Aplicació J2EE per la Gestió de Carteres i Anàlisis Tècnic de Valors Bursátils

Josep Mª Martínez Bort
ETIS

Albert Grau Perisé

16 de Juny de 2006

A Esther, Maria i família,

Gràcies pel vostre suport, comprensió i paciència en les innombrables hores dedicades a la realització d'aquest projecte i durant tota la carrera.

Agraïments:

A tota la comunitat de l'UOC, gràcies!. Gràcies per un projecte educatiu de tan alta qualitat. Només tinc paraules d'agraïment per a tots els consultors que he tingut durant la carrera, per la bona preparació dels materials, i per tot el personal amb els qual he tingut tracte durant tots aquest anys, no em vull oblidar del Miguel que en els moments difícils sempre m'ha animat a continuar endavant, ànim!! tu també ho aconseguires.

Molt especialment, gràcies a l'Albert Grau, per la seva paciència i els seus bons consells en la realització d'aquest TFC.

Resum de Treball Final de Carrera

Aquest treball final de carrera es basa en l'anàlisi, disseny i implementació d'una aplicació que permet als usuaris gestionar carteres de valors, així com, utilitzar eines gràfiques per realitzar anàlisis tècnics dels valors bursátils, l'àmbit de valors serà el Mercat Continuo de la societat "Bolsas y Mercados Españoles" (BME) i els principals índex mundials (IBEX-35, DOW JONES, NASDAQ, EURO STOXX, CAC-40, DAX XETRA, FOOTSIE-100,...).

L'anàlisi i disseny del projecte s'inscriu en el paradigma de l'orientació a objectes, en un marc de comunicació entre l'aplicació i els proveïdors de dades bursátils basada en la xarxa Internet. Un aspecte especialment interessant ha estat la necessitat d'utilitzar una arquitectura distribuïda per comunicar, no només les diferents components de l'aplicació, sinó també els diversos proveïdors de dades de què disposa l'aplicació

La implementació s'ha basat en el llenguatge Java, i per a fer possibles tots els requeriments funcionals de l'aplicació, s'ha optat per utilitzar l'arquitectura J2EE. El conjunt de tecnologies agrupades sobre **J2EE** ens ha permès realitzar una aplicació robusta, escalable i altament reutilitzable, característiques avui en dia imprescindibles en qualsevol aplicació empresarial a Internet.

Finalment, un aspecte que m'ha agradaria remarcar d'aquest TFC és la utilització de diversos **patrons de disseny i *frameworks***. En la realització d'aquest projecte s'ha utilitzat el patró MVC/ Model 2 a partir de la utilització del bastiment **Struts**. La utilització d'aquest bastiment ha permès realitzar un producte amb una clara separació de funcionalitats entre components, amb la finalitat de desacoblar el codi d'accés a les dades, el codi de la lògica de negoci (implementada en **EJBs**) i el codi de presentació. Tot això ha permès un alt grau de reutilització i reducció de codi duplicat, que fa que l'aplicació web sigui molt senzilla de mantenir, reutilitzar o modificar (tant dels components, com del disseny d'interfície).

Índex

Agraïments: 2
Resum de Treball Final de Carrera 3
Índex 4
1. INTRODUCCIÓ. 6
1.1 Justificació del TFC: punt de partida i aportació. 6
1.2 Objectius del TFC. 7
1.3 Enfocament i mètode seguit. 7
1.4 Planificació del projecte. 7
1.5 Productes obtinguts. 9
1.6 Descripció de la resta de capítols de la memòria. 9
2. ESPECIFICACIÓ I ANÀLISI DELS REQUERIMENTS 10
2.1 Introducció. 10
2.2 Descripció del Projecte. 10
2.3 Composició del programari 11
2.4 Diagrama de paquets i subsistemes. 12
2.5 Especificacions de les funcionalitats per subsistema 13
2.6 Casos d'ús. 15
2.7 Requisits del maquinari i programari. 27
2.8 Seguretat. 28
2.9 Funcionalitats de les properes versions. 28
2.10 Recollida i documentació dels requisits de UI. 29
3. DISSENY. 30
3.1 Introducció. 30
3.2 Java 2 Enterprise Edition (J2EE). 30
3.3 Disseny Arquitectònic. 31
3.4 Diagrama estàtic de disseny: classes i jerarquies 39
3.5 Diagrames de seqüència per subsistema. 40
3.6 Disseny de la persistència. 46
3.7 Disseny de la interfície d'usuari. 48
4. IMPLEMENTACIÓ. 51
4.1 Requeriments de programari. 51
4.2 Eines de desenvolupament. 51
5. VALORACIÓ ECONÒMICA. 52
6. CONCLUSIONS. 52
7. GLOSSARI. 53
8. BIBLIOGRAFIA CONSULTADA 54

9. ANNEX 1. SENTÈNCIES SQL 55

1. INTRODUCCIÓ.

1.1 Justificació del TFC: punt de partida i aportació.

Aquest TFC parteix de la necessitat de desenvolupar una aplicació web que presenti la característica de ser escalable, robusta i reutilitzable, dins del paradigma de la orientació a objectes i que permeti una estructura distribuïda, tant de components com de sistemes d'informació.

Com a qualsevol TFC, el objectiu principal d'aquest és mostrar l'assoliment de l'aprenentatge que s'ha dut a terme al llarg dels estudis d' Enginyeria Tècnica en Informàtica de Sistemes, analitzant un problema complex de tipus pràctic transformant-lo en un projecte informàtic, elaborant un pla de treball i treballant els aspectes formals en el desenvolupament (orientat a objectes en aquest cas), i finalment presentant una solució fiable i eficient.

El punt de partida són doncs els coneixements adquirits durant la carrera. Aquí apareix una característica singular d'aquest TFC; la necessitat d'un aprenentatge de coneixements nous que no es presenten duran la carrera d'enginyeria tècnica, referents a programació distribuïda, components, patrons (es veuen molt per sobre a Enginyeria de programari), etc.

Una dificultat afegida ha estat l'esforç que ha calgut realitzar, no només en l'assoliment de coneixement referents a programació distribuïda, patrons i bones pràctiques, sinó molt especialment a les característiques de la tecnologia escollida per a poder-ho implementar.

Així sí el punt de partida eren els coneixements adquirits en assignatures com programació OO, enginyeria del programari, o tècniques de desenvolupament de programari, el punt final, tal i com esperem quedi de manifest en la present memòria, ha estat l'aprenentatge d'un conjunt de tecnologies relacionades amb la programació distribuïda i concretament en el desenvolupament d'aplicacions web empresarials.

L'aportació característica d'aquest TFC aplicat a la tecnologia J2EE, com a arquitectura per donar resposta a les necessitats anteriorment exposades, ha estat combinar tots els coneixements teòrics amb una implementació que utilitza fonamentalment el patró MVC model 2 (patró adaptat a les característiques de les aplicacions web).

Concretament s'ha fet servir el *framework* Struts (la seva justificació s'exposa més endavant). Aquest TFC fa servir el bastiment Struts en les capes client i web, però utilitza EJB en la capa model. Els avantatges de tota aquesta arquitectura s'ha aniran exposant durant el desenvolupament de la memòria.

1.2 Objectius del TFC.

Els principals objectius del TFC ha estat l'anàlisi, disseny i implementació d'una aplicació web per la gestió de carteres i anàlisi tècnic de valors bursàtil, utilitzant la tecnologia J2EE.

He aquest sentit he après els avantatges i les característiques del desenvolupament d'aplicacions multicapa i distribuïdes i la utilització de les tecnologies que s'agrupen sota J2EE. A més de conèixer les característiques de l'arquitectura J2EE, un dels objectius més importants ha estat treballar amb tecnologies com JSPs, servlets, EJBs, JDBC, JNDI, etc.

A més, el treball final de carrera s'ha fixat com a objectiu la utilització de EJB en la capa model, però la utilització del patró MVC/2 a través d'un *framework* com Struts.

1.3 Enfocament i mètode seguit.

L'enfocament utilitzat en el desenvolupament del TFC ha estat treballar en paral·lel diversos aspectes. Per una banda realitzar l'anàlisi i disseny de l'aplicació mitjançant el paradigma de l'OO, des d'un punt de vista tant estàtic com dinàmic. Però per una altra, avançant en la corba d'aprenentatge de totes les tecnologies relacionades inicialment amb J2EE i posteriorment amb Struts.

