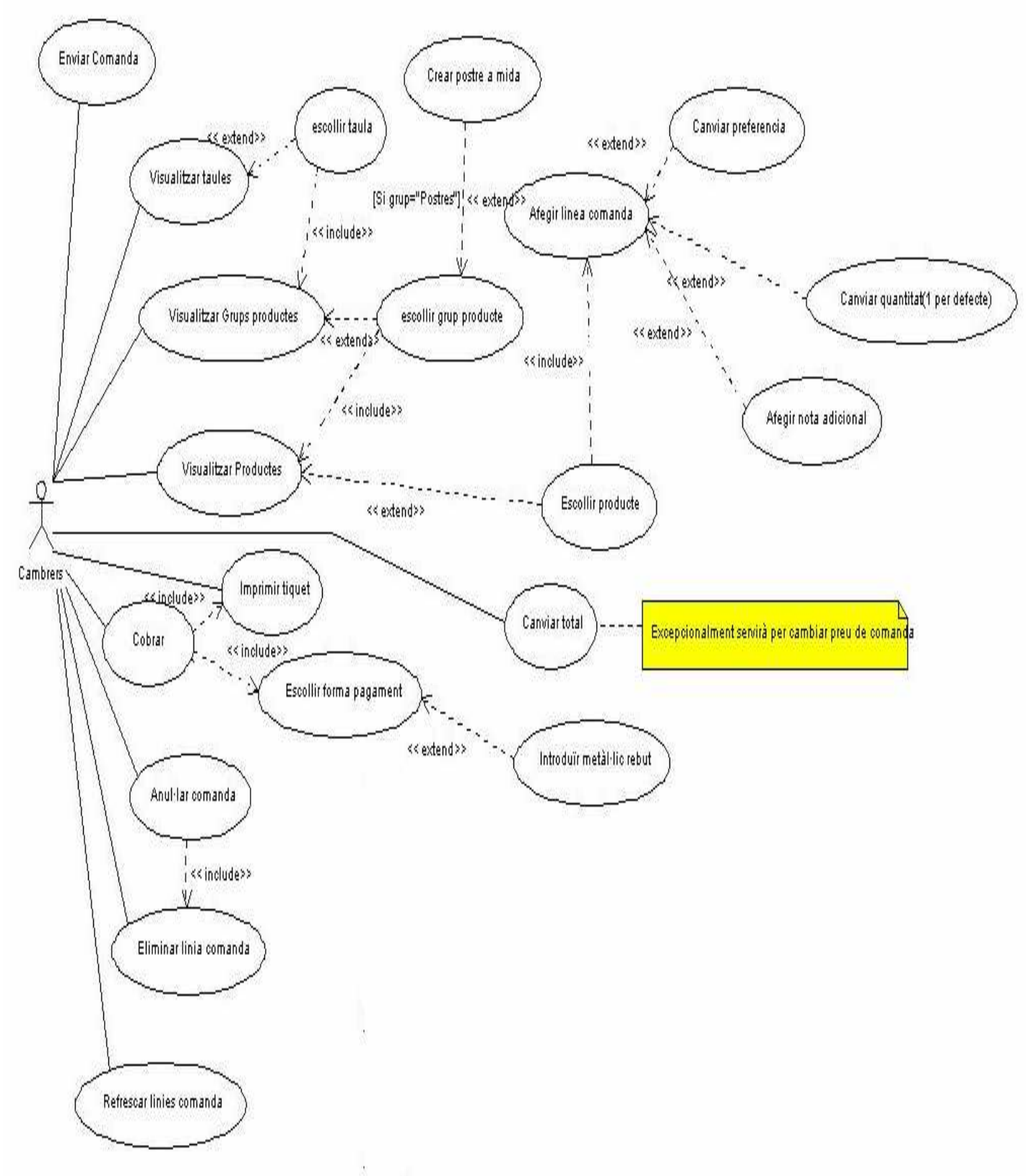
2) Incrementar productes acabats

CAS D'ÚS	Incrementar productes acabats
Descripció	Quan els cuiners hagin acabat d'elaborar un producte, aquest cas d'ús els permetrà reflexar-ho en el terminal per tal de que els cambrers se'n donin compte.
Actors	Cuiner
Precondició	Estem tractant una comanda de una taula existent i existeix alguna línea de comanda.
Flux principal	1.- Prémer el botó de incrementar producte sobre la línea sobre la que vulguem actuar.
Fluxos alternatius	
Postcondició	La línea de comanda on acabem de fer l'acció ha incrementat en 1, el nombre de productes finalitzats.
Comentaris	

3) Decrementar productes acabats

CAS D'ÚS	Incrementar productes acabats
Descripció	Pot passar que els cuiners s'hagin equivocat al marcar els productes que tenen acabats,per tant aquest cas d'ús es per rectificar i restar-ne.
Actors	Cuiner
Precondició	Estem tractant una comanda de una taula existent i existeix alguna línea de comanda.
Flux principal	1.- Prémer el botó de decrementar producte sobre la línea en la que vulguem actuar.
Fluxos alternatius	
Postcondició	La línea de comanda on acabem de fer l'acció ha decrementat en 1, el nombre de productes finalitzats.
Comentaris	

3.2.4 Diagrama de cas d'ús amb detall de cambrers



Fitxes dels casos d'ús de cambrers

1)Enviar Comanda

CAS D'ÚS	Enviar Comanda
Descripció	Envia la comanda als terminals corresponents.Segons el tipus de familia de producte del que sigui cada línia anirà a un terminal o un altre.
Actors	Cambrer
Precondició	Comanda amb un mínim d'una línia de comanda.
Flux principal	1-Prémer el botó d'enviar comanda un cop tinguem la tinguem ja demanada.
Fluxos alternatius	
Postcondició	Comanda enviada i rebuda en els terminals de recepció.
Comentaris	

2)Visualitzar taules

CAS D'ÚS	Visualitzar taules
Descripció	Ens mostra totes les taules disponibles al restaurant amb un color diferent segons l'estat en el que estan.
Actors	Cambrer i cuiner
Precondició	
Flux principal	1-Prémer botó de veure taules. Punt d'extensió 1:Cas d'ús escollir taula.
Fluxos alternatius	
Postcondició	Se'ns mostren totes les taules disponibles al restaurant amb un color diferent segons l'estat en el que estan
Comentaris	

3)Visualitzar Grups de productes

CAS D'ÚS	Visualitzar Grups de productes
Descripció	Llistat amb totes les famílies de productes donades d'alta a l'aplicació.
Actors	Cambrer
Precondició	
Flux principal	1.-Un cop hem escollit una taula,se'ns mostren els grups de productes. Punt d'extensió 1:Cas d'ús escollir grup producte.
Fluxos alternatius	
Postcondició	Llistat amb totes les famílies de productes donades d'alta a l'aplicació.
Comentaris	

4)Visualitzar Productes : És igual que Grups productes, però mostrant productes.

5)Escollir Taula

CAS D'ÚS	Escollir Taula
Descripció	Al escollir la taula sobre la que volem demanar una comanda,se'ns mostra a continuació la pantalla per fer la elecció de productes.
Actors	Cambrer
Precondició	Ha d'existir almenys una taula.
Flux principal	1.-Cas d'ús Visualitzar taula per veure les taules disponibles. 2.- Prémer sobre la taula que desitjem. Punt d'inclusió 1: Visualitzar Grups productes.
Fluxos alternatius	
Postcondició	Tot seguit després de escollir la taula,es mostra el panell per escollir grup producte.
Comentaris	

6) Escollir grup producte

CAS D'ÚS	Escollir grup producte
Descripció	Després de decidir a quina família es troba el producte que volem,cliquem sobre aquest grup.
Actors	Cambrer
Precondició	Existeix almenys un grup de producte.
Flux principal	1.-Clickar sobre el grup de productes que volem. Punt d'inclusió 1: Visualitzar productes.
Fluxos alternatius	1.1-Si el grup escollit és el de família de postres, es mostrarà una funcionalitat especial de elaboració de postres.
Postcondició	Es mostren els productes del grup escollit.
Comentaris	

7) Escollir producte

CAS D'ÚS	Escollir producte
Descripció	Elecció del producte desitjat.
Actors	Cambrer
Precondició	Existeix almenys un producte d'aquesta família.
Flux principal	1.-Clickar sobre el producte que volem. Punt d'inclusió 1: Afegir línia comanda.
Fluxos alternatius	1.1 Si és producte de la família de postres i és concretament el producte "elaborar postre" tenim Punt d'inclusió : Crear Postre a mida.
Postcondició	El producte escollit,s'afegeix a la llista de la comanda amb quantitat 1 i estat inicial.
Comentaris	

8)Crear postre a mida

CAS D'ÚS	Crear postre a mida
Descripció	Ens permet elaborar un postre a partir de altres postres més simples.
Actors	Cambrer.
Precondició	Existeix algun postre.
Flux principal	1.-Escollir el producte amb nom "elaborar postre". 2.-Seleccionar els productes dels que estarà format. 3.-Confirmar
Fluxos alternatius	2.1:Esborrar i començar de nou. 3.1:Cancel·lar.
Postcondició	És afegit a la llista de comanda el nou postre acabat d'elaborar.
Comentaris	

9)Afegir línia comanda

CAS D'ÚS	Afegir línia comanda
Descripció	S'afegeix a la comanda una nova entrada.
Actors	Cambrer.
Precondició	
Flux principal	1.-Escollir el producte desitjat. 2.-Escollir ordre de preferència (per saber en quin ordre servir-lo). Punt d'extensió 1:Afegir nota addicional Punt d'extensió 2:Canviar quantitat de productes Punt d'extensió 3:Canviar preferència
Fluxos alternatius	
Postcondició	Línia afegida a la comanda.
Comentaris	

10)Canviar Preferència

CAS D'ÚS	Canviar Preferència
Descripció	Serveix per indicar als cambrers l'ordre en que han de ser servits els plats. Hi ha preferència de 1r plat,2n plat,postres i sense pref.
Actors	Cambrer.
Precondició	
Flux principal	1.-Escollir la preferència desitjada per les pròximes línees entrants.
Fluxos alternatius	1.1:Pot canviar-se la preferència a una línea ja entrada seleccionant-la i aplicant els canvis.
Postcondició	Cada línia té una preferència seleccionada.
Comentaris	

11)Canviar quantitat

CAS D'ÚS	Canviar quantitat
Descripció	Permet canviar el nombre de productes que volem en una línia de comanda.
Actors	Cambrer
Precondició	Línia existent.
Flux principal	1.-Escollir la línia de comanda que volem canviar. 2.-Introduir la quantitat desitjada. 3.-Guardar canvi.
Fluxos alternatius	2.1:Esborrar quantitat. 3.1:Cancel·lar.
Postcondició	La quantitat de productes de la línia ha canviat.
Comentaris	

12)Afegir nota addicional

CAS D'ÚS	Afegir nota addicional
Descripció	Permet afegir una nota a la línia de comanda.
Actors	Cambrer
Precondició	Línia existent.
Flux principal	1.-Escollir la línia a la que volem afegir la nota. 2.-Escollir les notes previstes al sistema per cada producte. 3.-Guardar canvis.
Fluxos alternatius	3.1:Cancel·lar.
Postcondició	Nota addicional afegida a la línia.
Comentaris	

13) Canviar total

CAS D'ÚS	Canviar total
Descripció	Permet canviar el total final de una comanda.
Actors	Cambrer
Precondició	La comanda té mínim una línia de comanda.
Flux principal	1.-Introduir el preu final que volem. 2.-Prémer el botó de canviar total per guardar-ho.
Fluxos alternatius	
Postcondició	Total de la comanda canviat.
Comentaris	

14)Cobrar

CAS D'ÚS	Cobrar
Descripció	Permet canviar el total final de una comanda.
Actors	Cambrer
Precondició	La comanda està en estat pendent de cobrar. És a dir que tota la comanda ja ha estat servida.
Flux principal	1.-Prémer el botó de cobrar. 2.-Punt d'inclusió : Escollir forma de pagament. 3.-Punt d'inclusió : Imprimir tiquet.
Fluxos alternatius	2.1:Si el tipus de pagament es metàl·lic, introduir import entregat per saber el canvi a retornar. 3.1:Si no s'ha entregat cap quantitat el pagament passa a ser un deute.
Postcondició	Total de la comanda canviat.
Comentaris	

15)Anul·lar comanda

CAS D'ÚS	Anul·lar comanda
Descripció	Ens esborra tota la comanda feta fins ara.
Actors	Cambrer
Precondició	
Flux principal	1.-Prémer el botó de anul·lar comanda. 2.-Repetir fins acabar: Eliminar línia comanda.
Fluxos alternatius	
Postcondició	Comanda eliminada completament.
Comentaris	

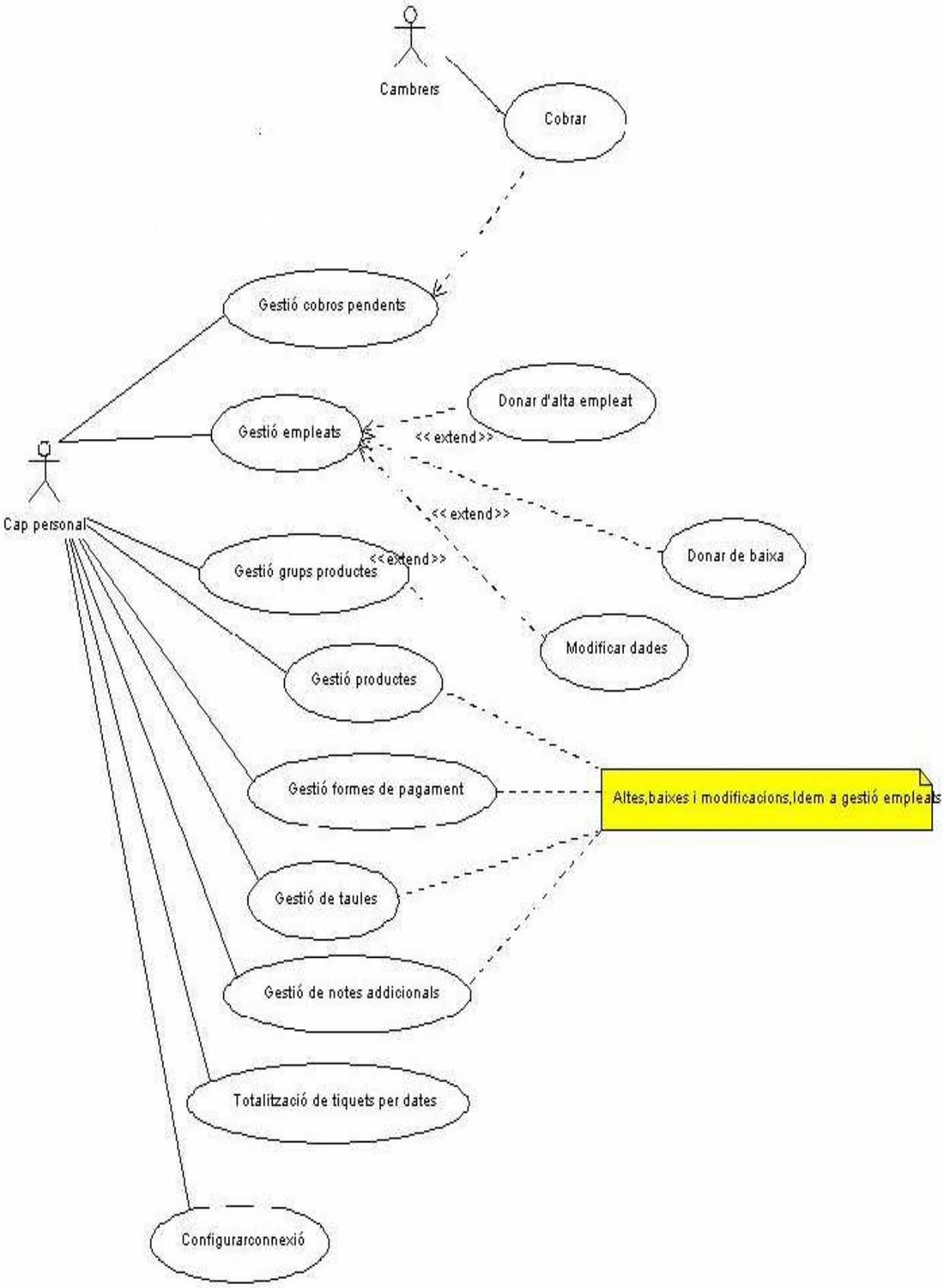
16)Eliminar línia comanda

CAS D'ÚS	Eliminar línia comanda
Descripció	Elimina la línia de comanda seleccionada.
Actors	Cambrer
Precondició	
Flux principal	1.-Seleccionar la línia que volem esborrar. 2.-Prémer el botó de esborrar línia comanda.
Fluxos alternatius	
Postcondició	La línia ha estat esborrada de la comanda
Comentaris	