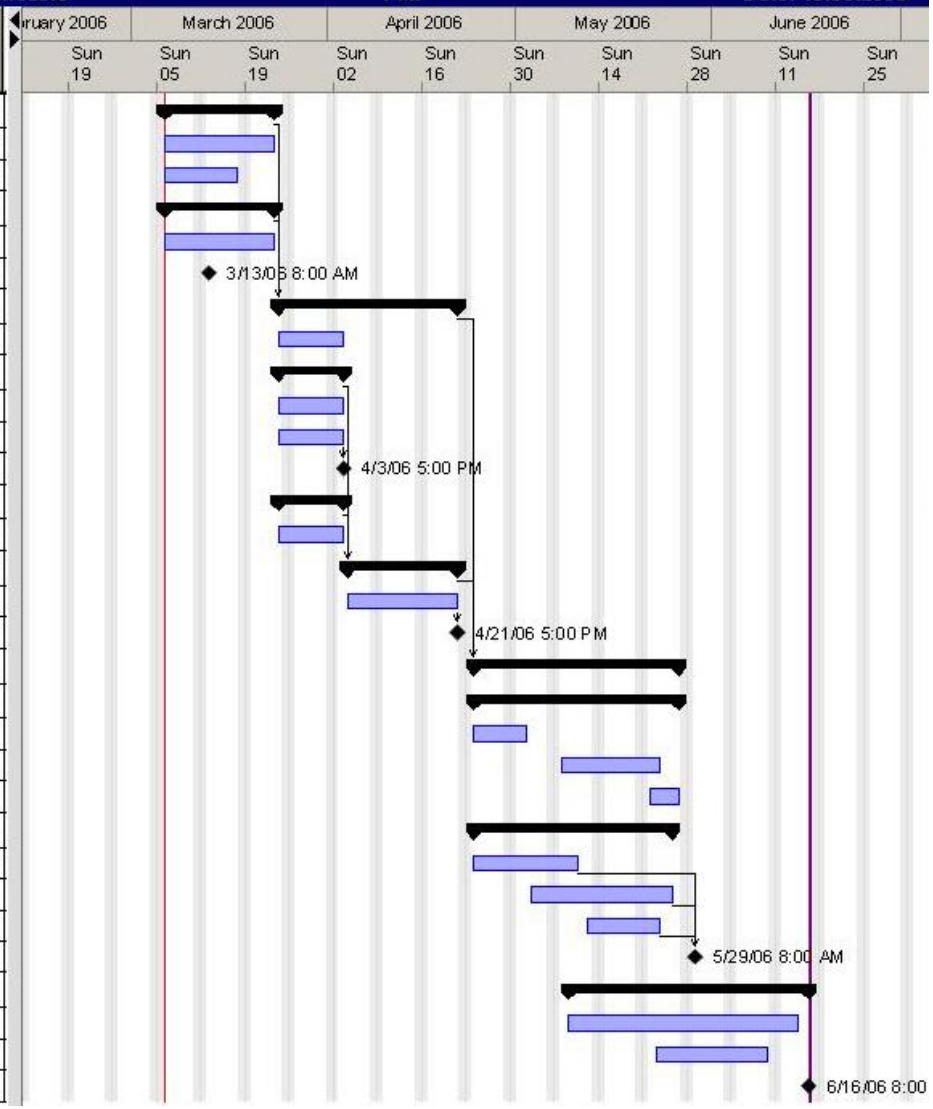
Aquesta ha estat la dificultat principal del projecte. La gran quantitat de coneixements nous a adquirir, i la seva aplicació a un projecte real. Si l'aprenentatge de l'arquitectura i tecnologies J2EE és dificultosa, encara ho és més si cal combinar-ho amb l'aprenentatge de marcs de treball (Struts), sobre tot quan **partia de cap coneixement** sobre J2EE.

1.4 Planificació del projecte.

Amb respecte de la planificació inicial els principals incompliments s'han produït a l'etapa d'implementació i testing. He hagut de modificar la planificació per a poder realitzar part del testing a mesura que anava finalitzant la implementació dels diversos subsistemes de l'aplicació.

00002 Afinion Project Viewer Free Version — **Project: Planificació** — **PM:** — **Date: 16.06.2006**

Id of	Name	Duratior	Start	Finish	Total Sla
1	⊟ Fase d'Inici del projecte	14d	06.03.2006	23.03.2006	22d
2	Definició de Requeriments	14d	06.03.2006	23.03.2006	14d
3	Planificació del projecte	10d	06.03.2006	17.03.2006	18d
4	⊟ Anàlisi	14d	06.03.2006	23.03.2006	22d
5	Identificació casos d'us generals	14d	06.03.2006	23.03.2006	22d
6	PAC 1. Planificació prèvia	0d	13.03.2006	13.03.2006	55d
7	⊟ Fase d'elaboració	21d	24.03.2006	21.04.2006	22d
8	Instal.lacio software de base.	7d	24.03.2006	03.04.2006	36d
9	⊟ Anàlisi	7d	24.03.2006	03.04.2006	22d
10	Especificació de casos d'us d'usuari	7d	24.03.2006	03.04.2006	22d
11	Diagrama de clases	7d	24.03.2006	03.04.2006	22d
12	Entrega d'Anàlisi	0d	03.04.2006	03.04.2006	22d
13	⊟ Disseny	7d	24.03.2006	03.04.2006	22d
14	Disseny de l'arquitectura	7d	24.03.2006	03.04.2006	22d
15	⊟ Implementació	14d	04.04.2006	21.04.2006	22d
16	Implementació de les classes d'entitat i manejadors de	14d	04.04.2006	21.04.2006	22d
17	PAC 2 Lliurament de l'anàlisi i el disseny	0d	21.04.2006	21.04.2006	25d
18	⊟ Fase de Construcció	25d	24.04.2006	26.05.2006	14d
19	⊟ Disseny	25d	24.04.2006	26.05.2006	14d
20	Especificació dels casos d'us d'usuari	7d	24.04.2006	02.05.2006	18d
21	Especificació casos d'us d'inversor	12d	08.05.2006	23.05.2006	17d
22	Especificació de casos d'ús d'administrador	5d	22.05.2006	26.05.2006	14d
23	⊟ Implementació	24d	24.04.2006	25.05.2006	15d
24	Desenvolupament dels casos d'us d'usuari	13d	24.04.2006	10.05.2006	26d
25	Desenvolupament casos d'us d'inversor	17d	03.05.2006	25.05.2006	15d
26	Desenvolupament casos d'us d'administrador	8d	12.05.2006	23.05.2006	17d
27	PAC 3. Implementació	0d	29.05.2006	29.05.2006	14d
28	⊟ Transició	28d	09.05.2006	16.06.2006	0d
29	Memòria del projecte	27d	09.05.2006	14.06.2006	1d
30	Test de funcionament de l'aplicació.	14d	23.05.2006	09.06.2006	4d
31	16 de Juny Lliurament final	0d	16.06.2006	16.06.2006	0d



1.5 Productes obtinguts.

El producte obtingut en la implementació d'aquest projecte ha estat una aplicació web realitzada amb tecnologia J2EE.

Es descompon en els següent elements:

- fitxer de distribució .ear que inclou el .war de la capa web +.jar de les classes de la lògica de negoci.
- El codi font de totes les classes utilitzades
- Els fitxers de desplegament i de configuració utilitzats, conjuntament amb les llibreries necessàries pel seu funcionament.
- El fitxers de creació de les taules i la inserció de dades necessàries a la BD a utilitzar.

1.6 Descripció de la resta de capítols de la memòria.

En la resta de capítols de la memòria es comentaran les fases d'anàlisi, disseny i implementació per les quals ha passat el desenvolupament d'aquest TFC.

2. ESPECIFICACIÓ I ANÀLISI DELS REQUERIMENTS

2.1 Introducció.

Aquest apartat recull i documenta, detalladament, les necessitats d'informació que haurà de resoldre l'aplicació de gestió de carteres i anàlisis tècnic de valors bursátils BORSA-X_{TRADER}. Per tant el seu objectiu es definir detalladament les necessitats d'informació que haurà de resoldre el programari. S'inclou l'anàlisi necessària per a poder traduir els requisits en especificacions més formals (diagrames d'UML) que facilitin posteriorment el desenvolupament de l'etapa de disseny, així com la identificació de les classes fonamentals i l'expressió dels casos d'ús.[1]

Com es podrà desprendre de la seva lectura, es comença efectuant una aproximació general al sistema mitjançant una anàlisi global d'aquest, per després analitzar el sistema en subsistemes.

2.2 Descripció del Projecte

BORSA-X_{TRADER} permet als usuaris gestionar carteres de valors, així com, utilitzar eines gràfiques per realitzar anàlisis tècnics dels valors bursátils, l'àmbit de valors serà el Mercat Continu de la societat "Bolsas y Mercados Españoles" (BME) i els principals índex mundials (IBEX-35, DOW JONES, NASDAQ, EURO STOXX, CAC-40, DAX XETRA, FOOTSIE-100,...).

Les dades d'aquests valors (accions i índexs) seran subministrades per pàgines d'Internet de forma automàtica per l'usuari.

La interfície serà web, es a dir, s'accedirà a l'aplicació a través d'un navegador, així mateix, la interfície estarà dissenyada de manera clara i amigable, i la navegació en l'aplicació serà intuïtiva.

L'aplicació estarà configura en un servidor d'aplicacions i qualsevol que tingui accés i permisos podrà executar l'aplicació

La funcionalitat de l'aplicació permetrà als usuaris les següents prestacions:

- Visualització en temps real dels 35 valors de l'IBEX-35.
- Top dels cinc millors inversors registrats
- Gestió de carteres
- Utilització d'eines gràfiques per l'análisis tècnic dels valors
- Configuració d'alarmes
- Visualització gràfica de tots els valors de l'àmbit de l'aplicació

Com s'ha comentat a la introducció l'accés a l'aplicació es farà a través d'un navegador, aquesta primera plana web presentarà informacions dinàmiques obtingudes de la base de dades de l'aplicació com:

- Visualització en temps real, format taula, dels 35 valors de l'IBEX-35 (preu actual, màxim, mínim, % de variació respecte al dia anterior).

- Top dels cinc millors inversors registrats.

- Top dels millors i pitjors accions del IBEX-35 del dia anterior.

[1] CAMPDERRICH, BENET. *Enginyeria del programari; Anàlisi orientada a objectes* . UOC.

- Accés per login i password dels diferents rols d'usuaris

Totes aquestes prestacions podem ser explorades per qualsevol usuari, a més els inversors (usuaris registrats) podem gestionar crear, modificar i eliminar diferents carteres de valors. Una vegada accedeixi al seu espai se li mostrarà el llistat de les diferents carteres. Tindrà la possibilitat de crear un màxim de 10 carteres i a cada cartera podrà afegir un màxim de 35 accions, igualment, podrà afegir (compra) i eliminar (venta) accions de la cartera seleccionada.

Per configurar una cartera, se li donarà un nom i se li afegiran accions, d'aquestes s'indicarà el nombre de títols, la data de compra i el preu de compra, automàticament l'aplicació calcularà la rendibilitat en funció de l'últim preu, quan es tanqui una posició, total o parcial, el inversor indicarà el nombre de títols, la data i el preu de venda.