17)Refrescar línees comanda

CAS D'ÚS	Refrescar línees comanda
Descripció	Actualitza totes les línees per saber l'estat en cada moment.
Actors	Cambrer
Precondició	Comanda enviada anteriorment.
Flux principal	1.-Prémer el botó de refrescar línies de comanda.
Fluxos alternatius	
Postcondició	Tenim la llista de comanda amb l'estat de cada línia actualitzat per saber el seu estat.
Comentaris	

3.2.5 Diagrama de cas d'ús amb detall de Cap de personal



Fitxes dels casos d'ús de Cap de personal

1)Gestió de cobros pendents

CAS D'ÚS	Gestió de cobros pendents
Descripció	Permet gestionar els cobros que tenim pendents.
Actors	Cap de personal
Precondició	
Flux principal	1.-Visualitzar els cobros que tenim pendents Punt d'extensió 1:Cobrar
Fluxos alternatius	
Postcondició	Si hem realitzat la funció de cobrar, tenim el tiquet pagat. Altrament només l'haurem visualitzat i seguirà pendent.
Comentaris	

2)Gestió de empleats

CAS D'ÚS	Gestió de empleats
Descripció	Permet donar d'alta,baixa o modificar el personal del restaurant.
Actors	Cap de personal
Precondició	
Flux principal	1.-Anar a la pantalla de visualització de personal. Punt d'extensió 1:Donar d'alta empleat. Punt d'extensió 2:Donar de baixa un empleat. Punt d'extensió 3:Modificar dades empleat.
Fluxos alternatius	
Postcondició	Dades de empleat gestionades i guardades.
Comentaris	

3)Donar d'alta un empleat

CAS D'ÚS	Donar d'alta un empleat
Descripció	Afegim les dades de un nou treballador al sistema.
Actors	Cap personal
Precondició	
Flux principal	1.-Prémer el botó de nou empleat 2.-Omplir les dades de nom,cognom i password. 3.-Desar les dades.
Fluxos alternatius	3.1-Cancel·lar.
Postcondició	Empleat donat d'alta al sistema.
Comentaris	

4)Donar de baixa un empleat

CAS D'ÚS	Donar de baixa un empleat
Descripció	Eliminem del sistema les dades de un empleat
Actors	Cap personal.
Precondició	Empleat existent.
Flux principal	1.-Seleccionar empleat que volem eliminar. 2.-Prémer botó de eliminar empleat. 3.-Confirmar.
Fluxos alternatius	3.1.-Cancel·lar.
Postcondició	Empleat eliminat del sistema.
Comentaris	

5)Modificar dades de un empleat

CAS D'ÚS	Modificar dades de un empleat
Descripció	Modifica les dades de un empleat donat d'alta.
Actors	Cap personal.
Precondició	Empleat existent.
Flux principal	1.-Seleccionar empleat que volem modificar. 2.-Prémer botó de modificar empleat. 3.-Canviar les dades que es vulgui,excepte l'id. 4.-Desar canvis.
Fluxos alternatius	4.1.-Cancel·lar.
Postcondició	Dades de l'empleat seleccionat canviades.
Comentaris	

6)Gestió de grup producte

CAS D'ÚS	Gestió de grup producte
Descripció	Permet donar d'alta,baixa o modificar les famílies de productes.
Actors	Cap de personal
Precondició	
Flux principal	1.-Anar a la pantalla de visualització de grup producte. Punt d'extensió 1:Donar d'alta un grup producte. Punt d'extensió 2:Donar de baixa un grup producte. Punt d'extensió 3:Modificar dades grup producte.
Fluxos alternatius	
Postcondició	Dades de grup producte gestionades i guardades.
Comentaris	

7)Donar d'alta un grup producte

CAS D'ÚS	Donar d'alta un grup producte
Descripció	Afegim les dades de un nou grup producte al sistema.
Actors	Cap personal
Precondició	
Flux principal	1.-Prémer el botó de nou grup producte 2.-Omplir les dades. 3.-Desar les dades.
Fluxos alternatius	2.1.-Opcionalment es pot carregar una imatge associada a aquest grup de productes. 3.1.-Cancel·lar.
Postcondició	Empleat donat d'alta al sistema.
Comentaris	

8)Donar de baixa un grup producte

CAS D'ÚS	Donar de baixa un grup producte
Descripció	Eliminem del sistema les dades de un grup producte
Actors	Cap personal.
Precondició	Grup producte existent.
Flux principal	1.-Seleccionar grup producte que volem eliminar. 2.-Prémer botó de eliminar grup producte. 3.-Confirmar.
Fluxos alternatius	3.1.-Cancel·lar.
Postcondició	Grup producte eliminat del sistema.
Comentaris	

9)Modificar dades de un grup producte

CAS D'ÚS	Modificar dades de un grup producte
Descripció	Modifica les dades de un grup producte donat d'alta.
Actors	Cap personal.
Precondició	Grup producte existent.
Flux principal	1.-Seleccionar grup producte que volem modificar. 2.-Prémer botó de modificar grup producte. 3.-Canviar les dades que es vulgui, excepte l'id. 4.-Desar canvis.
Fluxos alternatius	4.1.-Cancel·lar.
Postcondició	Dades de grup producte seleccionat canviades.
Comentaris	

10)Gestió de notes addicionals

CAS D'ÚS	Gestió de notes addicionals
Descripció	Permet donar d'alta, baixa o modificar les notes addicionals. Aquestes notes serveixen per intentar preveure els comentaris que poden fer els clients sobre un producte en concret.
Actors	Cap de personal
Precondició	
Flux principal	1.-Anar a la pantalla de visualització de notes addicionals. Punt d'extensió 1:Donar d'alta una nota addicional. Punt d'extensió 2:Donar de baixa una nota addicional. Punt d'extensió 3:Modificar dades nota addicional.
Fluxos alternatius	
Postcondició	Dades de nota addicional gestionades i guardades.
Comentaris	

11)Donar d'alta una nota addicional

CAS D'ÚS	Donar d'alta una nota addicional
Descripció	Afegim les dades de una nova nota addicional.
Actors	Cap personal
Precondició	
Flux principal	1.-Prémer el botó de nova nota addicional. 2.-Omplir les dades. 3.-Desar les dades.
Fluxos alternatius	3.1.-Cancel·lar.
Postcondició	nota addicional donada d'alta al sistema.
Comentaris	

12)Donar de baixa una nota addicional

CAS D'ÚS	Donar de baixa una nota addicional
Descripció	Eliminem del sistema les dades de una nota addicional.
Actors	Cap personal.
Precondició	Nota addicional existent.
Flux principal	1.-Seleccionar la nota que volem eliminar. 2.-Prémer botó de eliminar nota. 3.-Confirmar.
Fluxos alternatius	3.1.-Cancel·lar.
Postcondició	Nota eliminada del sistema.
Comentaris	

13)Modificar dades de una nota addicional

CAS D'ÚS	Modificar dades de una nota addicional
Descripció	Modifica les dades de una nota addicional donat d'alta.
Actors	Cap personal.
Precondició	Nota addicional existent.
Flux principal	1.-Seleccionar nota que volem modificar. 2.-Prémer botó de modificar nota. 3.-Canviar les dades que es vulgui, excepte l'id. 4.-Desar canvis.
Fluxos alternatius	4.1.-Cancel·lar.
Postcondició	Dades de nota seleccionada canviades.
Comentaris	

14)Gestió de empleats

CAS D'ÚS	Gestió de empleats
Descripció	Permet donar d'alta, baixa o modificar el personal del restaurant.
Actors	Cap de personal
Precondició	
Flux principal	1.-Anar a la pantalla de visualització de personal. Punt d'extensió 1:Donar d'alta empleat. Punt d'extensió 2:Donar de baixa un empleat. Punt d'extensió 3:Modificar dades empleat.
Fluxos alternatius	
Postcondició	Dades de empleat gestionades i guardades.
Comentaris	

15)Donar d'alta un empleat

CAS D'ÚS	Donar d'alta un empleat
Descripció	Afegim les dades de un nou treballador al sistema.
Actors	Cap personal
Precondició	
Flux principal	1.-Prémer el botó de nou empleat 2.-Omplir les dades de nom,cognom i password. 3.-Desar les dades.
Fluxos alternatius	3.1-Cancel·lar.
Postcondició	Empleat donat d'alta al sistema.
Comentaris	

16)Donar de baixa un empleat

CAS D'ÚS	Donar de baixa un empleat
Descripció	Eliminem del sistema les dades de un empleat
Actors	Cap personal.
Precondició	Empleat existent.
Flux principal	1.-Seleccionar empleat que volem eliminar. 2.-Prémer botó de eliminar empleat. 3.-Confirmar.
Fluxos alternatius	3.1.-Cancel·lar.
Postcondició	Empleat eliminat del sistema.
Comentaris	

17)Modificar dades de un empleat

CAS D'ÚS	Modificar dades de un empleat
Descripció	Modifica les dades de un empleat donat d'alta.
Actors	Cap personal.
Precondició	Empleat existent.
Flux principal	1.-Seleccionar empleat que volem modificar. 2.-Prémer botó de modificar empleat. 3.-Canviar les dades que es vulgui,excepte l'id. 4.-Desar canvis.
Fluxos alternatius	4.1.-Cancel·lar.
Postcondició	Dades de l'empleat seleccionat canviades.
Comentaris	

Tots els casos d'ús de gestió son molt semblants,per tant no apareixen les fitxes de cas d'ús de les gestions restants.

18)Totalització de tiquets per data.

CAS D'ÚS	Totalització de tiquets per data.
Descripció	Permet ajuntar tiquets que segueixin uns criteris. Es poden obtenir tiquets segons data,hora i tipus de pagament.
Actors	Cap personal.
Precondició	Tenir un mínim de un tiquet.
Flux principal	1.-Visualitzar tiquets. 2.-Escollir la opció de filtratge per tipus de pagament. 3.-Escollir opció de filtratge de data i hora inicial. 4.-Escollir opció de filtratge de data i hora final. Punt d'extensió: Imprimir.
Fluxos alternatius	
Postcondició	Llistat filtrat de tiquets
Comentaris	

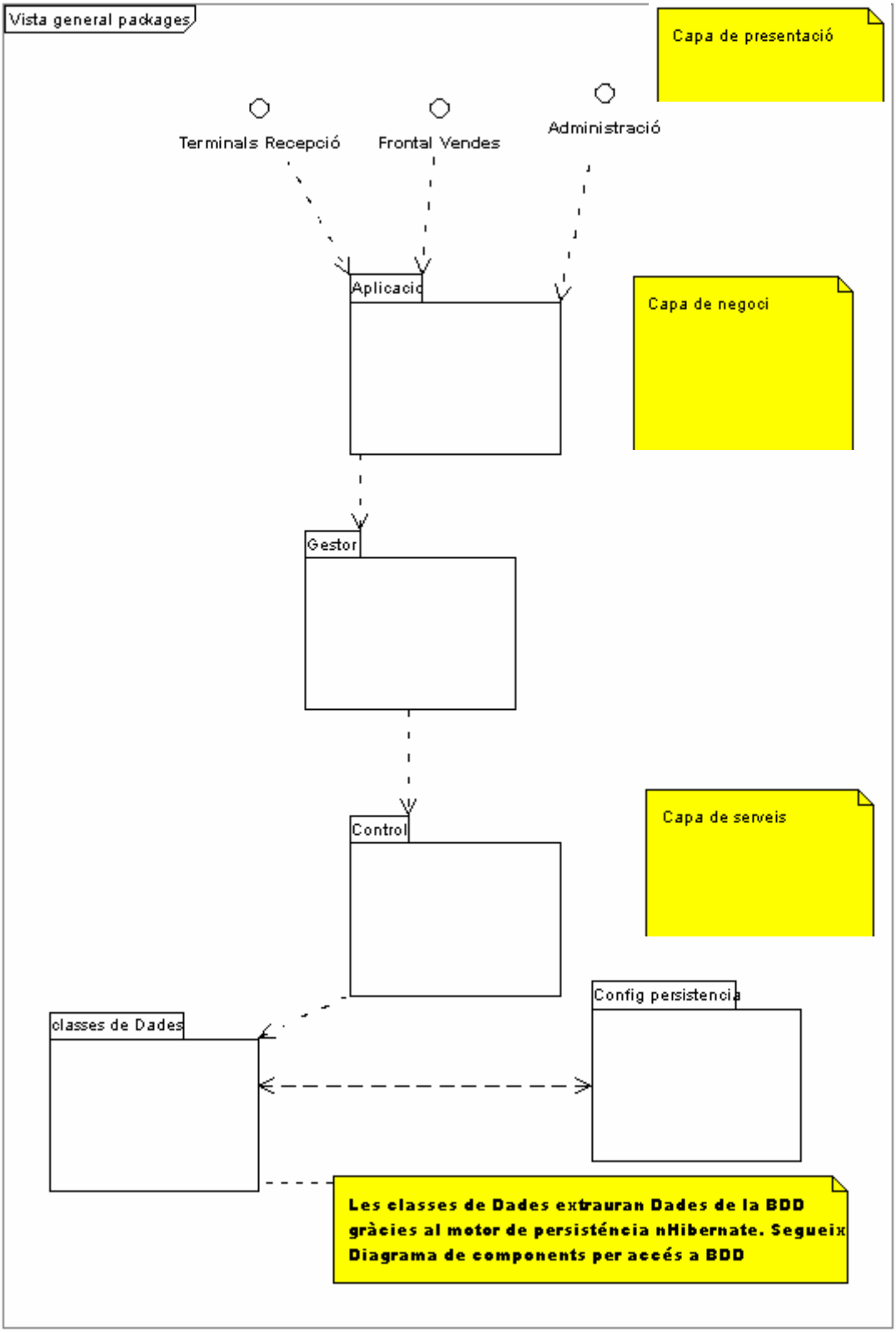
9)Configurar connexió

CAS D'ÚS	Configurar connexió
Descripció	Permet configurar la ip i altres paràmetres de connexió a la base de dades del servidor.
Actors	Cap personal.
Precondició	
Flux principal	1.-Introduir la ip allà on es troba la base de dades. 2-Introduir nom d'usuari de la base de dades. 3-Introduir password per accedir a la base de dades. 4-Aplicar canvis.
Fluxos alternatius	4.1-Cancel·lar
Postcondició	Els paràmetres de configuració han estat modificats.
Comentaris	

3.3 ANÀLISI DE CLASSES

3.3.1 Vista general dels packages utilitzats

A continuació es mostra el diagrama dels paquets de classes utilitzats en el soft. Anirem veient amb detall totes les classes que componen cada paquet, així com la seva relació entre ells.



Tal com es pot comprovar, he establert una jerarquia de classes, en la que és diferencia entre 3 capes diferenciades segons funcions.

- Tenim la capa de serveis que compren el package de classes de dades conjuntament amb els fitxers xml de nhibernate per persistir els objectes amb les dades de la base de dades. Aquesta es la que realitza els accessos a la base de dades tant per llegir-la com per escriure-hi.

- La capa de negoci s'encarrega de comunicar la interfície final de usuari amb les dades de la base de dades i està compresa per els packages control,gestorAplicacio i Aplicacio.

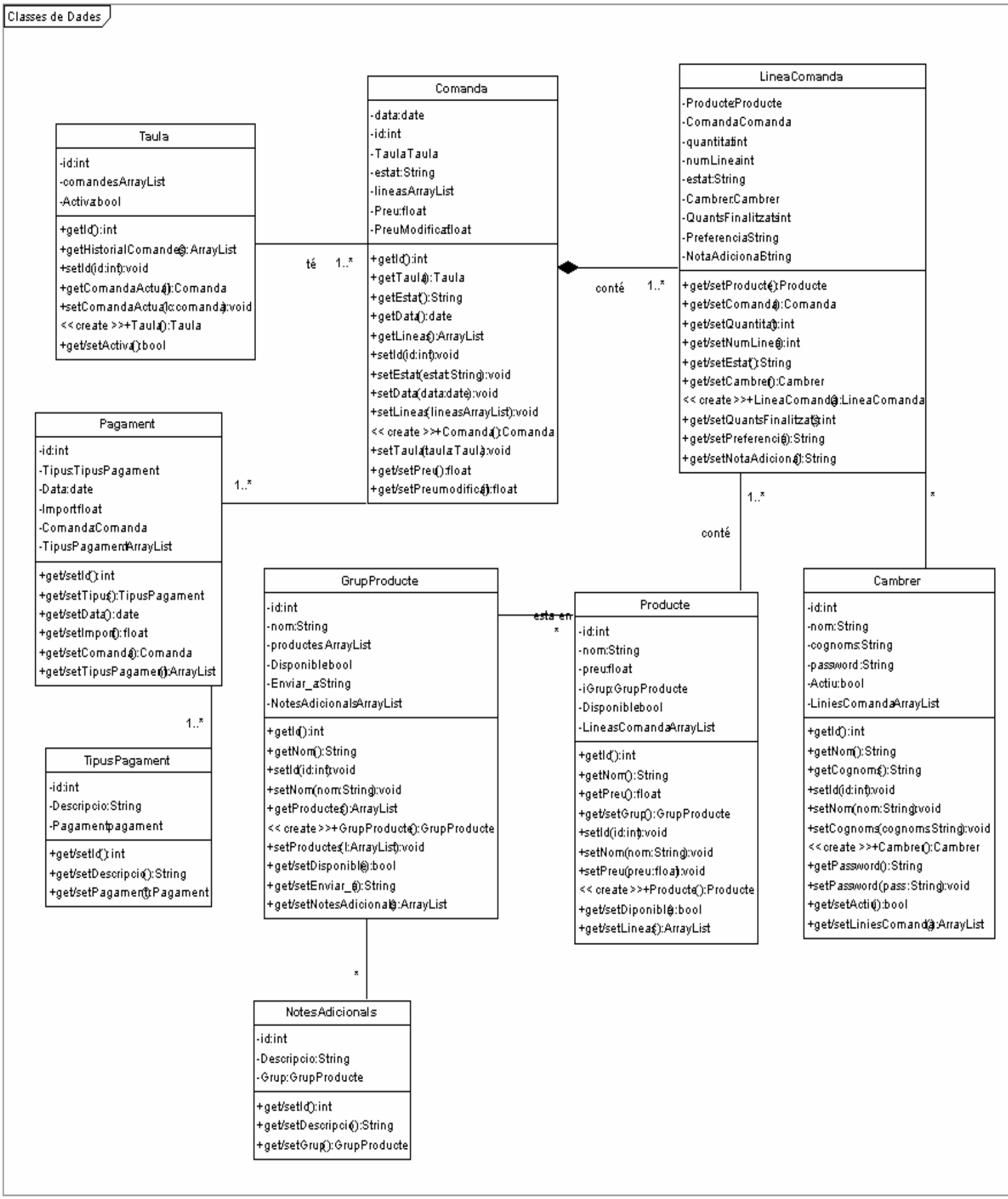
- Finalment la capa de presentació és la que proporciona una manera gràfica de comunicar-se amb totes aquestes dades. Estem parlant de la interfície d'usuari. Per tant aquesta capa la comprenen totes les classes del package interfícies.

Aplicant aquest patró d'arquitectura de sistema obtenim que cada component treballi a un cert nivell d'abstracció. Això porta a que si fem canvis en una part del codi no impliqui canviar totes les capes de sota seu.

3.3.2 Diagrama de classes de dades

A continuació es mostra detalladament totes les classes utilitzades en el desenvolupament.

Primer de tot veurem les classes de les dades principals del sistema,és a dir, aquelles que representaran les taules que hem vist anteriorment, però aquest cop amb objectes.


Classes de Dades
Taula
-id:int
-comandesArrayList
-Activa:bool
+getId():int
+getHistorialComandes():ArrayList
+setId(id:int):void
+getComandaActual():Comanda
+setComandaActual(c:comanda):void
<< create >>+Taula():Taula
+get/setActiva():bool
Comanda
-data:date
-id:int
-TaulaTaula
-estat:String
-lineasArrayList
-Preu:float
-PreuModificatfloat
+getId():int
+getTaula():Taula
+getEstat():String
+getData():date
+getLineas():ArrayList
+setId(id:int):void
+setEstat(estat:String):void
+setData(data:date):void
+setLineas(lineasArrayList):void
<< create >>+Comanda():Comanda
+setTaula(taula:Taula):void
+get/setPreu():float
+get/setPreumodificat():float
té
1..*
LineaComanda
-ProducteProducte
-ComandaComanda
-quantitatint
-numLineaint
-estat:String
-CambrerCambrer
-QuantsFinalitzatsint
-PreferenciaString
-NotaAdicionaBtring
+get/setProducte():Producte
+get/setComanda():Comanda
+get/setQuantitat():int
+get/setNumLinea():int
+get/setEstat():String
+get/setCambrer():Cambrer
<< create >>+LineaComanda():LineaComanda
+get/setQuantsFinalitzats():int
+get/setPreferencia():String
+get/setNotaAdicional():String
conté
1..*
Pagament
-id:int
-Tipus:TipusPagament
-Data:date
-Import:float
-Comanda:Comanda
-TipusPagamentArrayList
+get/setId():int
+get/setTipus():TipusPagament
+get/setData():date
+get/setImport():float
+get/setComanda():Comanda
+get/setTipusPagament():ArrayList
1..*
1..*
TipusPagament
-id:int
-Descripcio:String
-Pagamentpagament
+get/setId():int
+get/setDescripcio():String
+get/setPagament():Pagament
1..*
conté
*
GrupProducte
-id:int
-nom:String
-productes:ArrayList
-Disponiblebool
-Enviar_a:String
-NotesAdicionalsArrayList
+getId():int
+getNom():String
+setId(id:int):void
+setNom(nom:String):void
+getProductes():ArrayList
<< create >>+GrupProducte():GrupProducte
+setProductes(l:ArrayList):void
+get/setDisponible():bool
+get/setEnviar_a():String
+get/setNotesAdicionals():ArrayList
esta en
*
Producte
-id:int
-nom:String
-preu:float
-iGrup:GrupProducte
-Disponiblebool
-LineasComandaArrayList
+getId():int
+getNom():String
+getPreu():float
+get/setGrup():GrupProducte
+setId(id:int):void
+setNom(nom:String):void
+setPreu(preu:float):void
<< create >>+Producte():Producte
+get/setDiponible():bool
+get/setLineas():ArrayList
*
Cambrer
-id:int
-nom:String
-cognoms:String
-password:String
-Actiu:bool
-LiniesComandaArrayList
+getId():int
+getNom():String
+getCognoms():String
+setId(id:int):void
+setNom(nom:String):void
+setCognoms(cognoms:String):void
<< create >>+Cambrer():Cambrer
+getPassword():String
+setPassword(pass:String):void
+get/setActiu():bool
+get/setLiniesComanda():ArrayList
*
NotesAdicionals
-id:int
-Descripcio:String
-Grup:GrupProducte
+get/setId():int
+get/setDescripcio():String
+get/setGrup():GrupProducte

Tal com es pot observar, estem representant totes les dades necessàries del sistema representades abans com a taules pel model relacional en model de classes de programació orientada a objectes.

També es veu com totes les relacions de multiplicitat son de 1 a 1..* o a *. Això depèn de cada software, en el meu cas n'he tingut prou amb aquestes multiplicitats.

També cal dir que cada classe d'aquestes tindrà un fitxer "nomClase.hbm.xml" el qual contindrà un "mapping" de cada atribut de l'objecte amb cada atribut de la taula corresponent a la base de dades. D'aquesta manera es configura el nhibernate.

Veiem per exemple el fitxer de configuració d'hibernate per la classe taula:

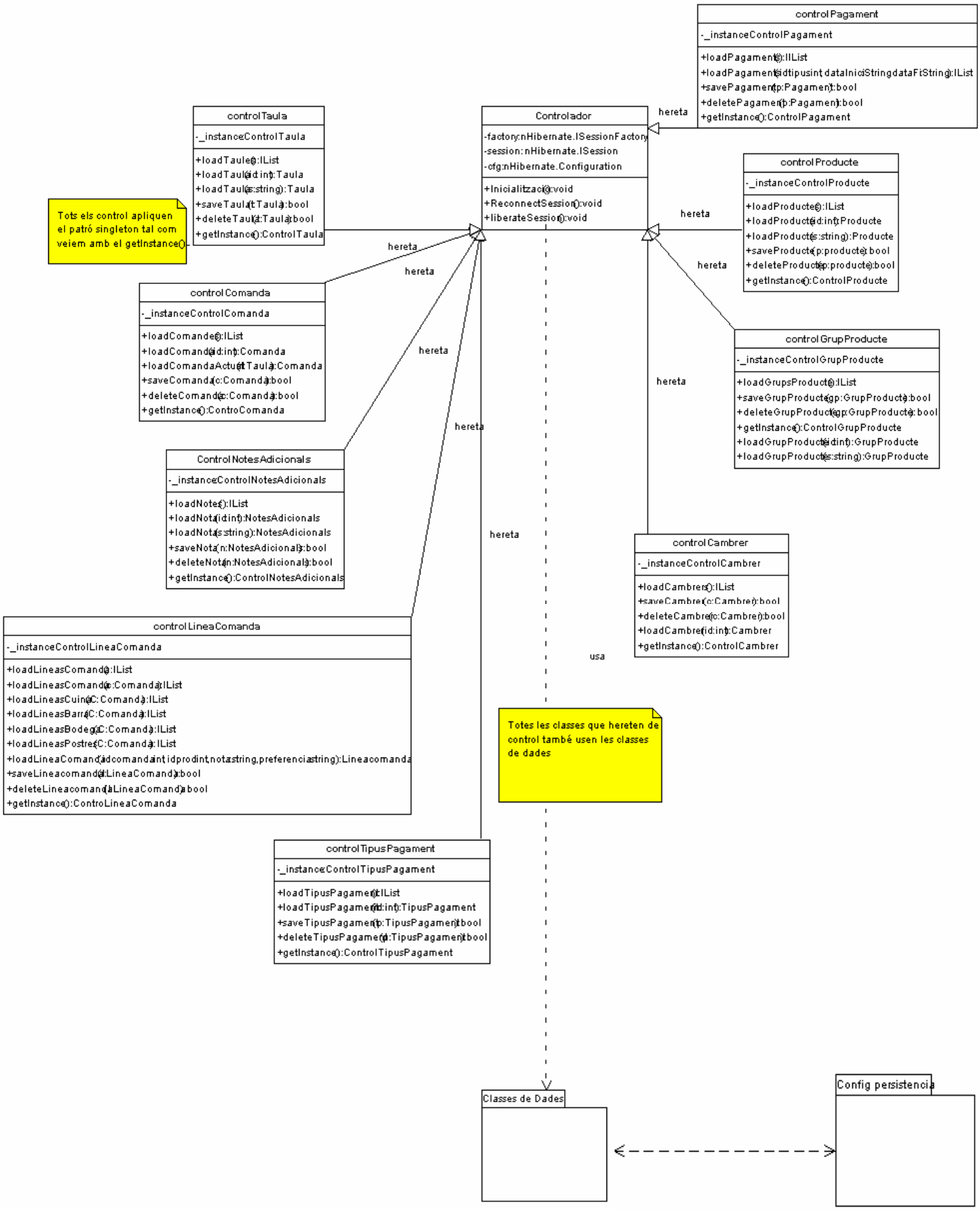
Taula.hbm.xml

```
<?xml version="1.0" encoding="utf-8" ?>

<hibernate-mapping xmlns="urn:nhibernate-mapping-2.0">
  <class  name="PFC.Taula, PFC"
    table="TAULA">
    <id name="Id" column="IDTAULA" type="Int32" unsaved-value="0">
      <generator class="identity" />
    </id>
    <property name="Nom" column="NOM" type="String(50)"/>
    <property name="Activa" column="ACTIVA" type="Boolean"/>


    <bag name="Comandes"  cascade="save-update" lazy="false"   inverse="true" >
      <key column="IDTAULA" />
      <one-to-many class="PFC.Comanda,PFC"  />
    </bag>

</class>
</hibernate-mapping>
```

Veiem com està totalment relacionada amb la taula de nom Taula de la base de dades. I veiem com cada atribut de el model relacional es correspon amb un atribut de la classe.