Una de les funcionalitats més importants de l'aplicació és la seva eina gràfica, amb ella els inversors podran visualitzar el gràfic històric de preus d'un valor, amb barres o espelmes (candlesticks), igualment, podran modificar l'espai temporal de visualització a dies, setmanes o mesos; el eix vertical mostrarà els preus i l'horitzontal el temps.

Una funcionalitat molt reclamada pels inversors en borsa que no podem estar contínuament veient-la, es el sistema d'alertes, pel qual l'inversor configurarà sobre un, o més, valors un llindar de preu que al ser superat per dalt o per baix, obligarà a l'aplicació a l'enviament d'un e-mail, indicant aquest fet.

A la borsa les empreses no son perennes i els índex els modifiquen cada sis mesos, per això, serà responsabilitat de l'administrador l'actualització de les empreses del IBEX-35 i Mercat Continu i configuració dels índexs.

Per obtindrà les dades bursátils, l'administrador donarà d'alta el sistema url's de proveïdors de dades i el sistema anirà a aquests proveïdors cada cert temps, 5 minuts per exemple, a agafar les dades mitjançant fitxers XML.

2.3 Composició del programari

A nivell de **funcionalitats** podem distingir 5 subsistemes:

- **Subsistema d'administració:** Aquest subsistema inclou totes les funcionalitats que permeten gestionar l'alta, baixa i modificació d'índexs, empreses i proveïdors de dades, així com, llistat tots el inversors donats d'alta al sistema. És competència dels usuaris administradors.

- **Subsistema d'informació bàsica:** Es la pàgina principal de la web BORSA-X$_{\text{TRADER}}$ des d'aquí es podrà veure informació del valors del IBEX-35, top dels millors inversors i top dels millors i pitjors valors del Mercat Continuo, igualment, permetrà a l'usuari donar-se d'alta com Inversor.

- **Subsistema de gestió de carteres:** Aquest paquet l'hem dividit en dos paquets de servei:

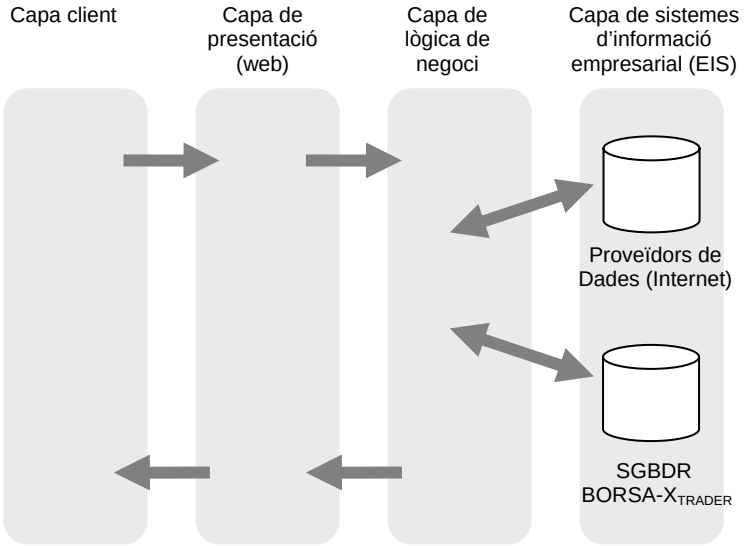
    i. **Eines Gràfiques:** Permet totes les funcionalitats pròpies de l'anàlisi tècnic de valors.

    ii. **Carteres:** Inclou totes les funcionalitats que permeten gestionar carteres de valors, com són, crear carteres, eliminar-les, afegir operacions, cercar empreses per veure les seves dades, afegir alertes, llistar operacions i actualitzar les dades de les empreses.

- **Subsistema de Dades Bursàtils:** S'encarrega de tota la part d'obtenció de les dades bursátils mitjançant bases de dades externes (Internet) i desar-les a la base de dades del sistema.

- **Subsistema de connexió i seguretat:** aquest subsistema és al que tenen accés els usuaris quan accedeixen al sistema. Realitza tasques d'autentificació, registre i funcions de seguretat.

A nivell **d'arquitectura**, el model de desenvolupament d'aplicacions es basa en la separació en capes (arquitectura multicapa). El nombre i tipus de capes utilitzades en aquest model són,



2.4 Diagrama de paquets i subsistemes.

Per facilitar la tasca d'anàlisi, disseny i implementació, el sistema s'ha subdividit en diversos subsistemes (veure apartat composició del programari). Atenent al grau de dependència dels casos d'ús que veurem posteriorment podem distingir els següents paquets d'anàlisi:

- Gestió de informació bàsica, compren els casos d'ús corresponents a l'alta com Inversor i la pàgina principal de la web BORSA-XTRADER

- Gestió de Carteres contempla els casos d'ús relatius a la creació, modificació i eliminació de carteres, així com, eines gràfiques per l'anàlisi tècnic d'empreses.

- Connexió: tasques relacionades amb la connexió i validació al sistema.

- Administració: casos d'ús relatius a la gestió d'empreses, índexs, proveïdors de dades i llistat de inversors.

- Dades Bursátils: casos d'ús relatius a l'obtenció de les dades de borsa i actualització d'aquestes a la BD del sistema

Com podeu observar hem fet correspondre a cada subsistema un paquet, però es podria haver tractat d'una manera diferent. La relació entre el paquet Dades Bursátils i Administració s'explica pel fet que es tasca de l'administrador dir on es tenen que buscar les dades.

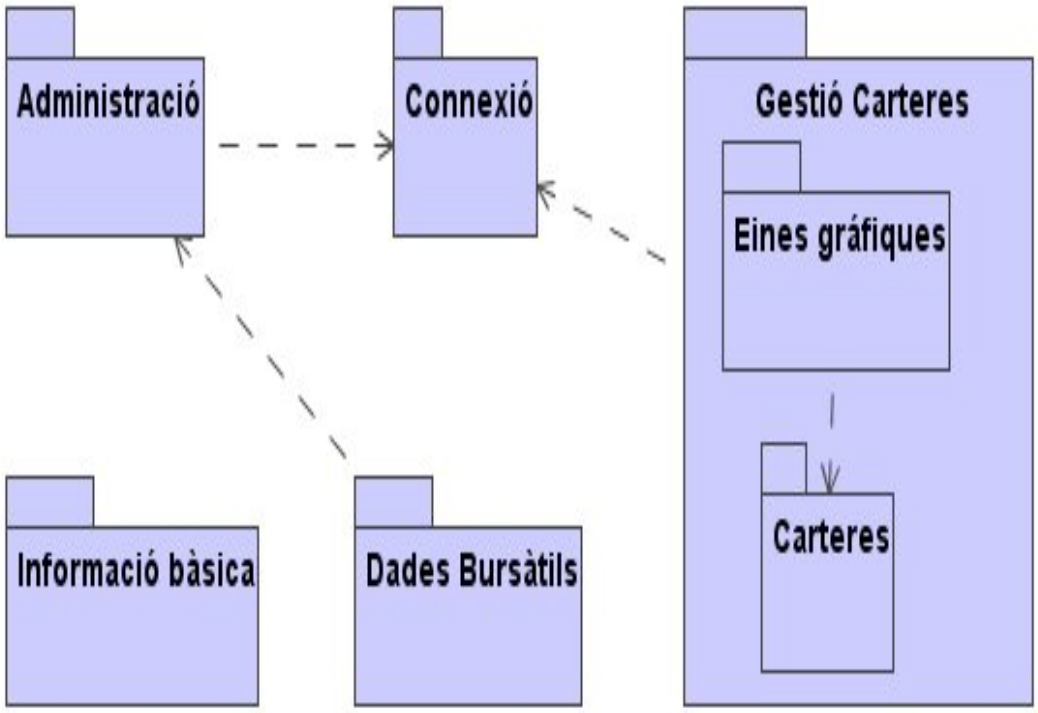


Figura 1. Diagrama de paquets

2.5 Especificacions de les funcionalitats per subsistema

Aquest apartat descriu de manera introductòria les funcionalitats de cadascun dels subsistemes, la qual cosa permetrà definir posteriorment els casos d'ús. Trobareu una descripció més detallada de les funcionalitats dels subsistemes en els casos d'ús.

- **Subsistema d'administració.**

  Aquest subsistema inclou totes les funcionalitats que permeten gestionar l'alta, baixa i modificació d'índexs, empreses i proveïdors de dades, així com, llistat tots el inversors donats d'alta al sistema.

  Descripció de funcionalitats:

  1. Gestió Proveïdor de Dades: permet donar d'alta, modificar i eliminar proveïdors de dades d'Internet per l'actualització dels preus.

  2. Gestió Empreses: permet donar d'alta, modificar i eliminar empreses de la BD.

  3. Gestió Índexs: permet donar d'alta, modificar i eliminar empreses de la BD.

  4. Llistar Inversors: permet llistar a tots els inversors del sistema.

- **Subsistema de connexió.**

  En aquest subsistema es gestiona l'accés al sistema per part dels diversos usuaris.

  Dóna accés a les diverses funcionalitats que els hi correspon en funció del rol que tenen assignat.

  Descripció de funcionalitats:

  1. accés al sistema.
  2. accés a les funcionalitats de cada subsistema què els hi correspon segons el rol d'usuari.

- **Subsistema de informació bàsica.**

  Aquest subsistema es correspon amb la pàgina principal de la web BORSA-X_{TRADER} des d'aquí es podrà veure informació del valors del IBEX-35, top dels millors inversors i top dels millors i pitjors valors del Mercat Continuo, igualment, permetrà a l'usuari donar-se d'alta com Inversor.

  Descripció de funcionalitats:

  1. alta inversor: Permet donar d'alta a un usuari com a inversor i així poder gaudir de tota la funcionalitat del sistema.
  2. consultar dades: Una vegada l'usuari accedeix a la pàgina principal de BORSA-X_{TRADER} el sistema mostra els valors del IBEX-35, el top dels cinc millors inversors registrats al sistema i el top dels cinc millor i pitjors valors del Mercat Continu.

- **Subsistema de gestió de carteres.**

  Aquest paquet l'hem dividit en dos paquets de servei:

  Eines Gràfiques: Permet totes les funcionalitats pròpies de l'anàlisi tècnic de valors.

  Carteres: Inclou totes les funcionalitats que permeten gestionar carteres de valors, com són, crear carteres, eliminar-les, afegir operacions, cercar empreses per veure les seves dades, afegir alertes, llistar operacions i actualitzar les dades de les empreses.

  Descripció de funcionalitats:

  1. cercar empresa.
  2. alta cartera
  3. veure cartera
  4. eliminar cartera
  5. veure alertes.
  6. modificar alertes
  7. eliminar alerta

8. compra. Afegeix una operació a la cartera
9. venta. Tanca una operació de la cartera
10. llistar operacions de tota la cartera

- **Subsistema de dades bursátils.**

  S'encarrega de tota la part d'obtenció de les dades bursátils mitjançant bases de dades externes (Internet) i desar-les a la base de dades del sistema.

  Descripció de funcionalitats:

  1. Actualitzar dades. El sistema actualitza les dades cada cinc minuts, ho pot fer el Inversor de forma manual o automàtica.

2.6 Casos d'ús

Identificació dels actors

Els actors identificats són 3: usuari, inversor i administrador, tots els quals són usuaris finals del sistema.

El usuari té 2 papers, donar-se d'alta com a inversor i veure les dades mostrades per l'aplicació al connectar-se al sistema.

El inversor té 5 papers, donar d'alta una cartera, eliminar una cartera, veure el contingut d'una cartera, cercar una empresa de la BD per veure el gràfic històric de preus i les seves dades i actualitzar els preus dels valors continguts en les seves carteres.

L'administrador té 4 papers, la gestió dels proveïdor de dades (alta, modificació i baixa) de borsa, la gestió de empreses (alta, modificació i eliminació), la gestió dels índexs (alta, modificació i eliminació) i veure el llistat d'inversors amb els seus guanys.

Diagrama de casos d'ús del model del negoci

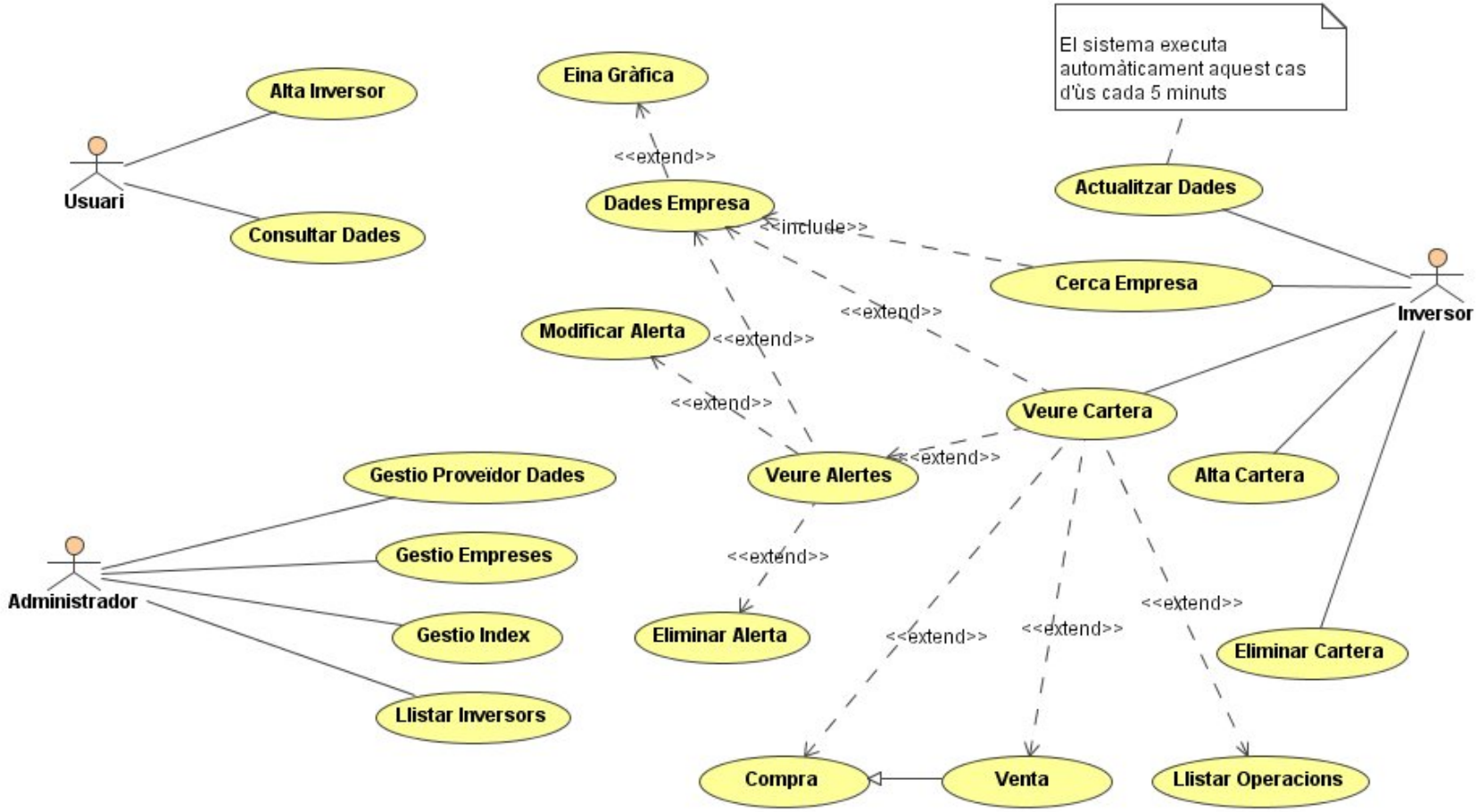


Figura 2. Casos d'ús del model del negoci.

Casos d'us per subsistema.

Subsistema d'informació bàsica.

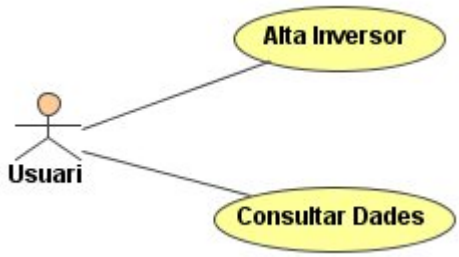


Figura 3. Casos d'ús del subsistema d'informació básica

Cas d'ús número 1. "Alta Inversor".

Resum de la funcionalitat: Amb aquesta funcionalitat un usuari s'enregistra al sistema com Inversor

Paper dins del treball de l'usuari: És un cas d'ús principal dels Usuaris.

Actor: Usuari

Casos d'ús relacionats: Cap.

Precondició: L'actor no esta enregistrat com a Inversor

Postcondició: Hi ha dues possibilitats quan l'usuari prem el botó acceptar:

i. si es produeix cap incidència s'informa a l'usuari.

ii. si l'operació d'alta és correcta, s'introdueix al inversor a la BD amb el seu nom, cognoms, telèfon, e-mail, login i password.

Descripció: El sistema demana a l'usuari les següents dades: nom, cognoms, telèfon, e-mail, login, password i confirmació de password. L'usuari introdueix aquestes dades i el sistema les enregistra a la BD

En cas de polsar el botó cancel·lar es mostra la pantalla anterior (pantalla principal de BORSA-X_{TRADER}).

Observacions: Cap.

Cas d'ús número 2. "Consultar Dades".

Resum de la funcionalitat: Son les dades mostrades pel sistema quan l'usuari accedeix a la pàgina principal de BORSA-X_{TRADER}.

Paper dins del treball de l'usuari: És un cas d'ús principal dels Usuaris.

Actor: Usuari

Casos d'ús relacionats: Cap

Precondició: L'usuari accedeix a la pàgina principal de BORSA-X$_{TRADER}$.

Postcondició: El sistema mostra a l'usuari les següents dades amb temps real diferit màxim quinze minuts: valors del IBEX-35, top dels millors inversors i top dels millors i pitjors valors del Mercat Continu.

Descripció: Una vegada l'usuari accedeix a la pàgina principal de BORSA-X$_{TRADER}$ el sistema mostra els valors del IBEX-35, el top dels cinc millors inversors registrats al sistema i el top dels cinc millor i pitjors valors del Mercat Continu.

Observacions: Cap

Subsistema de gestió de carteres.