3.3.3 Diagrama de classes de control

Aquí es detalla el diagrama de classes del package de control el qual comunica l'aplicació amb els serveis i les dades de la base de dades.


controlPagament
-_instanceControlPagament
+loadPagaments():IList
+loadPagaments(idtipusint,dataIniciString,dataFiString):IList
+savePagament(p:Pagament):bool
+deletePagament(p:Pagament):bool
+getInstance():ControlPagament
hereta
controlTaula
-_instanceControlTaula
+loadTaules():IList
+loadTaula(id:int):Taula
+loadTaula(s:string):Taula
+saveTaula(t:Taula):bool
+deleteTaula(t:Taula):bool
+getInstance():ControlTaula
Controlador
-factory:nHibernate.ISessionFactory
-session:nHibernate.ISession
-cfg:nHibernate.Configuration
+Inicialitzacio():void
+ReconnectSession():void
+liberateSession():void
controlProducte
-_instanceControlProducte
+loadProductes():IList
+loadProducte(id:int):Producte
+loadProducte(s:string):Producte
+saveProducte(p:producte):bool
+deleteProducte(p:producte):bool
+getInstance():ControlProducte
Tots els control apliquen el patró singleton tal com veiem amb el getInstance()
controlComanda
-_instanceControlComanda
+loadComandes():IList
+loadComanda(id:int):Comanda
+loadComandaActual(t:Taula):Comanda
+saveComanda(c:Comanda):bool
+deleteComanda(c:Comanda):bool
+getInstance():ControComanda
controlGrupProducte
-_instanceControlGrupProducte
+loadGrupsProducte():IList
+saveGrupProducte(gp:GrupProducte):bool
+deleteGrupProducte(gp:GrupProducte):bool
+getInstance():ControlGrupProducte
+loadGrupProducte(id:int):GrupProducte
+loadGrupProducte(s:string):GrupProducte
ControlNotesAdicionals
-_instanceControlNotesAdicionals
+loadNotes():IList
+loadNota(id:int):NotesAdicionals
+loadNota(s:string):NotesAdicionals
+saveNota(n:NotesAdicionals):bool
+deleteNota(n:NotesAdicionals):bool
+getInstance():ControlNotesAdicionals
controlCambrer
-_instanceControlCambrer
+loadCambrers():IList
+saveCambrer(c:Cambrer):bool
+deleteCambrer(c:Cambrer):bool
+loadCambrer(id:int):Cambrer
+getInstance():ControlCambrer
usa
controlLineaComanda
-_instanceControlLineaComanda
+loadLineasComanda():IList
+loadLineasComanda(c:Comanda):IList
+loadLineasCuina(C:Comanda):IList
+loadLineasBarra(C:Comanda):IList
+loadLineasBodega(C:Comanda):IList
+loadLineasPostres(C:Comanda):IList
+loadLineaComanda(idcomanda:int,idprodint,nota:string,preferencia:string):Lineacomanda
+saveLineacomanda(l:LineaComanda):bool
+deleteLineacomanda(l:LineaComanda):bool
+getInstance():ControLineaComanda
Totes les classes que hereten de control també usen les classes de dades
controlTipusPagament
-_instanceControlTipusPagament
+loadTipusPagament():IList
+loadTipusPagament(id:int):TipusPagament
+saveTipusPagament(p:TipusPagament):bool
+deleteTipusPagament(t:TipusPagament):bool
+getInstance():ControlTipusPagament
Classes de Dades
Config persistencia

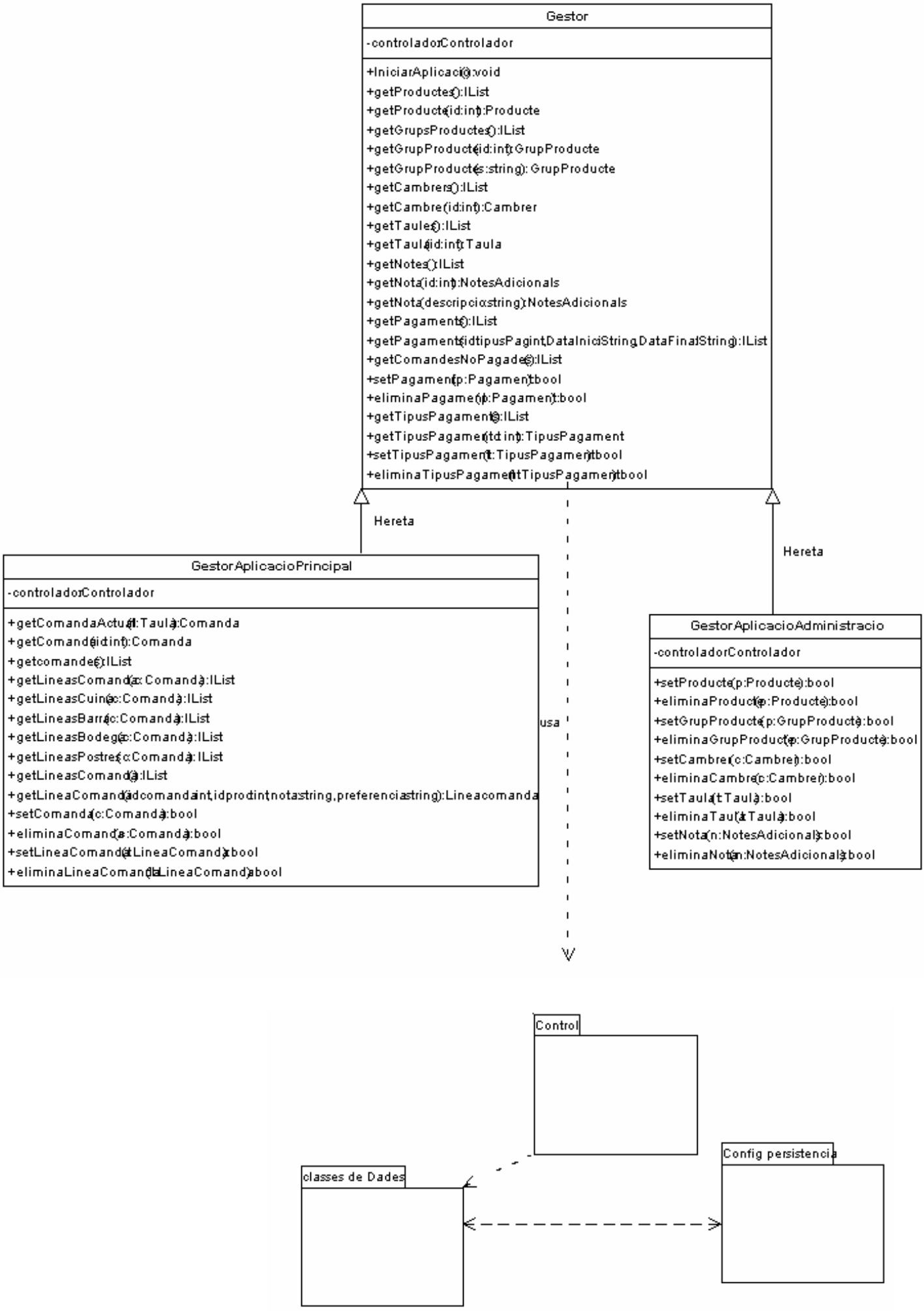
Veiem la classe pare que es controlador. D'aquesta hereten totes les altres.
Control ens proporciona la possibilitat de recollir una sessió, obrir-la,tancar-la, refrescar, etc. Per tant, ens permet gestionar la manera amb la que demanem les coses a la capa inferior de serveis.

Observem que hi ha un controlador per cada tipus de classe de dades. En cada classe de control a part de tenir la sessió comuna heretada del pare, s'implementa a cada classe els mètodes per guardar,carregar i modificar els objectes persistits.

També observem com totes les classes de control utilitzen el **patró de disseny singleton** que serveix per obtenir només una instància de una classe. Això es fa per treballar sempre amb la mateixa sessió i evitar de obrir-ne més.

3.3.4 Diagrama de classes de Gestor

A continuació el diagrama del package gestor. Aquest comunica l'aplicació i la capa de presentació amb el controlador.


Gestor
-controladorControlador
+IniciarAplicacio():void
+getProductes():IList
+getProducte(id:int):Producte
+getGrupsProductes():IList
+getGrupProducte(id:int):GrupProducte
+getGrupProducte(s:string):GrupProducte
+getCambrers():IList
+getCambre(id:int):Cambrer
+getTaules():IList
+getTaula(id:int):Taula
+getNotes():IList
+getNota(id:int):NotesAdicionals
+getNota(descripcio:string):NotesAdicionals
+getPagaments():IList
+getPagaments(idtipusPagint,DataIniciString,DataFinalString):IList
+getComandesNoPagades():IList
+setPagament(p:Pagament):bool
+eliminaPagament(p:Pagament):bool
+getTipusPagaments():IList
+getTipusPagament(id:int):TipusPagament
+setTipusPagament(t:TipusPagament):bool
+eliminaTipusPagament(tTipusPagament):bool
Hereta
Hereta
GestorAplicacioPrincipal
-controladorControlador
+getComandaActual(t:Taula):Comanda
+getComanda(id:int):Comanda
+getcomandes():IList
+getLineasComanda(c:Comanda):IList
+getLineasCuina(c:Comanda):IList
+getLineasBarra(c:Comanda):IList
+getLineasBodega(c:Comanda):IList
+getLineasPostres(c:Comanda):IList
+getLineasComanda():IList
+getLineaComanda(idcomandaint,idprodint,notastring,preferenciastring):Lineacomanda
+setComanda(c:Comanda):bool
+eliminaComanda(c:Comanda):bool
+setLineaComanda(tLineaComanda):bool
+eliminaLineaComanda(tLineaComanda):bool
usa
GestorAplicacioAdministracio
-controladorControlador
+setProducte(p:Producte):bool
+eliminaProducte(p:Producte):bool
+setGrupProducte(p:GrupProducte):bool
+eliminaGrupProducte(p:GrupProducte):bool
+setCambrer(c:Cambrer):bool
+eliminaCambre(c:Cambrer):bool
+setTaula(t:Taula):bool
+eliminaTaula(t:Taula):bool
+setNota(n:NotesAdicionals):bool
+eliminaNota(n:NotesAdicionals):bool
Control
classes de Dades
Config persistencia

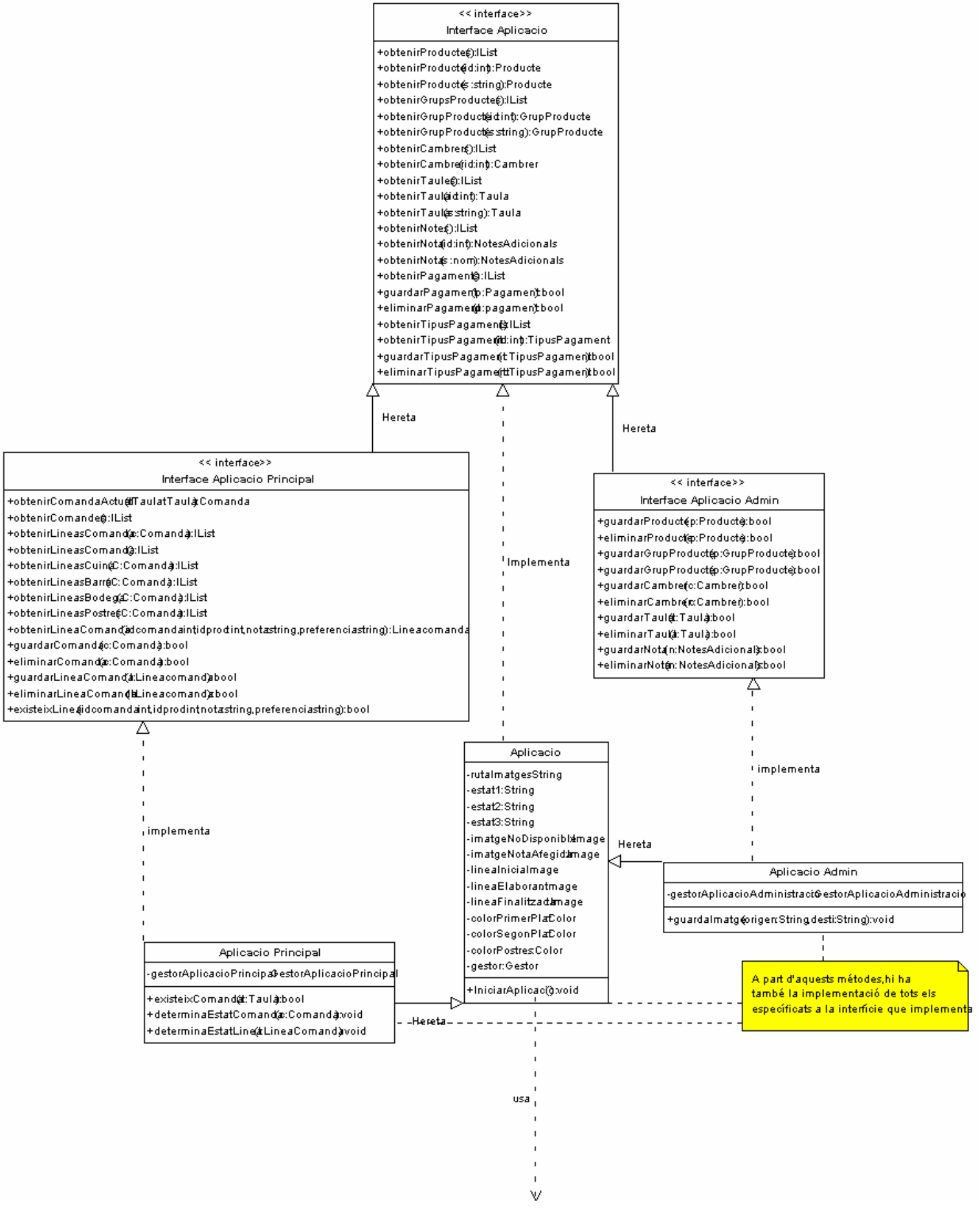
Veiem el pare Gestor del que hereten dos gestors més. El gestor de la aplicació principal, que serà la aplicació que tractarà les comandes, i les línees de comanda bàsicament. És a dir, aquest gestor proporcionarà a les interfícies de les pda's i del frontal de vendes dels cambrers tots els mètodes necessaris per extreure i guardar informació.

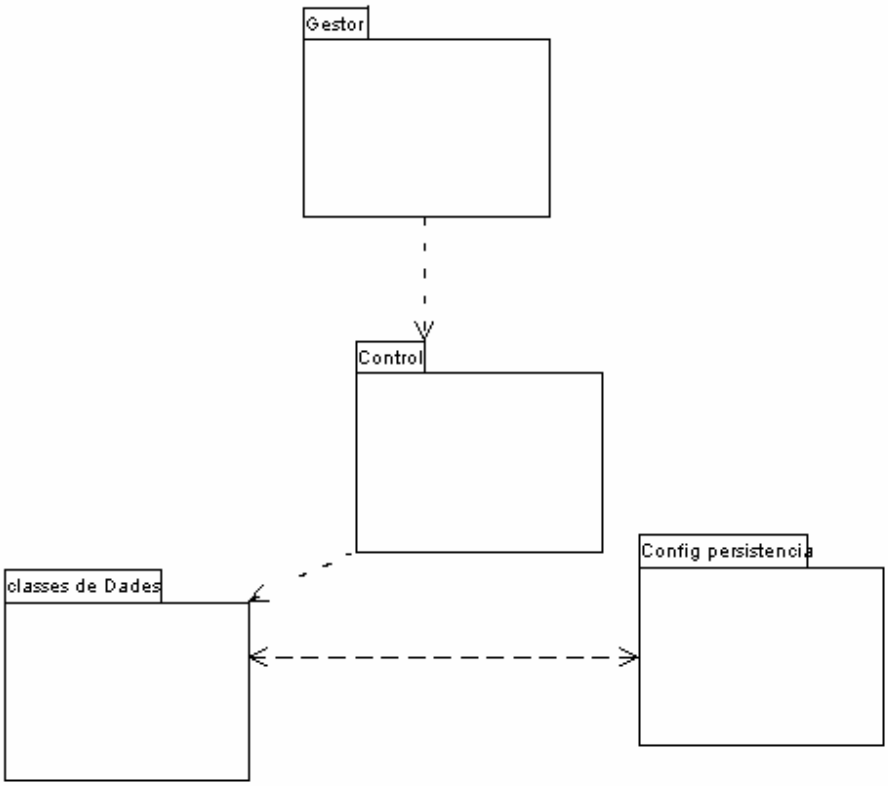
D'altra banda el gestor de l'aplicació administració servirà per brindar mètodes per l'aplicació de gestió del sistema. Aquesta aplicació permetrà donar d'alta els nous productes al sistema, famílies de productes, cambrers, etc., així com també permetrà modificar les dades i eliminar-les.