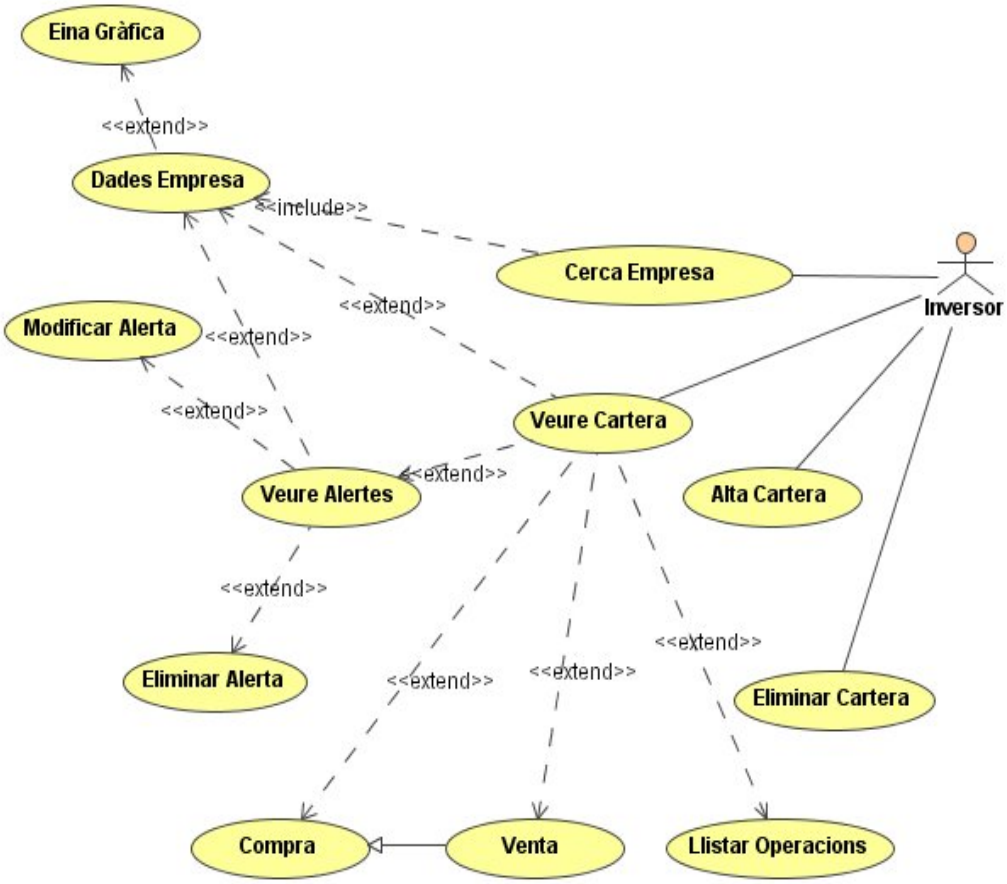


Figura 4. Casos d'ús del subsistema de gestió de carteres.

Cas d'ús número 3. "Cercar Empresa".

Resum de la funcionalitat: El Inversor introdueix el nom de la empresa o el seu ticker i el sistema retorna les dades bursátils de l'empresa més el gràfic de preus diari dels últim tres mesos.

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors.

Actor: Inversor

Casos d'ús relacionats: Dades Empresa, Eina Gràfica

Precondició: L'Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de cercar una empresa.

Postcondició: Hi ha dues possibilitats quan l'usuari prem el botó acceptar

i. El sistema no troba l'empresa seleccionada a la BD

ii. El sistema mostra les dades bursátils de l'empresa cercada més el gràfic de preus diari dels últims tres mesos.

Descripció: : El Inversor introdueix el nom d'una empresa o el seu ticker. El sistema cerca a la BD i mostra la següent informació de l'empresa: últim preu, hora ultima actualització, canvi nominal, canvi percentual, preu tancament dia anterior, rang del dia, rang de les ultimes 52 setmanes, percentatge de variació anual i el gràfic de preus diari dels últims tres mesos. Si el sistema no troba l'empresa a la BD informa a l'usuari indicant-li aquesta acció. Desprès de llegir el missatge d'error l'usuari retornarà la pantalla principal prenent el botó d'acceptar.

Observacions: Per veure les dades de l'empresa la selecció de l'empresa la pot fer de la composició de les seves carteres.

Cas d'ús número 4. "Eina Gràfica".

Resum de la funcionalitat: La següent funcionalitat permet al Inversor afegir al gràfic de preus diferents modalitats de anàlisis tècnic.

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors

Actor: Inversor

Casos d'ús relacionats: Cercar Empresa, Dades Empresa

Precondició: L'actor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció d'afegir elements d'anàlisis tècnic al gràfic de preus

Postcondició:Cap

Descripció: El Inversor pot modificar l'espai temporal de visualització del gràfic de preus des de el primer dia de cotització i el dia actual, pot modificar la composició temporal de la barra de preus de l'empresa a intradia, dies, setmanes o mesos i canviar a espelmes (candlesticks) les barres de preus.

Observacions: Els canvis a la composició temporal de les barres de preus es farà mitjançant una llista desplegable on apareixeran: intradia, dia, setmana, mesos, igualment els canvis a l'espai temporal de visualització es faran mitjançant dos botons per augmentar i disminuir el rang temporal

Cas d'ús número 5. "Veure Cartera".

Resum de la funcionalitat: Permet veure l'estat actual d'una cartera de valors.

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors

Actor: Inversor

Casos d'ús relacionats: Dades Empresa, Eina Gràfica, Veure Alertes, Afegir Operació, Llistar Operacions, Eliminar Cartera

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de veure una cartera.

Postcondició: El sistema mostra l'estat actual de la cartera.

Descripció: Amb aquesta funcionalitat es mostra al Inversor la composició de la cartera seleccionada i el seu estat actual. La informació mostrada per cada empresa es la següent: títols del valor, nom de l'empresa, últim preu, diferència percentual, hora de l'últim preu, preu de compra, plusvàlues i minusvàlues, rendibilitat (latent), així com, les rendibilitat fixes de la cartera

Observacions: Des de aquesta pàgina es podem afegir alertes els diferents valors, veure el llistat d'operacions de la cartera, compra i/o vendre nous valors i seleccionar una empresa de la cartera per veure les seves dades i gràfic.

Cas d'ús número 6. "Alta Cartera".

Resum de la funcionalitat: Crear una cartera

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors

Actor: Inversor

Casos d'ús relacionats: Veure Cartera, Eliminar Cartera

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de donar d'alta una cartera.

Postcondició: El sistema afegeix una cartera a l'actor.

Descripció: El sistema presenta els següents camps per la creació d'una cartera: nom de la cartera, data de creació i observacions. El Inversor introdueix aquestes dades i el sistema crea la cartera.

Observacions: Cap

Cas d'ús número 7. "Eliminar Cartera".

Resum de la funcionalitat: Permet eliminar una cartera

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors

Actor: Inversor

Casos d'ús relacionats: Cap

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de donar d'eliminar una cartera.

Postcondició:El sistema elimina la cartera

Descripció: Amb aquesta funcionalitat el sistema elimina de la llista de carteres la cartera seleccionada. Una vegada esborrada, el sistema informa al Inversor de l'operació i torna a la pantalla principal del Inversor quan aquest pressiona el boto d'acceptar.

Observacions: Cada cartera te un boto per seleccionar aquesta funcionalitat.

Cas d'ús número 8. "Veure Alertes".

Resum de la funcionalitat: Permet veure les alertes del valors d'una cartera

Paper dins del treball de l'usuari: Es un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Modificar Alerta, Eliminar Alerta, Dades Empresa

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de veure les alertes d'una cartera.

Postcondició:El sistema mostra les alertes de cada valor de la cartera

Descripció: Per cada valor de la cartera es mostra el límit superior i el límit inferior de preus que fan que el sistema alerti al inversor. Quan es produeix l'alerta s'envia un e-mail al inversor indicant aquest fet.

Observacions: Per cada valor de la cartera es pot modificar i esborrar els límits d'alerta.

Cas d'ús número 9. "Modificar Alerta".

Resum de la funcionalitat: Permet modificar els límits d'alerta d'un valor de la cartera.

Paper dins del treball de l'usuari: Es un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Veure Alertes, Eliminar Alerta, Compra

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de modificar els límits d'alertes d'un valor de la cartera.

Postcondició:El sistema modifica els límits d'alerta del valor de la cartera

Descripció: El sistema mostra el límit superior i el límit inferior de preus del valor que es vol modificar. El inversor els modifica i accepta els nous límits.

Observacions: El Inversor modifica els límits escrivint el quadre de text corresponent.

Cas d'ús número 10. "Eliminar Alerta".

Resum de la funcionalitat: Permet eliminar els límits d'alerta d'un valor de la cartera.

Paper dins del treball de l'usuari: Es un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Modificar Alertes, Eliminar Alerta, Compra

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció d'eliminar els límits d'alertes d'un valor de la cartera.

Postcondició:El sistema eliminar els límits d'alerta del valor de la cartera

Descripció: El sistema elimina el límit superior i el límit inferior de preus del valor desitjat.

Observacions: Cap.

Cas d'ús número 11. "Compra".

Resum de la funcionalitat: Afegeix un valor a la cartera

Paper dins del treball de l'usuari: És un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Venta, Veure Cartera, Llistar operacions

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció d'afegir una operació de compra a la cartera.

Postcondició: El sistema afegeix la operació de compra a la cartera

Descripció: El sistema sol·licita al Inversor les següents dades: la empresa que vol afegir a la cartera, la data, el preu de compra, el nombre de títols i els límits d'alertes. EL Inversor introdueix aquestes dades i el sistema afegeix el valor a la cartera

Observacions: La selecció de l'empresa es realitza mitjançant un menú desplegable que inclou totes les empreses del Mercat Continuo.

Cas d'ús número 12. "Venta".

Resum de la funcionalitat: Tanca una posició contra un valor a la cartera

Paper dins del treball de l'usuari: És un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Compra, Veure Cartera, Llistar operacions

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de tancar una operació de la cartera.

Postcondició: El sistema tanca la operació oberta de la cartera i calcular la rendibilitat obtinguda.

Descripció: Els passos són els mateixos de **Compra** més els següents: la llista d'empreses que apareixen són les de la cartera, el nombre de títols a vendre no pot ser més gran que els de la posició de compra. El sistema calcula la rendibilitat obtinguda amb l'operació i l'afegeix a la rendibilitat de la cartera. Si el nombre de títols és més gran que el de la posició oberta, el sistema mostra una pantalla d'error indicant aquest fet. Si el nombre de títols es inferior al de la posició oberta el sistema calcula la rendibilitat seguint un criteri FIFO

Observacions: Cap

Cas d'ús número 13. "Llistar Operacions".

Resum de la funcionalitat: El sistema mostra totes les operacions realitzades a la cartera.

Paper dins del treball de l'usuari: És un cas d'ús esporàdic dels Inversors

Actor: Inversor

Casos d'ús relacionats: Compra, Venta, Veure Cartera

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de llistar totes les operacions de la cartera.

Postcondició: El sistema mostra per pantalla totes les operacions realitzades a la cartera

Descripció: El sistema mostra per pantalla totes les operacions realitzades a la cartera

Observacions: Si el nombre d'operacions és més gran que l'espai destinat a la pantalla, s'utilitzarà la barra de desplaçament vertical per veure totes les operacions.

Subsistema d'administració

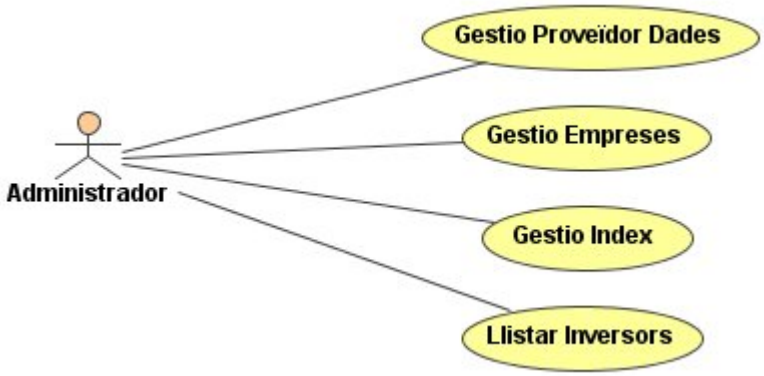


Figura 5. Casos d'ús del subsistema d'administració

Cas d'ús número 14. "Gestió Proveïdors Dades".

Resum de la funcionalitat: Aquest cas d'ús permet donar d'alta, modificar i eliminar proveïdors de dades d'Internet per l'actualització dels preus.

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Administradors

Actor: Administradors

Casos d'ús relacionats: Actualitzar dades

Precondició: L'Administrador s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de gestionar els proveïdors de dades

Postcondició: El Sistema actualitza a la BD les accions executades pel Administradors referents als proveïdors de dades.

Descripció: El sistema mostra de tots els proveïdors de dades donats d'alta, el nom, la url i la qualitat del servei, aquest últim es calcula en funció de les respostes fallides del proveïdor. L'administrador pot donar d'alta un nou proveïdor indicant el nom i la url on buscarà les dades. Pot modificar d'un proveïdor la seva url o eliminar un proveïdor.

Observacions: El sistema obté les dades d'un arxiu XML que ofereix el proveïdor.

Cas d'ús número 15. "Gestió Empreses".

Resum de la funcionalitat: Aquest cas d'ús permet donar d'alta, modificar i eliminar empreses de la BD

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Administradors

Actor: Administradors

Casos d'ús relacionats: Dades Empresa, Cercar Empresa i Actualitzar Dades.

Precondició: L'Administrador s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de gestionar els proveïdors de dades

Postcondició: El Sistema actualitza a la BD les accions executades pel Administradors referents a les empreses.

Descripció: El sistema mostra de totes les empreses incloses a la BD, el seu nom, ticker. L'Administrador pot donar d'alta una nova empresa, indicant el seu nom i el seu ticker. Pot modificar el nom i/o el ticker d'una empresa o eliminar una empresa de la BD.

Observacions: Cap

Cas d'ús número 16. "Gestió Indexs".

Resum de la funcionalitat: Aquest cas d'ús permet donar d'alta, modificar i eliminar índexs de la BD

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Administradors

Actor: Administradors

Casos d'ús relacionats: Consultar Dades

Precondició: L'Administrador s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de gestionar els índexs

Postcondició: El Sistema actualitza a la BD les accions executades pel Administradors referents als índexs

Descripció: El sistema mostra de tots els índexs inclosos a la BD, el seu nom i ticker. L'Administrador pot donar d'alta una nou índex, indicant el seu nom i el seu ticker. Pot modificar el nom i/o el ticker d'un índex així com la composició d'empreses que el formen, també pot eliminar un índex de la BD.

Observacions: Per conformar la composició del índex apareixeran per pantalla dos finestres, una amb les empreses que te el índex i un altre amb les empreses que no pertanyen al índex, mitjançant dos botons (esquerra, dreta) es podrà dur una empresa d'una finestra a l'altre.

Cas d'ús número 17. "Llistar Inversors".

Resum de la funcionalitat: Permet llistar els inversors de la BD

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Administradors

Actor: Administradors

Casos d'ús relacionats: Cap

Precondició: L'Administrador s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció de llistar els inversors de la BD

Postcondició: El sistema mostra el llistat de tots els inversors de la BD.

Descripció: El sistema mostra el llistat de tots els inversors de la BD, de cadascun mostra el nom i les rendibilitats obtingudes (latents + fixes).

Observacions: Si el nombre de inversors és més gran que l'espai destinat a la pantalla, s'utilitzarà la barra de desplaçament vertical per veure tots les inversors.

Subsistema de dades bursátils.

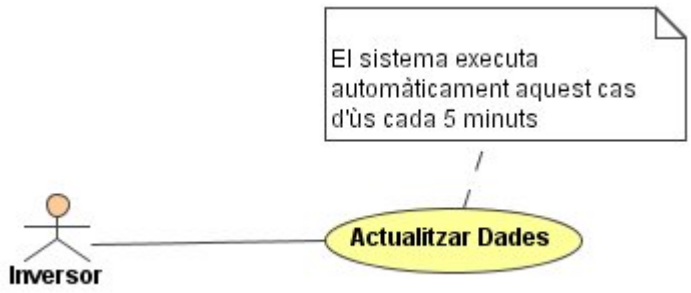


Figura 6. Casos d'ús del subsistema de dades bursàtils.

Cas d'ús número 18. "Actualitzar Dades".

Resum de la funcionalitat: Permet actualitzar els preus dels valors

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors

Actor: Inversor

Casos d'ús relacionats: Veure Cartera

Precondició: El Inversor s'ha validat correctament en el sistema, té el rol que li permet utilitzar aquesta funcionalitat i ha triat l'opció d'afegir elements d'anàlisis tècnic al gràfic de preus

Postcondició: S'ha actualitzen el preus de les empreses.

Descripció: El sistema automàticament cada 5 minuts accedeix als diferents proveïdors de dades per actualitzar els preus de les empreses, amb aquesta funcionalitat l'actor ho pot fer en qualsevol moment.

Observacions: Cap

Subsistema de connexió.

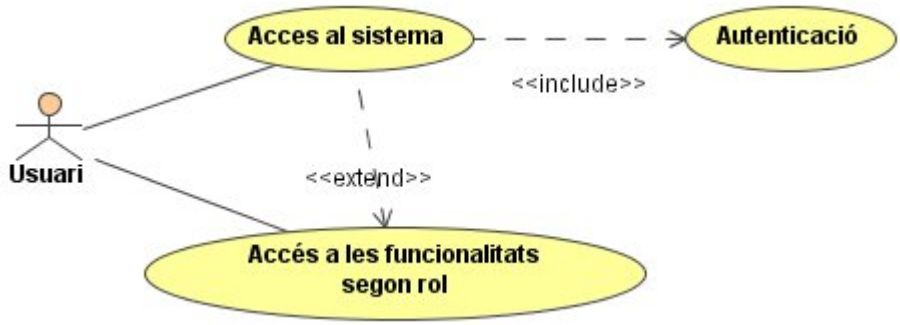


Figura 7. Casos d'ús del subsistema de connexions.

Cas d'ús número 19. "Accés al sistema".

Resum de la funcionalitat: Permet als inversor i administradors entra el sistema per desenvolupar les diferents funcionalitat

Paper dins del treball de l'usuari: Es un cas d'ús principal dels Inversors i d'administradors.

Actor: Inversor i Administrador, anomenats indistintament actor

Casos d'ús relacionats: Tots els principals de Inversors i Administradors

Precondició: L'actor accedeix a la pàgina principal de BORSA-X_{TRADER}.

Postcondició: : Hi ha dues possibilitats quan l'usuari prem el botó acceptar:

i. si es produeix cap incidència s'informa a l'usuari.

ii. si l'operació de login és correcta, l'actor entra al sistema en el rol corresponent.

Descripció: El sistema sol·licita a l'actor el seu login i password, aquest els introdueix i el sistema comprova a la BD que existeixi el binomi correctament. A continuació, el sistema mostra la pantalla principal de l'actor en funció del seu rol.

Observacions: Si el binomi es incorrecte s'informa a l'actor que torni a introduir les dades ja que no existeix a la BD aquesta combinació.

Cas d'ús número 19. "Accés a les funcionalitats segons rol".

Resum de la funcionalitat: permet accedir a les funcionalitats que li correspon a l'usuari en funció del rol que té assignat.

Paper dins del treball de l'usuari: És un cas d'ús principal dels usuaris.

Actor: tots.

Casos d'ús relacionats: accés al sistema.

Precondició: El sistema està en funcionament i admet connexions dels usuaris. L'usuari s'ha validat correctament en el sistema.

Postcondició: Hi ha dues possibilitats:

i. si es produeix cap incidència s'informa a l'usuari.

ii. L'usuari té accés a les funcionalitats que li corresponen segon el rol que té.

Descripció: L'usuari introdueix el seu codi d'usuari i la seva clau i se li dóna accés al sistema amb les funcionalitats que li corresponent.

Observacions: Els rols són excloents entre ells.