Aquests dos gestors tindran en comú els mètodes heretats del pare que serveixen per obtenir dades en comú que utilitzaran les dos parts de l'aplicació.

3.3.5 Diagrama de classes de Aplicació

El package aplicació serà el més pròxim a la capa de presentació. Aquest proporciona mètodes amb noms més clarificadors a la capa superior i ofereix mètodes més concrets per les interfícies, a part dels seus inferiors que es comuniquen amb els gestor.


<< interface>>
Interface Aplicacio
+obtenirProductes():IList
+obtenirProducte(id:int):Producte
+obtenirProducte(s:string):Producte
+obtenirGrupsProductes():IList
+obtenirGrupProducte(id:int):GrupProducte
+obtenirGrupProducte(s:string):GrupProducte
+obtenirCambrers():IList
+obtenirCambrer(id:int):Cambrer
+obtenirTaules():IList
+obtenirTaula(id:int):Taula
+obtenirTaula(s:string):Taula
+obtenirNotes():IList
+obtenirNota(id:int):NotesAdicionals
+obtenirNota(s:nom):NotesAdicionals
+obtenirPagaments():IList
+guardarPagament(p:Pagament):bool
+eliminarPagament(id:pagament):bool
+obtenirTipusPagaments():IList
+obtenirTipusPagament(id:int):TipusPagament
+guardarTipusPagament(t:TipusPagament):bool
+eliminarTipusPagament(t:TipusPagament):bool
Hereta
Hereta
<< interface>>
Interface Aplicacio Principal
+obtenirComandaActual(t:TaulatTaula):Comanda
+obtenirComandes():IList
+obtenirLineasComanda(c:Comanda):IList
+obtenirLineasComanda():IList
+obtenirLineasCuina(C:Comanda):IList
+obtenirLineasBarra(C:Comanda):IList
+obtenirLineasBodega(C:Comanda):IList
+obtenirLineasPostres(C:Comanda):IList
+obtenirLineaComanda(idcomanda:int,idprod:int,nota:string,preferencia:string):Lineacomanda
+guardarComanda(c:Comanda):bool
+eliminarComanda(c:Comanda):bool
+guardarLineaComanda(l:Lineacomanda):bool
+eliminarLineaComanda(l:Lineacomanda):bool
+existeixLinea(idcomanda:int,idprod:int,nota:string,preferencia:string):bool
Implementa
<< interface>>
Interface Aplicacio Admin
+guardarProducte(p:Producte):bool
+eliminarProducte(p:Producte):bool
+guardarGrupProducte(p:GrupProducte):bool
+guardarGrupProducte(p:GrupProducte):bool
+guardarCambrer(c:Cambrer):bool
+eliminarCambrer(c:Cambrer):bool
+guardarTaula(t:Taula):bool
+eliminarTaula(t:Taula):bool
+guardarNota(n:NotesAdicionals):bool
+eliminarNota(n:NotesAdicionals):bool
implementa
implementa
Aplicacio
-rutaImatgesString
-estat1:String
-estat2:String
-estat3:String
-imatgeNoDisponibleImage
-imatgeNotaAfegidaImage
-lineaInicialImage
-lineaElaborantImage
-lineaFinalitzadaImage
-colorPrimerPla:Color
-colorSegonPla:Color
-colorPostres:Color
-gestor:Gestor
+IniciarAplicacio():void
Hereta
Aplicacio Admin
-gestorAplicacioAdministracio:GestorAplicacioAdministracio
+guardaImatge(origen:String,desti:String):void
Aplicacio Principal
-gestorAplicacioPrincipal:GestorAplicacioPrincipal
+existeixComanda(t:Taula):bool
+determinaEstatComanda(c:Comanda):void
+determinaEstatLinea(l:LineaComanda):void
Hereta
A part d'aquests mètodes,hi ha també la implementació de tots els especificats a la interfície que implementa
usa


Gestor
Control
classes de Dades
Config persistencia

Aquí podem observar tres interfícies i tres classes que les implementen. En primer lloc tenim la interfície aplicació que al igual que el gestor, proporciona mètodes comuns per els que l'hereten.

La interfície aplicació principal i la de aplicació administració hereten de la interfície pare i actuen igual que el gestor. Tots tenen els mètodes concrets per realitzar les seves tasques i els comuns del pare que s'utilitzaran tant en una aplicació com en l'altra.

Seguidament veiem que aquestes interfícies son implementades per tres classes que segueixen la mateixa jerarquia d'herència entre elles.

La classe aplicació implementa la interfície aplicació i a més , afegeix atributs que seran utilitzats al llarg de la implementació de presentació.

Les classes aplicació principal i aplicació administració, hereten de aplicació i al mateix temps implementen les interfícies principal i administració respectivament. Cada una d'elles implementa els mètodes definits a la interfície i a més afegeix alguns mètodes concrets que han anat sorgint per les exigències de les interfícies de l'aplicació final.

3.4 PROGRAMARI, TÉCNIQUES I LLENGUATGES UTILITZATS

Primers passos i problemes trobats

Primer de tot cal comentar que abans de començar a programar vaig estar investigant sobre llenguatges de programació específics per pda. Però per evitar complicacions vaig decidir que faria la interfície dels pda aprofitant els coneixements que anava adquirint del llenguatge c#.

La segona alternativa, era programar amb el compact framework .net, que és el mateix que el framework que utilitza el .net normal però amb moltes limitacions ja que està destinat només a dispositius mòbils com pda's.

Però va sorgir un altre problema, el compact framework no suporta el nhibernate, pel que havia de decidir entre nhibernate o compact framework. Evidentment em vaig decantar per seguir amb nhibernate que és el més important d'aquesta aplicació.

Per tant, vaig haver de decidir que la interfície del pda seria feta integrament amb el framework .net original. Això implica que l' executable resultant no funcionaria en una pda. I per aquest fet he decidit utilitzar la connexió de terminal service de les pda. Això permet connectar-se a una sessió de un servidor on en aquest hi hagi el software instal·lat.

Així només falta escollir que volem interfície de pda al programa i automàticament se'ns mostra la interfície reduïda a la mida de un dispositiu mòbil.

El fet de utilitzar la comunicació a escriptori remot dona unes certes avantatges respecte la programació de un soft específic per una pda.

- Si la bateria de la pda s'acaba en un moment donat,no es perd cap dada ja que la sessió queda en l'últim estat que estava abans d'apagar-se. Només falta tornar a connectar-se a la sessió i seguir per allà on ho havíem deixat.

- Descarreguem tota la feina de la pda, cada acció, mètode, càlcul es realitza en el servidor on està connectada remotament. La pda només ens proporciona la interfície.

- Finalment, des de el punt de vista de programació, es un estalvi enorme de temps el fet d'aprofitar en un 75% la interfície creada primerament per una tpv, simplement s'ha de reduir i casi no es toca res de codi.

A continuació es mostra tot el que finalment he utilitzat per realitzar el desenvolupament del programa.

Programari:

Per realitzar tota la part d' anàlisi de casos d'ús, de classes de desplegament i altres he utilitzat una eina CASE anomenada **Poseidon for UML.**

Gràcies a les eines CASE com aquesta, es pot obtenir generació de codi en varis llenguatges per estalviar-nos la feina a la hora de programar i evitar moltes repeticions.

D'altra banda, per realitzar la implementació del software he utilitzat la tecnologia .net, concretament el llenguatge **visual c#.** I he treballat amb el **Microsoft Visual studio 2005.**

Per realitzar les proves de la interfície de la pda he utilitzat una pocket pc HP OPTICON i un programa que emula aquesta pda en un pc amb windows per treballar més comodament. Per tant he treballat conjuntament amb el **Activesync** per sincronitzar el pc i la pocket, i amb el programa **Activesync remote display.**

Pel tractament de colors i imatges he utilitzat varis editors de imatges:

- Adobe photoshop CS
- Paint shop Pro
- Snagit Editor

Tècniques:

Tal com he comentat en un altre apartat he utilitzat un framework de persistència anomenat *nhibernate* per .net.

A part d'això, he utilitzat tècniques del UML que ens permet reutilitzar el codi, modificar-lo sense problema i fer-lo extensible sempre que vulguem. També s'han utilitzat patrons de disseny comentats posteriorment amb més detall.

Per guardar les configuracions de impressió, configuració i d'altres he utilitzat fitxers xml. Aquests fitxers contenen la configuració per cada usuari, per tant, es guarda en la carpeta de "dades de programa" de cada usuari per tal de no trepitjar-se les configuracions.

He utilitzat recursos del llenguatge .net per tractar les lectures i escriptures de xml. De fet .net està orientat al treball amb aquest tipus d'estandart i no hi ha cap tipus de dificultat per fer-ne un correcte ús.

3.5 ESTRUCTURA I PROTOTIPUS DE INTERFÍCIES

3.5.1 Interfícies principals

Aquest sistema informàtic té varies interfícies totalment diferenciades.

- Primerament hi ha la necessitat de un apartat des de el que es puguin dona d'alta tots els productes, treballadors, configuracions,etc. Per tant, estem parlant de una primera **interfície d'administració**.

- Hi ha d'haver una **interfície d'usuari pels pantalles tàctils de les terminals de punt de venta (tpv)** , que se'n podrà utilitzar només una (a la barra) o més (un en cada sector del restaurant) segons conveniència. Aquest apartat del programa és el que permet realitzar totes les funcionalitats de recollida de comandes, enviar, cobrar, imprimir tiquet, etc. Es comentarà amb tot detall al següent capítol.

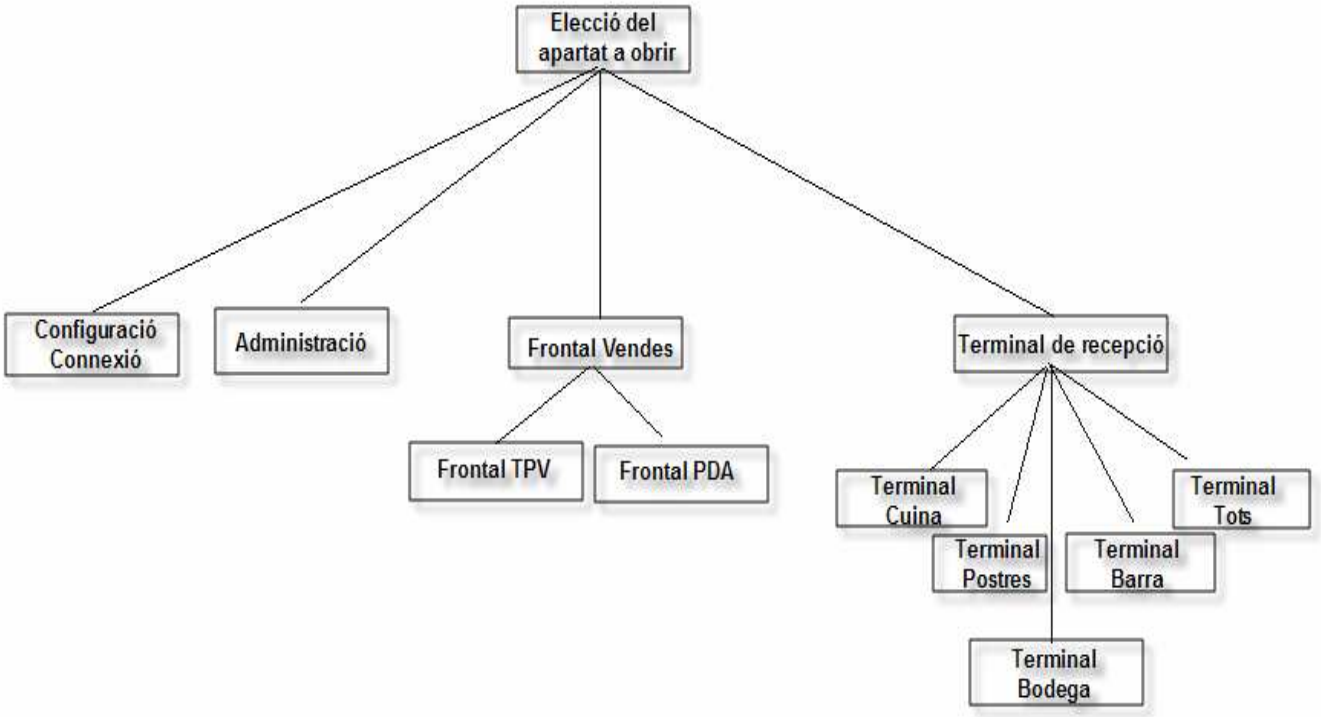
- Evidentment es necessita també una **interfície per les pda**. Aquesta serà exactament igual en quant a funcionalitat que la comentada anteriorment, però amb una interfície adaptada a la petita mida del dispositiu portàtil.

- Hi ha una **interfície per les pantalles tàctils de rebuda de comandes**. Es poden configurar tantes terminals com vulguem segons la organització del local. Per exemple es pot tenir una terminal de rebuda per la cuina,una per la barra,una altre a la bodega,etc.

- Finalment també hi ha una **interfície de configuració de connexió**. La considero de les principals perquè és al primer lloc on accedirem abans d'entrar a l' aplicació per configurar bé la connexió.

A part d'aquestes interfícies principals, n'hi ha de secundaries, com les de elecció de tipus de terminal, o elecció sobre quin apartat del soft volem executar. Es mostraran a continuació en un petit gràfic.

A continuació un gràfic que representa totes les interfícies que trobarem al programa.