2.7 Requisits del maquinari i programari.

Requeriments dels clients.

Els usuaris per accedir al sistema només necessitaran d'un navegador web.

Requeriments del servidor.

S'escull J2EE con a tecnologia de desenvolupament del sistema, doncs Java està orientat a objectes, seguretat i independència de plataforma. L'existència de diverses implementacions de la tecnologia J2EE de codi obert és un altre dels motius per a l'elecció d'aquesta plataforma.

Com a servidor d'aplicacions J2EE utilitzaren JBoss (és de lliure distribució i amb una àmplia utilització en el mercat). Entre altres, incorpora dos contenidors web (un anomenat Jetty i desenvolupat per JBoss i un l'altre és l'Apache Tomcat) i una base de dades relacional (HyperSonic SQL). Per contra s'utilitzarà un SGBD de programari lliure com MySQL.

2.8 Seguretat.

Seguretat d'accés.

La seguretat d'accés fa referència a les mesures necessàries per garantir el correcte accés al sistema. Com es tracta d'una aplicació J2EE tenim la seguretat que es deriva de les característiques de les classes fonamentals de Java. Per exemple, tenim les mesures de seguretat que imposa la JVM, la qual verifica totes les classes abans de carregar-les.

L'aspecte fonamental en aquesta aplicació organitzada en capes, es troba a la capa web. En el nostre sistema s'ha implementat una autentificació bàsica (coneguda com *HTTP basic*), on el servidor (el component encarregat del subsistema de connexió) verifica la identificació de l'usuari i la seva clau, dades que s'han tramés mitjançant un formulari. Això ens ofereix un nivell baix de seguretat, per la qual cosa caldria enviar les dades des de la capa client mitjançant SSL (*Secure Sockets Layer*).

Caldrà definir un *time-out* (un temps màxim de treball amb una sessió), per evitar que un usuari deixés accidentalment oberta una connexió amb la possibilitat que es produís un accés no autoritzat.

També caldrà restringir el nombre d'intents d'accés al sistema, així com enregistrar les dades corresponents a cadascuna de les entrades al sistema (*log's* d'accés)

Seguretat de les dades.

Pel que fa a la seguretat de les dades, caldria definir, adquirir i/o configurar algun mecanisme per efectuar còpies de seguretat (*backup's*) de la informació continguda a la base de dades. Aquesta tasca es deixa sota la responsabilitat de l'administrador del sistema.

Seguretat del maquinari

S'aconsella la instal·lació d'un sistema RAID, així com disposar d'assistència tècnica professional que s'encarregui del manteniment i reparació del maquinari.

2.9 Funcionalitats de les properes versions.

- Les gràfiques intradia dels índex IBEX-35, EURO STOXX, DOW JONES i NASDAQ a la pàgina principal de BORSA-X$_{\text{TRADER}}$.

- Banner amb desplaçament horitzontal cap a la dreta de les accions del IBEX-35 amb el seu preu actual i % de variació a la part superior i a la part inferior les accions del Mercat Continu, en qualsevol cas, el ticker de l'acció es presentarà en color verd si el % es positiu, vermell si es negatiu i blanc si és neutre.

- Quan un inversor aposta per renda variable perd automàticament la rendibilitat d'aquesta inversió en renda fixa, es per això, que l'aplicació restarà a la rendibilitat obtinguda per renda variable la rendibilitat perduda per renda fixa, aquest es un paràmetre que fixarà l'administrador del sistema tenint en compte l'interès de les Lletres del Tresor.

- Incorporar un nivell de seguretat SSL (Secure Socket Layer) al sistema

- Sobre l'eina gràfica l'inversor podrà dibuixar ratlles de tendència, ratlles de suport i resistència, paral·leles, retrocessos fibonacci, oscil·ladors, etc... S'emmagatzemaren tots els dibuixos fets per un inversor sobre un valor, així quan recuperi un gràfic en una altre sessió no ho tindrà que tornar a dibuixar.

- Utilitzar la plataforma de serveis web per accedir-hi a la informació borsària mitjançant el protocol SOAP, WDSL i UDDI

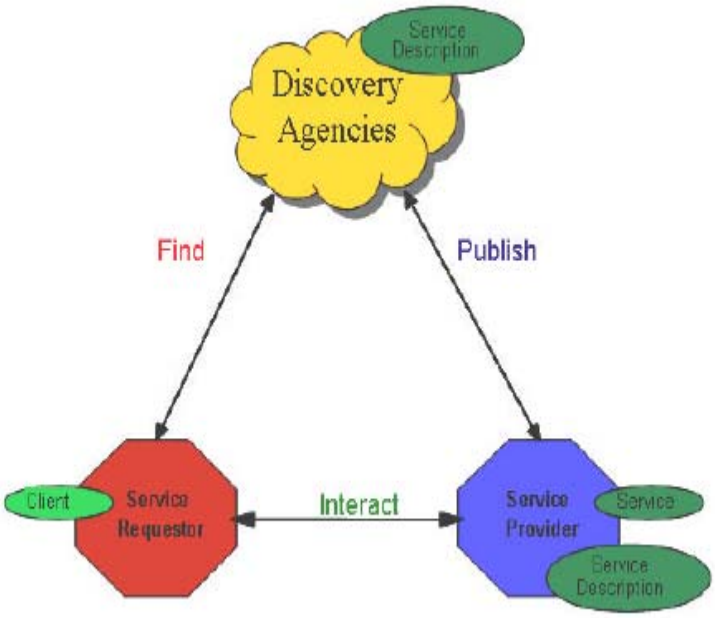


Figura 8. Components del Serveis Web

2.10 Recollida i documentació dels requisits de UI.

Veure document annex de disseny de la interfície d'usuari. Apartat context d'ús.

3. DISSENY.

3.1 Introducció.

Aquest apartat proposa una solució als requeriments i funcionalitats presentades al apartat anterior. Si en el apartat d'anàlisi descrivíem que havia de fer el programari, en aquest apartat explicarem com ho ha de fer. Es discutirà sobre l'arquitectura J2EE a utilitzar, els patrons a emprar, els bastiments que utilitzarem, les estructures de dades i l'especificació de cada programa.

3.2 Java 2 Enterprise Edition (J2EE).

Plataforma escollida, J2EE.

J2EE és una especificació oberta que proporciona un model complert per suportar arquitectures distribuïdes i representa un conjunt de tecnologies que es poden utilitzar conjuntament per implementar un model arquitectònic d'aplicacions distribuïdes de gran dimensions (orientat al món de les empreses).

Bàsicament el model J2EE es basa en quatre nivells (arquitectura multicapa) per a una aplicació empresarial:

- La capa client proporciona la interfície d'usuari. Aquesta capa està formada per programes que demanen dades als usuaris i converteixen les seves respostes en peticions que s'envien a un component que processa la petició. Habitualment es comunica amb el nivell web mitjançant protocol HTTP, però en algunes aplicacions es pot interactuar directament amb el nivell EJB o amb el de la Base de dades [2].

- La capa web representada per un navegador o una aplicació client proporciona funcionalitat a l'usuari. El nivell web proporciona funcionalitat a partir un conjunt de continguts transportats al client a través de HTTP, rebent peticions i enviant respostes als clients que poden estar en qualsevol capa. Controla doncs la interacció entre l'usuari i la capa de negoci.

- La capa EJB o capa de la lògica de negoci ofereix un conjunt d'objectes relacionats amb aquesta i proporciona solucions a aquests problemes. Aquí es troben un o més Enterprise JavaBeans (EJB), cadascun dels quals conté les regles de negoci i als que s'accedeix des dels clients de manera indirecta. Un aspecte molt important a considerar és que la capa EJB permet tenir accés a la lògica i dades del negoci de forma concurrent, sense que afecti al rendiment del sistema. Els EJB's es troben en el servidor d'objectes distribuïts on es gestiona (cada vegada que es crida un EJB) les transaccions, la seguretat i s'assegura que els fils i la persistència s'implementin de forma correcta. Aquests components habitualment accedeixen als components de la capa EIS.

[2] Podeu veure més informació sobre arquitectures J2EE en el llibre JOHNSON, ROB. *J2EE Design and development*. Wrox Press.

- La capa EIS (Enterprise Information System també anomenada capa d'integració) representa tots els recursos empresarials per a l'aplicació (bases de dades, sistemes empresarials de col·laboració, sistemes heretats,etc.)

Com podem observar cadascuna de les capes s'orienta a proporcionar una determinada funcionalitat a l'aplicació. És important distingir entre la ubicació física i la funcional. Dues o més capes poden existir en una mateixa JVM. L'aplicació J2EE que es dissenyi utilitzarà només aquelles capes que necessiti.

Finalment indicar que J2EE es basa en components, contenidors i connectors. Els components són les unitats bàsiques de programació. Les tecnologies de components fonamentals són els Enterprise JavaBeans (EJB) que s'utilitzen en el nivell del servidor d'aplicacions, i les JavaServer Pages (JSP) i *Servlets* que són les tecnologies a nivell de web. Els contenidors són programari que gestiona els components i que proporciona accés als serveis del sistema. La relació entre components i contenidor s'anomena freqüentment contracte.

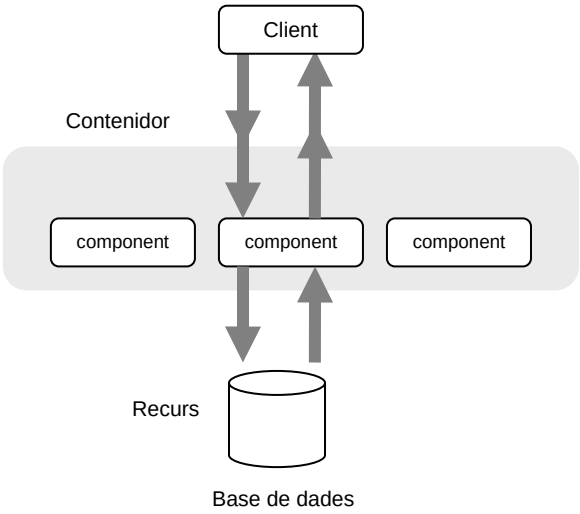


Figura 9. Components, contenidors i recursos.