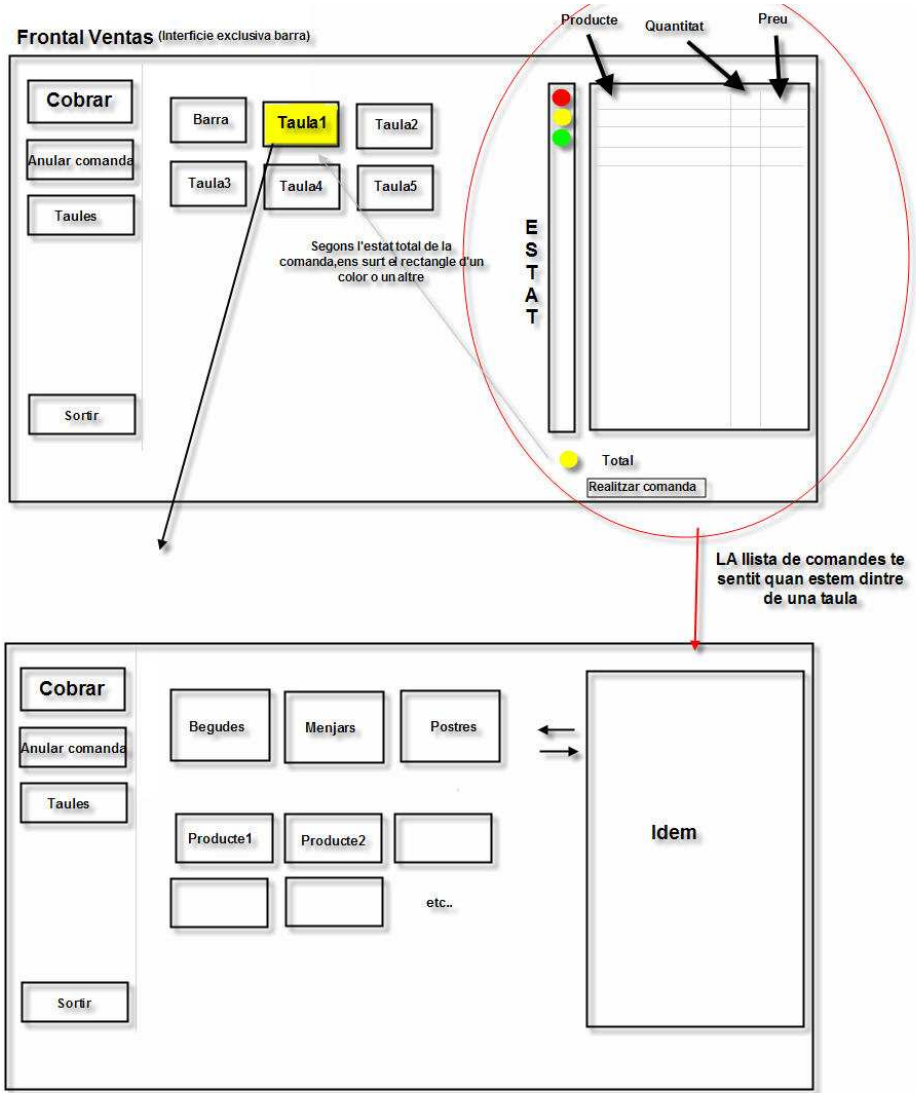


3.5.2 Alguns prototipus d'interfície

Durant la fase d'anàlisi les interfícies son molt poc concretes, però sempre va bé basar-se en alguna idea. He intentat plasmar els requeriments en forma de interfície per poder evolucionar.

A continuació es mostren uns prototips molt poc concrets sobre els que vaig planificar l'anàlisi i que més endavant evolucionaran a interfícies ben definides i acabades.

- Prototip interfície de frontal de vendes per tpv.

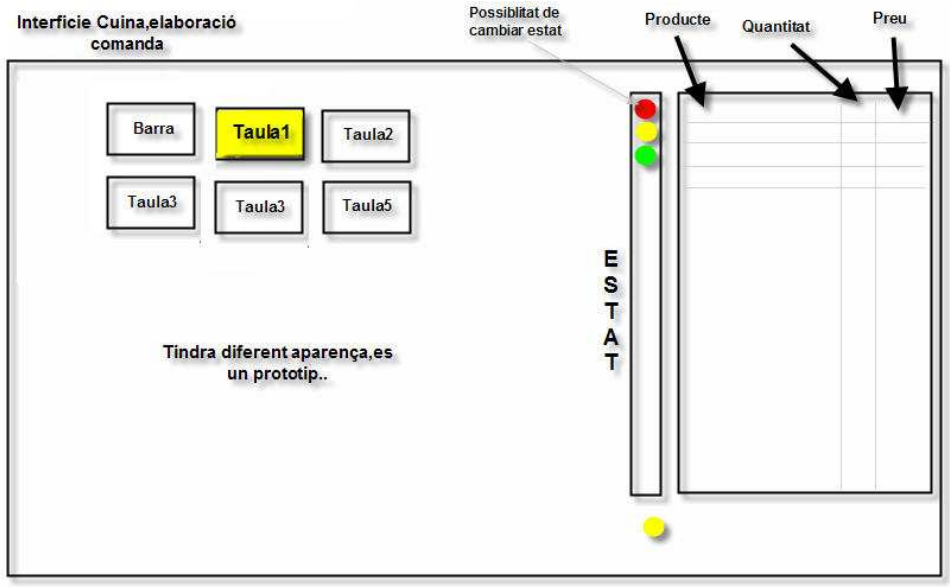


El més destacable és el fet dels colors del estats. Més endavant durant l'evolució del sistema profunditzaré sobre el significat de cada estat i la manera de canviar-los i de treballar amb ells.

També es destaca una idea de mantenir una grup de botons visibles des de cada apartat de la interfície al llarg de tota l'execució.

A la primera imatge veiem les taules disponibles en el local amb un color segons l'estat en el que estan i al seu costat la llista de comanda d'aquella taula. Veiem que cada línea té també un color que representa un estat.

- Prototip interfície de terminal de recepció de comandes.



Seguint la línia del primer prototipus,aquesta també mostra la llista de comanda de cada taula i el color segons l'estat. El que el diferencia de les terminals de recollida de comanda es que aquestes son simplement de rebuda i tenen la funcionalitat de poder canviar d'estat al acabar de elaborar els productes.

Es una fase molt inicial per decidir la aparença final de la interfície. Més endavant s'incidirà en l'acabat i cada funcionalitat de la mateixa.

4. DSI:DISSENY DEL SISTEMA D'INFORMACIÓ

4.1 Patrons de disseny

Des de un principi he seguit la idea de fer un software amb els objectius que pretén assolir l'UML, és a dir, un codi reutilitzable, senzill de mantenir i totalment modulat per evitar l' acoblament i tenir una alta cohesió.

Els patrons de disseny ajuden molt a complir aquests objectius i aquest software segueix bastants d'aquests patrons comentats a continuació.

4.1.1 Patrons GRASP

GRASP es l'acrònim de General Responsibility Assignment Software Patterns.

Una de les coses més complicades en Orientació de objectes consisteix en elegir les classes adequades i decidir com aquestes classes han d'actuar.

Els patrons han adquirit una gran importància en el disseny i implementació d' aplicacions, brindant solucions que poden ser usades una i altre vegada, adaptant-les a l' escenari del desenvolupament.

Els patrons GRASP son necessaris en el desenvolupament de un software ja que ens indiquen com:

- Preveure el cost futur de la execució del projecte
- Evitar riscos amb tasques no planificades.
- Eliminar dependències a persones per produir peces de una manera estandarditzada i documentada.
- Augmentar la confiança en els sistemes desenvolupats i reduir els seus errors
- Afavorir la reutilització de mòduls (amb la conseqüent reducció de costos).

Un cop introduïts els patrons GRASP comentaré els utilitzats en el desenvolupament del meu software:

Patró Expert:

Tal com indica el seu nom, és un patró expert en informació. Aquest patró diu que la millor classe per realitzar una tasca es aquella que té accés a totes les dades que fan falta.

Per tant cada classe té unes responsabilitats concretes segons les seves dades i una mateixa classe no pot tenir responsabilitats que haurien de ser d'altres classes.

Per exemple, la classe Cambrer té la responsabilitat de saber el nom del cambrer, cognoms i password i cap més.

Patró Creator:

El que defineix aquest patró és que una instància de un objecte l'ha de crear l' objecte que té la informació.

El més senzill per comprendre aquest patró es amb un exemple.

Per exemple, tinc la classe línies de comanda que té les responsabilitats de donar informació sobre una línia de comanda. Però que passa si vull obtenir varies línies de comanda? El patró creador ens diu que podem crear una classe per tal de poder complir amb el que volem. Per tant tinc la classe aplicació que s'encarregaria de recollir les línies que vulguem.

Patró High Cohesion (Alta cohesió):

Aquest patró diu que una classe ha de tenir un comportament el més cohesionat possible, és a dir que ha de fer coses el més específiques possibles.

La idea principal és que una classe tingui pocs mètodes públics perquè en lloc de que hi hagi una super classe que ho faci tot, hi hagi moltes classes que facin poques tasques cada una de forma coordinada. D'aquesta manera és molt més fàcil de mantenir el codi.

Un exemple clar és el package Classes de dades, on cada tipus de dada del sistema és representat per la seva classe amb els mètodes justos i amb un alt grau de cohesió.

Patró Low Coupling (Baix acoblament):

Aquest patró diu que una classe ha de tenir la menor quantitat possible de coneixement sobre les altres classes. Amb això s'aconsegueix que sigui més fàcil de reutilitzar i reemplaçar.

Per exemple, si hi hagués un alt acoblament les classes de dades com comanda, hauríem de conèixer detalls de la connexió. Això implicaria que qualsevol canvi en la connexió ens obligaria a canviar totes les dades que depenen d'aquella informació. Però aplicant aquest patró, qualsevol canvi en la connexió no implicarà per res a cap classe superior.

Patró Controlador:

Aquest patró assigna la responsabilitat de controlar el fluxe d' events del sistema a classes específiques. Això facilita la centralització d'activitats.

El controlador no realitza aquestes activitats, simplement les delega a altres classes amb les que manté un model de alta cohesió.

Veiem un exemple claríssim al package Controlador del software, que delega activitats segons les dades que es requereixen.

Patró Pure fabrication (Pura fabricació):

Serveix per assignar un conjunt cohesionat de responsabilitats a una classe artificial (no representa cap concepte del domini del problema).

Aquesta classe és simplement una fabricació de la imaginació i ha de ser pura (dissenyada exclusivament per tal fi).

Un exemple clar, és la classe aplicació. És una classe inventada per facilitar el treball des de la interfície i tenir mètodes amb noms més entenedors i mètodes més polits pel seu ús final a la interfície.

4.1.2 Patrons de disseny GOF

Els Patrons de Disseny son la base per la cerca de solucions a problemes comuns en el desenvolupament del software i altres àmbits referents al disseny de interacció o interfícies.

És una solució a un problema no trivial que es efectiva (ja s'ha resolt el mateix problema en ocasions anteriors) i reutilitzable (es pot aplicar a diferents problemes de disseny en diferents circumstàncies).

Dins dels patrons de disseny hi ha tres tipus de patrons que són els de creació, estructurals i de comportament. A continuació comento els utilitzats en el meu software.

4.1.2.1 Patrons de disseny de creació utilitzats

Patró singleton (instància única)

Està dissenyat per restringir la creació de objectes que pertanyen a una classe o el valor d'un tipus a un únic objecte.

La seva intenció consisteix en garantir que una classe només tingui una instància i proporcionar un punt d' accés global a ella.

Aquest patró s' implementa creant a la classe un mètode que crea una instància del objecte només si encara no existeix alguna. Per assegurar que la classe no pot ser instanciada novament es regula l'abast del constructor (amb atributs protegits o privats).

Les situacions més habituals d' aplicació d'aquest patró son aquelles en les que una classe controla l'accés a un recurs físic únic (com pot ser el mouse o un fitxer obert en mode exclusiu) o quan determinat tipus de dades ha d' estar disponible per tots els demés objectes de la aplicació(aquest és el meu cas).

El patrón *singleton* proveeix una única instància global gràcies a que:

- La pròpia classe es responsable de crear la única instància.
- Permet l'accés global a dita instància per mitjà d' un mètode de classe.
- Declara el constructor de classe com privat per que no sigui instanciable directament.

L'he utilitzat en totes les classes que accedeixen a dades i aquí va un exemple del codi:

"ControlCambrer.cs"

```
public  class ControlCambrer : Controlador
   {
      static private ControlCambrer _instance;


public static ControlCambrer getInstance()
     {
        if (_instance == null) { _instance = new ControlCambrer(); }
        return _instance;
              }
```

Patró Factory Method

Centralitza en una classe constructora la creació d' objectes de un subtipus de un tipus determinat, ocultant al usuari la casuística per escollir el subtipus a crear.

Consisteix en utilitzar una classe constructora abstracta amb uns quant mètodes definits i altre(s) abstracte(s): el dedicat a la construcció d'objectes de un subtipus de un tipus determinat. Es una simplificació del Abstract Factory, en la que la classe abstracta té mètodes concrets que usen alguns dels abstractes; segons utilitzem una o altra filla d'aquesta classe abstracta, tindrem un o altre comportament.

El factory l'usa principalment l'hibernate per treballar amb sessions i per no dependre del tipus de bases de dades amb la que treballa. Veiem un tall de codi.

"**Controller.cs**"

```
namespace PFC.Controller
{

   public  class Controlador
   {
      static protected ISessionFactory factory; //El factory serà compartit per tota l'aplicació
      static protected ISession session ; //Sessió diferent per cada instancia de classe
      static Configuration cfg; //També pot ser compartida per l'aplicació sencera, cap problema..



   public  void Inicialitzacio(){ //Només es cridarà un cop per aplicació..

       try
       {
          cfg = new Configuration();
          cfg.AddAssembly("PFC");

          factory = cfg.BuildSessionFactory();
          session = factory.OpenSession(); //Sessió diferent per cada instancia de classe
       }
       catch (Exception) { throw new Exception(); }
    }

   public void ReconnectSession()
   {
      session.Flush();
      session.Close();
      session = factory.OpenSession();
            }
```

4.1.2.2 Patrons de disseny estructurals utilitzats

Patró Façade

Coneix quines classes son responsables de quines peticions i així les delega als objectes apropiats.

Per exemple el meu package Gestor, delega les peticions al controlador indicat. Si es requereix informació de un producte, aquest delegarà la petició al controlador producte, si es necessita de una comanda, la petició serà per el controlador comanda i així successivament.

```
namespace PFC.gestorAplicacio
{
   class gestor
   {
//*****************************Obtenim tots els productes disponibles************************
       public IList getProductes(){
           IList i = null;
           try
           {
             i= ControlProducte.getInstance().loadProductes();
           }
           catch (Exception ex) { MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error); }
           return i;
       }


//*****************************Obtenim tots els cambrers disponibles************************
       public IList getCambrers()
       {
           IList i = null;
           try
           {
           i= ControlCambrer.getInstance().loadCambrers();
            }
            catch (Exception ex) { MessageBox.Show(ex.Message, "Error",
MessageBoxButtons.OK,MessageBoxIcon.Error); }
            return i;
                  }
```

4.1.2.3 Patrons de disseny de comportament utilitzats

Patró Chain of responsability (Cadena de responsabilitat)

Es defineix una interfície per manejar peticions. Aquesta petició la atén la classe que implementa la interfície i si no pot la delega a una nivell inferior. Aixa dons es va enviant la petició als successors fins a trobar el que tingui aquella responsabilitat concreta.

Aquest patró està molt lligat amb el patró GRASP Expert comentat anteriorment ja que es dedica a arribar a la classe que té la responsabilitat per respondre a aquella petició.

En el meu cas, la interfície aplicació es implementada per una classe que realitza la funció d'aquest patró. Si ella no té la responsabilitat de la petició la delega al successor i així successivament.

Imaginem que des de la interfície del usuari es demana la llista de tots els productes, veiem la cadena de responsabilitats amb el codi:

```
namespace PFC
{
   public partial class Frontal_PDA : Form
   {
[…]
IList i=aplicacio.obtenirProductes();
[…]
```

```
namespace PFC.Aplicacio_
{
   interface InterfaceAplicacio
   {
[…]

      //*****************************Obtenim tots els productes dis          s***********************
                IList obtenirProductes();

[…]
```

```
namespace PFC.Aplicacio_
{


   class Aplicacio:InterfaceAplicacio
   {
     […]
      gestor Gestor = new gestor();


         public IList obtenirProductes()
      {
         return Gestor.getProductes();

      }

      […]
```

```
namespace PFC.gestorAplicacio
{
   class gestor
            {

//*****************************Obtenim tots els productes disponibles***********************
      public IList getProductes(){
         IList i = null;
         try
         {
           i= ControlProducte.getInstance().loadProductes();
         }
         catch (Exception ex) { MessageBox.Show(ex.Message,      or",
MessageBoxButtons.OK, MessageBoxIcon.Error); }
         return i;
               }
          […]
```

4.2 Interfícies d'usuari

Un cop va estar tot el sistema d'informació estudiat, analitzat i posteriorment implementat vaig començar a dissenyar les interfícies d'usuari.

Aquestes havien de ser amenes, senzilles d'utilitzar, intuïtives i amb totes les possibles opcions que ofereixia el software a la vista.

A més, al tractar-se de interfícies per pantalles tàctils per una part i per pda's per l'altre havia de pensar la millor forma d'aprofitar aquests elements.

Així doncs, la interfície per les tpv amb pantalla tàctil disposa de una mida de lletres força grossa per no haver de mirar d' aprop tota l'estona cada línia de comanda entrada. També els botons amb els que el personal ha d'interactuar són de mides força grosses per evitar el problema de perdre temps fent punteria amb el dit sobre aquell botó.

El fet d'oferir espais amples aprofitant tota la pantalla, i botons grossos resulta molt còmode sobretot en les terminals de recepció de comanda situades a la cuina i/o a altres llocs d'elaboració on hi hagi aquestes terminals. Està pensat d'aquesta manera per no haver de fer perdre temps al personal en buscar allà on han de picar per marcar que acaben de realitzar una línea de comanda.

D'altra banda, per realitzar la interfície de la pda he hagut d'adaptar la interfície creada primerament per les tàctils.

A part d'haver de reduir la mida, s'ha hagut de pensar en una manera còmoda pel cambrer de fer les comandes.

En una pantalla tàctil és molt senzill marcar els productes que volem i enviar la comanda perquè ho estem visualitzant tot al mateix temps es una sola pantalla. Però en una pda no hi cap ni una sisena part del que hi cap en un monitor de una resolució estàndard de 1024 x 768 píxels.

Degut a la impossibilitat de col·locar tots els controls en una sola pantalla vaig haver de dividir el procés de demanar una comanda en varies parts, és a dir, en varies pantalles de diàleg a fi d'aconseguir el mateix efecte de demanar la comanda en una pantalla tàctil.

Fins ara només he parlat de les interfícies que serveixen per recollir comandes i de les terminals on es reben aquestes. Però a part d'aquestes 3 diferents parts, hi ha també l'apartat d'administració.

La interfície d'administració ofereix les funcionalitats típiques de gestió de dades. Permet donar d'alta, eliminar i modificar totes les dades que formen el sistema, així com els productes, les famílies de productes, etc.

Sistema d'autenticació d'usuari

Per entrar a qualsevol apartat del programa es requereix autenticació del personal. Això significa que es necessita tenir un nom d'usuari i password donat d'alta prèviament al sistema.

Funciona com el típic sistema de loggin, ha de coincidir el nom d'usuari i password amb les dades de la base de dades per poder accedir.

Resolucions

El mateix menú inicial que permet anar a un apartat o un altre del programa, contempla la possibilitat de que pot està essent mostrat en una pda. Per tant des de el principi es mostra la pantalla de selecció en una petita mida i situat a l'extrem superior esquerre.

A part de tenir en compte això i d' haver creat una interfície expressament per pda, el programa s'adapta a varis tipus de resolució:

- 800 x 600 px.: Si al obrir el programa, es detecta una pantalla amb resolució de 800 x 600, les interfícies adopten la forma convenient per poder-se mostrar correctament amb aquesta resolució.

- 1024 x 768 px.i superiors: Si es detecta una configuració de 1024 x 768 px. O superior, les interfícies aprofiten més l'espai i s'adopta una diferent forma que una per un monitor de 800 x 600.

Així dons, tenim un màxim aprofitament de pantalla segons la resolució en aquell monitor actual.

4.2.1 Interfície de l' apartat d'administració

A continuació es mostra una captura de pantalla de l'apartat d'administració.

Des de un principi he seguit els prototips inicials de interfícies,pel que no m'he desviat molt del planejat inicialment. Totes les idees de disseny de la interfície es bassa en agafar idees de altres programes que realitzin tasques similars.

Hi ha varis apartats ben diferenciats:

Barra d'eines

Aquí veiem quatre apartats diferents:

- Aplicació: Funcionalitats d'aquesta aplicació.

  -Inici: Ens torna a la pantalla inicial,la que veiem al principi.
  -Sortir :Tanca l'aplicació.

- Sistema: És l'apartat més important,des de on realitzarem altes, baixes modificacions de les dades de la base de dades.

  -Família productes: Manteniment de les famílies de productes.

  -Productes: Manteniment de productes del restaurant.

  -Personal: Manteniment dels treballadors de l'establiment.

  -Taules: Manteniment de les taules del restaurant.

  -Notes Addicionals: Manteniment de les notes addicionals que poden tenir les famílies de productes.

  -Formes de pagament:Manteniment de totes les formes de pagament que s'admeten.

- Configuració: En aquest apartat es s'afegiran futurament totes les possibilitats de configuració de les terminals entre d'altres. De moment només té la possibilitat de canviar les dades de la connexió al mysql del servidor.

- Gestió cobros: Aquí tenim dos seccions.

  -Cobros pendents: Ens mostra les comandes que no han estat pagades. Hi ha la possibilitat de pagar-la.

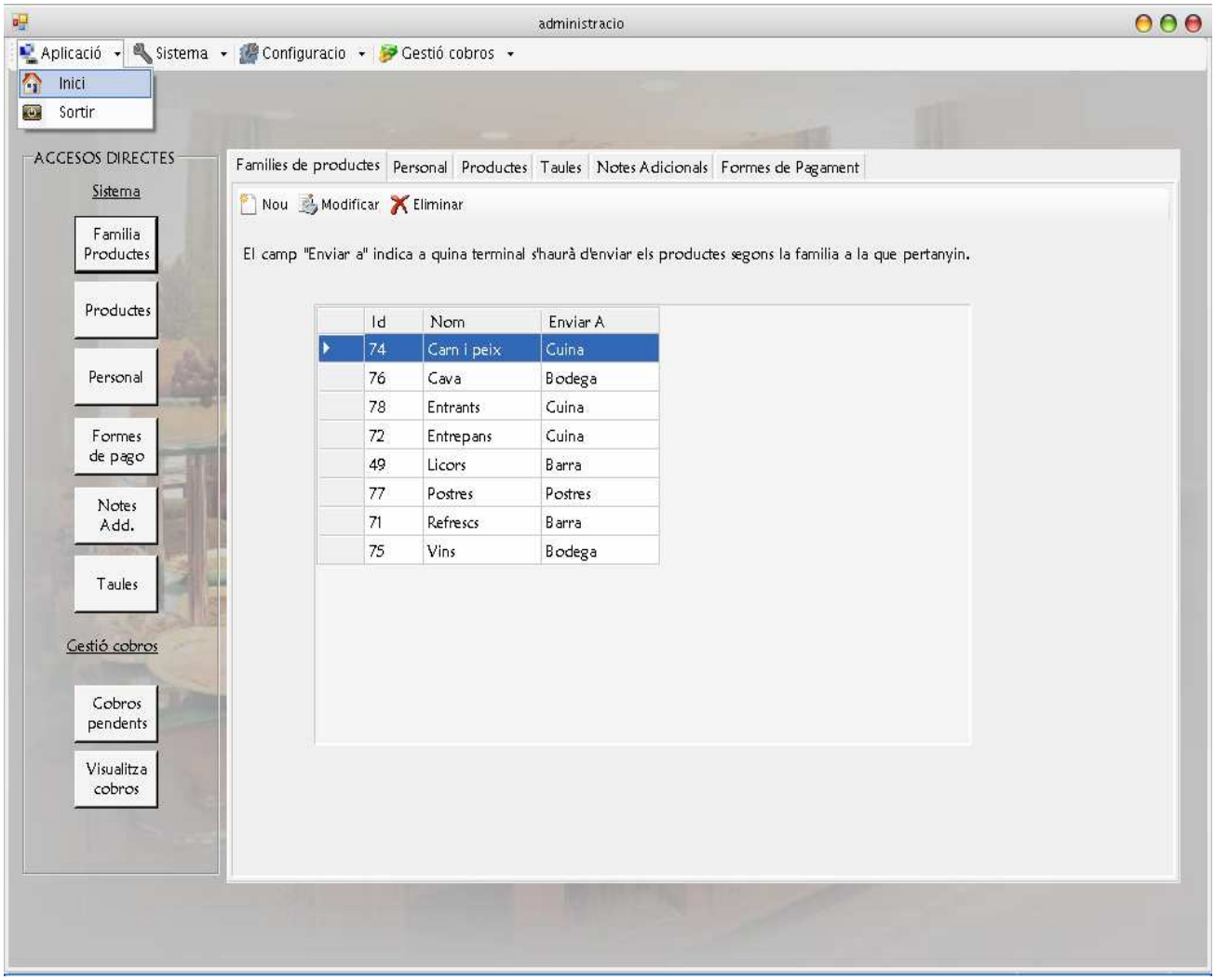
  -Visualitzar cobros: Apareixen uns filtres per obtenir els cobros realitzat per comandes, entre unes dates en concret i tipus de pagament.

Accessos directes

Aquí hi ha un accés ràpid a cada un dels apartats de sistema de dades. Hi ha les opcions que s'utilitzaran amb més freqüència.

Part central

Al centre es carreguen els datagrids corresponents a cada apartat de sistema. Des de aquí podrem donar d'alta , baixa i modificar cada una de les dades del sistema. Ja es veurà com es fa amb detall en la guia del usuari.


administracio
Aplicació
Sistema
Configuracio
Gestió cobros
Inici
Sortir
ACCESOS DIRECTES
Sistema
Familia Productes
Productes
Personal
Formes de pago
Notes Add.
Taules
Gestió cobros
Cobros pendents
Visualitza cobros
Families de productes
Personal
Productes
Taules
Notes Adicionals
Formes de Pagament
Nou
Modificar
Eliminar
El camp "Enviar a" indica a quina terminal s'haurà d'enviar els productes segons la familia a la que pertanyin.
Id
Nom
Enviar A
74
Carn i peix
Cuina
76
Cava
Bodega
78
Entrants
Cuina
72
Entrepans
Cuina
49
Licors
Barra
77
Postres
Postres
71
Refrescs
Barra
75
Vins
Bodega

4.2.2 Interfície del frontal de vendes per pantalla tàctil

Aquesta interfície es divideix en 3 grans blocs.

Marge esquerra

Aquí apareixen uns botons amb varies funcionalitats. Encara que estiguin sempre presents, només realitzaran la seva feina quan realment sigui possible.

Per exemple, el botó cobrar, només anirà quan la comanda estigui enviada, servida i en estat pendent de cobrar (color vermell).

Bloc centre-esquerre

Aquest bloc serveix per 2 coses diferents.

Per una part, es mostren totes les taules disponibles amb el seu estat. Quan una taula és seleccionada, es carrega un altre bloc sobre d'aquest, el qual permet escollir els productes que formalitzaran la comanda en aquesta taula escollida.

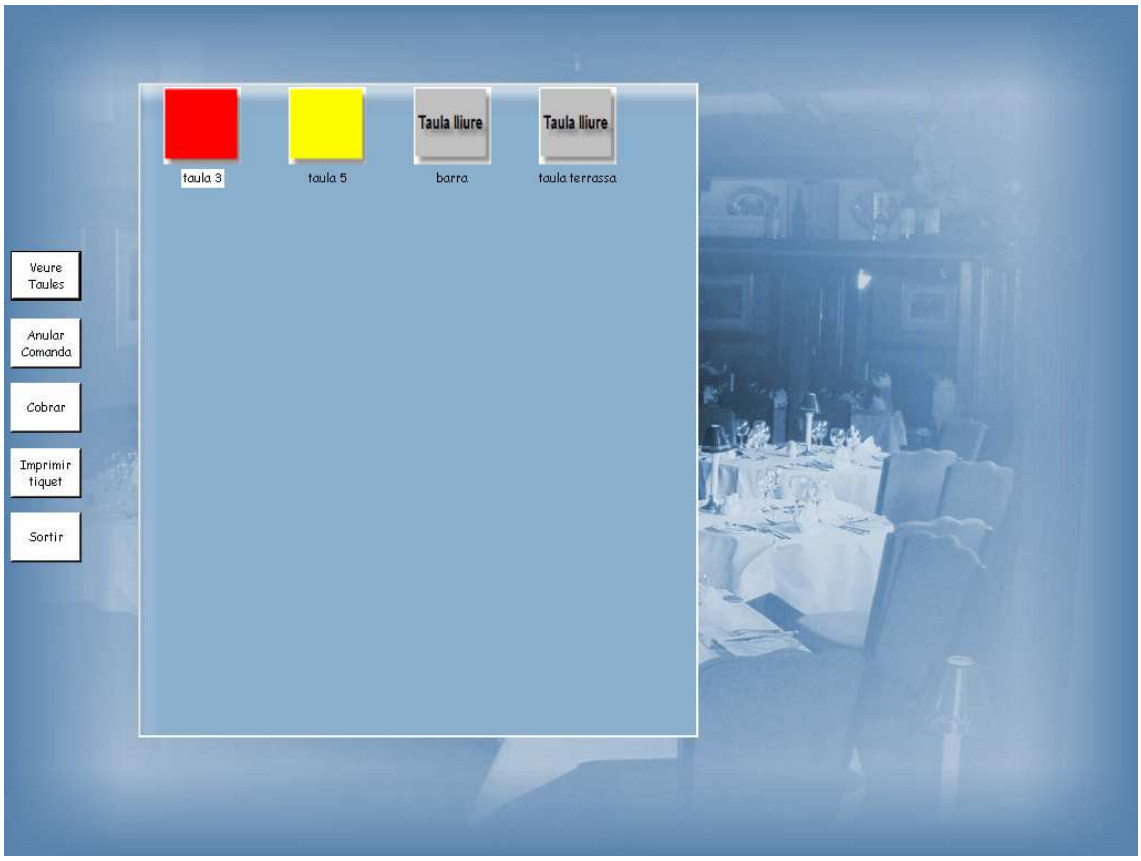
Un cop escollit el producte que desitgen , es carrega una nova línea de comanda en el bloc de detall de comanda.

Bloc dret, detall de comanda

Aquí apareixen totes les línees de comanda d'aquella comanda, així com les funcionalitat per poder canviar les quantitats de cada línea, refrescar les línies , canviar el total de la comanda ,et c.

Aquí també hi ha el botó que permet enviar la comanda a la seva terminal corresponent.

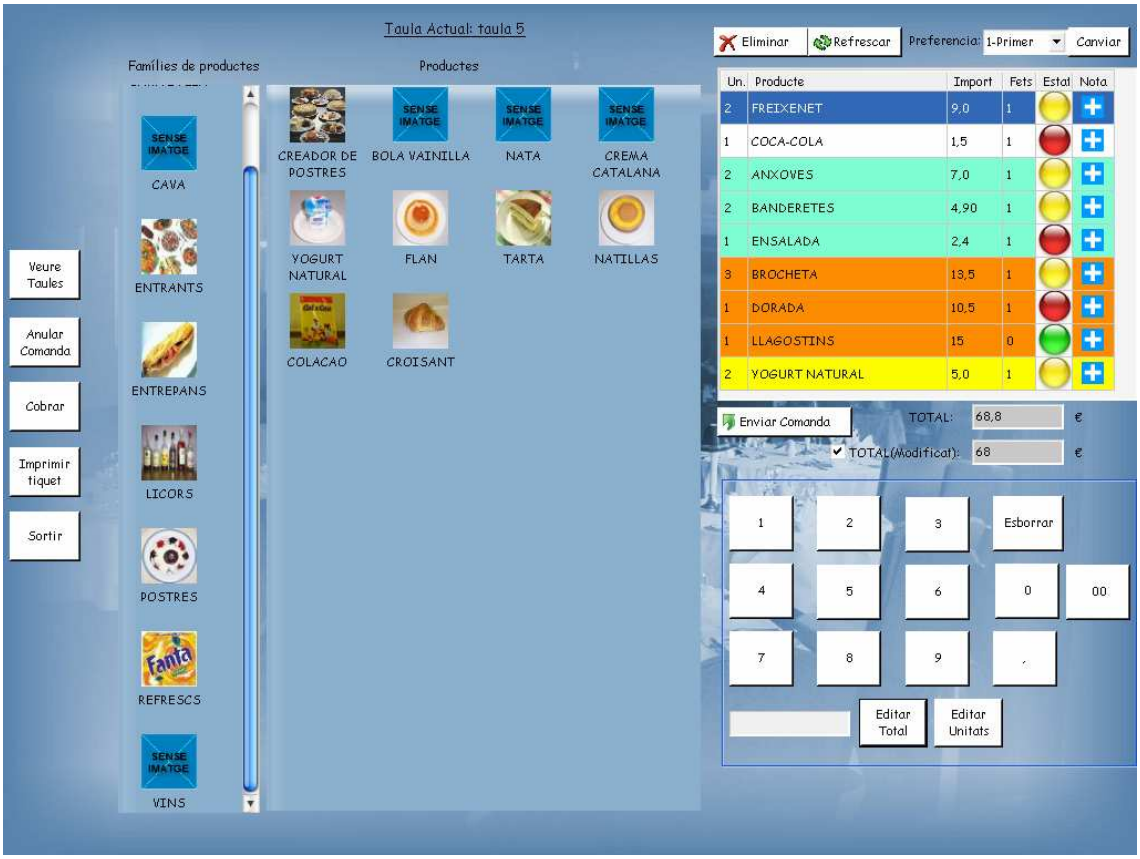
Visualització de taules



Aquesta es la primera pantalla que veiem al entrar a l'apartat de vendes. Aquí escollim la taula que volem.

En la imatge veiem que hi ha una taula en estat pendent de cobrar. Més endavant veurem tots els estats possibles.

Frontal Vendes



Un cop seleccionada la taula ja veiem tota la resta del panell. Des de la elecció dels productes a editatge de cada línia de comanda.

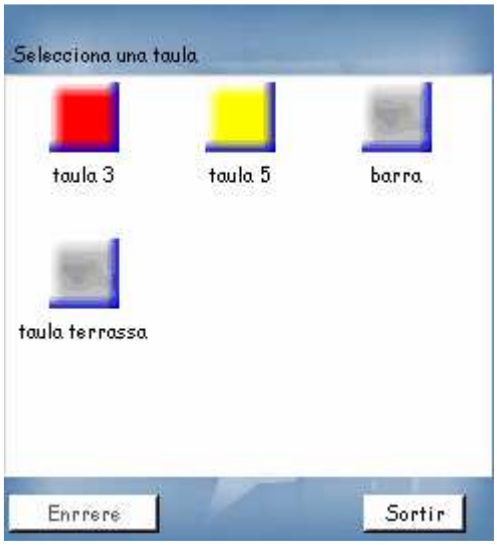
4.2.3 Interfície del frontal de vendes per PDA

Aquesta interfície parteix de la mateixa que ja he creat per el frontal de venda per tpv.

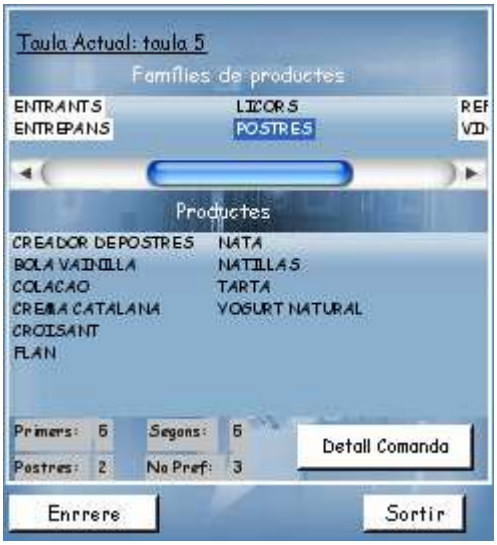
A part de reduir tots els controls i components, s'han de seguir varis passos abans de realitzar la comanda.

Cada pas es realitza en un bloc carregat sobre tota l'amplitud de la pantalla.

Selecció de taula



Selecció de producte



Selecció de quantitat i preferència del producte escollit

Detall comanda

4.2.4 Interfície de les terminals de recepció de comanda

Aquesta interfície està pensada per ser totalment còmoda en un lloc tan complicat com en una cuina. Això significa utilitzar una font de lletra de mida gran, i uns botons amb els que interactuar notablement grans.

Hi ha només 2 apartats:

Visualització de taules

Al igual que les 2 últimes interfícies ja comentades, aquí també apareix la possibilitat d'escollir la taula sobre la que volem fer el tractament de la comanda. Per tant, se'ns mostren les taules disponibles amb el color associat indicant l'estat.

Línies de comanda

Des de aquest bloc on es mostren totes les línies, hi ha la possibilitat de afegir o restar productes acabats. L'objectiu es elaborar tants productes per línia com indica el primer camp(unitats). Un cop estan tots realitzats, l'estat de la línia passa a finalitzat.

Tal com he comentat anteriorment, els botons d'afegir i treure, intenten ser tant grans com ha estat possible, així com l'amplada de la fila i mida de la lletra.

Primer escollim el terminal que volem

Terminal de tots els productes

4.3 Futures millores

El desenvolupament d'un software és un procés llarg i sense un final concret. Les possibilitats d'ampliació d'aquest projecte son molt grans.

Una possible millora per facilitar la gestió de les taules, seria el de crear una secció que permetés configurar la disposició de les taules en el restaurant, és a dir, una manera gràfica i intuïtiva en que l'usuari pogués situar cada taula allà on vol dins un quadre definit que seria el perímetre de l'establiment per identificar de manera més ràpida la taula allà on està servint la comanda.

D'altra banda, també es podria crear un apartat d l'àrea de gestió que permetés exportar les factures i tots els moviments de caixa en general en fitxers xml per tal de que poguessin ser utilitzades per softwares de comptabilitat o facturació.

Una ampliació força significativa seria afegir el control d'estoc entrant i sortint per dur un major control sobre les existències en el restaurant.

Finalment, el programa també podria oferir una secció d'administració remota. És a dir, aprofitant la tecnologia .net i la connectivitat d'aplicacions via internet, aconseguir administrar el restaurant des de casa sense necessitat d'haver d'estar davant del servidor del local.

A part d'aquestes millores i moltes altres que es podrien anar desenvolupant al llarg del temps, també es podria dedicar més temps a crear una interfície més amigable amb possibilitats de events de arrastrar i similars.

5. CONCLUSIONS

L'objectiu d'aquest projecte era oferir un sistema informàtic per el control de comandes en restaurants de la manera més senzilla i al·legant possible.

Aquest projecte està destinat a qualsevol negoci relacionat amb la restauració. És totalment adaptable tant per un gran restaurant com per un petit bar.

Els objectius marcats inicialment han estat assolits satisfactòriament, aconseguint informatitzar per complet el sistema de comandes de restaurants i facilitant la feina a tots els treballadors que disposin d'aquest.

Un dels principals objectius acadèmics era el de saber desenvolupar software en varis llenguatges i tecnologies de programació. Finalment s'ha utilitzat el llenguatge c# de la tecnologia .net que tant d'èxit està tenint en el món del desenvolupament del software actual.

He aprofitat els recursos d'aquesta tecnologia per tractar el llenguatge XML, destinat sobretot a fitxers de configuració de l'aplicació.

A més, l' ús de nhibernate, motor de persistència per programació orientada a objectes i bases de dades relacionals, ha estat una metodologia fortament utilitzada al llarg de tota la implementació, ja que la base del sistema informàtic resideix sobre aquest.

Un altre objectiu principal era el de aprofitar tots els recursos i tècniques apresses durant la formació acadèmica i aplicar-ho en aquest projecte. Em refereixo a la orientació a objectes, tècniques de modelatge i estandarditzación del software i el model relacional entre molts altres.

Hi ha una fort component de enginyeria del software basat en el modelat de un llenguatge unificat (UML) amb el que es pretén crear un software que segueixi uns estàndards i sobretot, un software el qual pugui ser totalment reutilitzable, ampliable i fàcil de mantenir. Aquests tres conceptes són la base principal sobre la que es fonamenta una bona enginyeria del programari, uns conceptes recalcats al llarg de tots els anys de formació acadèmica i totalment aplicats en aquest projecte.

Per realitzar l'anàlisi del software he utilitzat eines CASE com Poseidon la qual ha ajudat molt en la autogeneració de codi de les classes principals.

Amb aquest projecte he après a aplicar a la pràctica els patrons de disseny que ens ofereix el UML. Així dons he solucionat molts problemes de estandarditzación i optimització de codi amb patrons Gof i GRASP. Concretament s'han utilitzat els següents patrons:

- Patró Expert
- Patró Creator
- Patró High Cohesion (Alta cohesió)

-Patró Low Coupling (Baix acoblament)
-Patró Controlador
-Patró Pure fabrication (Pura fabricació)
-Patró singleton (instància única)
-Patró Factory Method
-Patró Façade
-Patró Chain of responsability (Cadena de responsabilitat)

He seguit totalment la filosofia de la enginyeria del software que ens diu que el desenvolupament de un software està compost per les quatre grans fases de anàlisi, disseny, implementació i proves i refinament. A més el desenvolupament del projecte ha estat un procés iteratiu i incremental el qual ha anat patint canvis al llarg de la seva creació, tal com es defineix tota creació de un programari ben enginyat.

El concepte de crear interfícies amigables i amb una gran usabilitat ha estat molt present en tot moment. Aquest concepte engloba una sèrie de paràmetres que donen com a resultat unes interfícies molt fàcils d'usar i intuïtives. Les característiques són les següents:

- Les interfícies s'han de poder entendre per l' usuari amb facilitat.
- Han de ser intuïtives, oferint cada cosa on s' espera que estigui, de tal manera que la realització de les tasques es produeixi amb una successió de continuïtat lògica i senzilla.
- Facilitar l' aprenentatge de tal forma que l' usuari sigui capaç d'entendre i assolir els conceptes de l'aplicació.
- Ha de ser agradable estèticament.

Inicialment pretenia realitzar una aplicació per la PDA amb un llenguatge específic per ella. Vaig estudiar el Visual c# smart device el qual utilitza una versió més comprimida del framework .net (principal motor del .net) anomenat Compact framework .net. Però aquest no suportava les llibreries de nhibernate, pel que vaig haver de deixar de banda aquesta opció.

Finalment, com s'ha comentat al llarg de la memòria, vaig implementar la interfície de PDA aprofitant tot el codi ja creat i utilitzant el servei de terminal service del windows CE de les PDA per obtenir una sessió remota al servidor.

El fet d' haver treballat utilitzant tots els coneixements adquirits i utilitzant eines i llenguatges que desconeixia m' ha ajudat a fer-me una imatge sobre l' entorn professional en el àmbit informàtic.

La realització d' aquest projecte ha estat una satisfacció personal, degut a que gràcies als coneixements aplicats he pogut materialitzar una idea que tenia en ment des de feia temps. A més el fet de saber-me moure amb moltes tecnologies, llenguatges o tècniques és una cosa que em satisfà com a estudiant i futur professional, ja que amb la gran i constant evolució tecnològica en el àmbit informàtic aquest serà el nostre pa de cada dia.

6. BIBLIOGRAFIA

Els llibres utilitzats són els següents:

- El lenguaje unificado de modelado. G**. Booch, J. Rumbaugh, I. Jacobson**
  Editorial: Addison-Wesley.

- El proceso unificado de desarrollo de software. G**. Booch, J. Rumbaugh, I. Jacobson**
  Editorial: Addison-Wesley.

- UML y Patrones. Segunda Edición. **Craig Larman**
  Editorial: Addison-Wesley.

- Patrones de diseño : elementos de software orientado a objetos reutilizable. **Gamma, Erich**
  Editorial: Pearson-Adison-Wesley

Les pàgines web més rellevants són:

- http://www.c-sharpcorner.com/

- http://www.javahispano.org/

- http://mcabrera.datacenter1.com/articles/dotNET/nhibernate/

- http://www.cs.queensu.ca/Software-Engineering/toolcat.html

- http://dariodotnet.blogspot.com/2005/10/tutoriales-de-nhibernate.html

- http://www.hibernate.org

7. ANNEX

ANNEX 1: NORMES DE DOCUMENTACIÓ

COS DEL DOCUMENT

Tipus lletra : Arial
Mida de la lletra: 12, justificat
Paraules rellevants: Negreta

Separació entre títols , subtítols i subapartats: un salt de línia.
Separació entre subapartats-text-subapartats: dos salts de línia.
Separació entre paràgrafs: dos salts de línia.
Separació entre el marge esquerra per a subapartats: un tabulador.

Títols

Tipus de lletra : Arial
Mida de la lletra: 16
Estil: Subratllat, negreta i majúscula

Subtítols

Tipus de lletra : Arial
Mida de la lletra: 12
Estil: Negreta i majúscula

Subapartats

Tipus de lletra : Arial
Mida de la lletra: 12
Estil: Negreta

ANNEX 2: CONTINGUT DEL CD-ROM

El CD conté les següents dades:

- Carpeta Instal·lació : El fitxer “setup.msi” és l’instal·lador de l’aplicació.
- Carpeta Documentació: Hi ha els documents lliurats en paper, però en format pdf.
- Carpeta codi font: Hi ha tot el codi font i fitxer de projecte.
- Framework 2.0.exe : És el fitxer instal·lador del framework.
- Fitxer “llegeixme.txt” : Hi ha una nota sobre el correcte funcionament del programa.