3.3 Disseny Arquitectònic.

Arquitectura de l'aplicació. [3]

Com hem vist, la separació entre capes ofereix una delimitació de responsabilitats i satisfà, entre altres, les necessitats d'escalabilitat, extensibilitat i flexibilitat. A més s'aconsegueix una reducció en l'acoblament entre les diferents parts.

[3] Per a més informació vegeu, ALUR, CRUPI i MALKS. *Core J2EE Patterns. Best practices and Design Strategies*. Sun Microsystems.

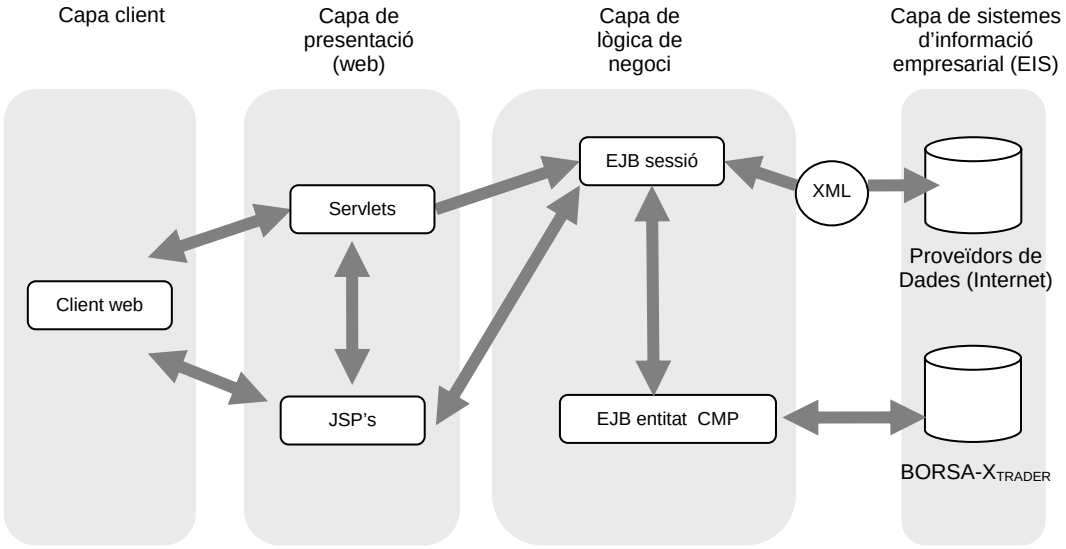


Figura 10. Arquitectura en capes del projecte

Com es veu a la figura anterior, els clients web (Servlets i JSP's) envien una petició als *bean* utilitzant RMI. Els *beans* d'entitat que utilitzen la persistència gestionada pel contenidor (CMP) accedeixen directament a les dades de la BD. Per accedir a les dades dels Proveïdors externs s'envien peticions HTPP i aquests retornen arxius XML .

Disseny capa client.

Al dissenyar la capa client ens podem trobar amb els següents tipus d'aplicacions client:

- navegadors web
- aplicacions d'escriptori.
- aplicacions per a dispositius mòbils.

Els dos primers són els més habituals. En funció del tipus d'aplicació que utilitzem ens caldrà un determinat disseny o un altre. La diferència fonamental entre tots dos, és que en el primer cas la lògica de la presentació s'efectua a la capa web (o capa de presentació), mentre que en el segon ho fa la pròpia aplicació. Un aspecte important és que en el cas d'utilitzar aplicacions d'escriptori la lògica de negoci és només accessible a través del patró *Session Façade*. En el cas de les aplicacions web (si el contenidor EJB i el contenidor web s'allotgen en el mateix servidor) no és necessari. En el cas de les aplicacions d'escriptori existeixen a més els problemes d'instal·lació de llibreries, etc.

En el nostre cas hem optat per utilitzar un navegador web a la capa client.

Disseny capa web.

La capa web és la encarregada de controlar la generació del contingut a l'usuari final. A més efectua la comunicació amb la lògica de negoci per obtenir les dades necessàries, control de flux de pantalles, perfils, autoritzacions, etc.

La característica més important del disseny de la capa web que definim, és la utilització del patró MVC (Model View Controller).

Com veurem més endavant, en projectes de determinada mida i estructura és recomanable la utilització d'un *framework* web que proporciona al desenvolupador una sèrie de serveis, facilitant la seva tasca. La major part dels *frameworks* implementen el patró MVC[4],

Disseny capa de la lògica de negoci.

Conté el conjunt de regles de negoci; entitats i regles que s'encarreguen de la implementació dels processos de negoci d'una empresa. Sun Microsystems recomana la implementació de les funcionalitats (o regles de negoci) com EJB de sessió (amb o sense estat), mentre que el tractament de les dades cal encapsular-ho en EJB d'entitat.[5]

Descartem la utilització de *POJO's* (Plain Old Java Objects). Encara que la lògica de negoci és més senzilla d'implementar, no aprofitem els avantatges que ens ofereixen la plataforma J2EE (transaccions, seguretat, etc.). Així doncs utilitzarem EJB de sessió pels processos, i EJB d'entitat per l'accés a les dades.

Hi ha tres tipus de EJB, *beans* de sessió, d'entitat i orientats a missatges.

Bean d'entitat.

Un bean d'entitat s'utilitza per representar les dades del negoci (entitats del domini de l'aplicació). Estan associats amb les dades allotjades a les bases de dades i saben com realitzar el tractament de la persistència.

Beans de Sessió.

Com hem vist, són l'opció recomanada per la implementació de la lògica de negoci de l'aplicació, i modelen el processos d'aquesta. Es basen en l'arquitectura orientada al servei (SOA), on cada EJB de sessió ofereix un servei a l'exterior. Existeixen dos tipus de beans de sessió. Els beans de sessió sense estat es comparteix entre diferents clients, mentre que un bean de sessió amb estat està dedicat a un únic client, i permet guardar l'estat intern del bean entre successives crides. Utilitzarem beans de sessió amb estat per mantenir el control de la sessió d'usuari en comptes de controlar-ho a nivell de la capa web.

Bean orientat a missatge

Permeten utilitzar un model de comunicació asíncrona en la plataforma J2EE, i són components que permeten rebre i tractar missatges enviats per un client mitjançant el servei de missatgeria de Java (Java Messaging Service o JMS).

Disseny capa EIS.

En aquesta capa s'efectua l'accés a les Bases de dades, sistemes heretats, etc.

Per accedir a les dades podem utilitzar *CMP* o *BMP*. El primer fa referència a que és el contenidor EJB el que gestiona la persistència. El segon, un EJB d'entitat (veure

[4] Per exemple, els models de mostra de Sun per J2EE (Pet Store, i Adventure Builder) recomanen l'ús de *frameworks*.

[5] Veure *blueprints* de Sun Microsystems.

aparat de EJB d'entitat) gestiona la persistència de les dades. En el cas d'utilitzar *POJO's* podem utilitzar directament *JDBC*.

Al nostre cas les dades externes arriben com a fitxers XML i per accedir a la BD del sistema utilitzarem EJB d'entitat amb CMP i CMR i així ens estalviem d'escriure codi JDBC d'accés a la base de dades per recuperar i mantenir l'estat persistent dels EJBs.

Utilització d'un framework (Struts).

Ja hem vist la necessitat d'utilitzar un *framework* a l'apartat de la capa web. Existeixen diversos *frameworks* diferents (*Struts*, Spring, *Tapestry*, *WebWork*, etc...). Escollim *Struts* per ser el més estàndard en el mercat, és programari obert, d'altres bastiments com *Expresso* estan basats en *Struts*, i perquè aplicacions propietàries com BEA *WebLogic* o IBM *WebSphere*, l'utilitzen internament.

Perquè un bastiment per la capa web?, evolució històrica.

Els *Servlets* van oferir una alternativa en el desenvolupament d'aplicacions als CGI, oferint portabilitat i extensibilitat. El problema més important era que calia incorporar l'HTML que s'enviava al client en múltiples sentències d'escriptura. Les JSP's van suposar una solució a aquest problema combinant codi Java amb seccions de codi en d'HTML. Les primeres implementacions (anomenat Model 1) combinaven en les mateixes classes, control de flux, presentació de dades i accés al model.

Posteriorment es va utilitzar *Servlets* i JSP's conjuntament. Les JSP s'encarregaven d'escriure el codi HTML en el navegador i els *Servlets* s'encarregaven del control de flux i de tractar la interacció amb l'usuari. Aquest és el model 2 de desenvolupament d'aplicacions web en Java, i consisteix com veiem en utilitzar el patró MVC que hem vist anteriorment.

El projecte *Struts* es va iniciar l'any 2000 per Craig R. McClanahan (el principal desenvolupador de *Tomcat*) per a proporcionar un *framework* de desenvolupament d'aplicacions. A més *Struts* segueix el patró MVC (entre altres).

Una possible implementació.

La vista estarà formada per JSP's. Per a efectuar les tasques de controlador utilitzarem un *Servlet* subministrat per *Struts*. Aquest controlador accedirà als objectes de la capa de la lògica de negoci, i generarà o seleccionarà els JSP's adients. El model estarà format per diversos EJB que encapsularan la lògica de negoci (responsables de les funcions per importants del sistema). Per a les funcionalitats utilitzarem EJB de sessió (amb estat o sense), i per a la persistència de dades els EJB d'entitat.

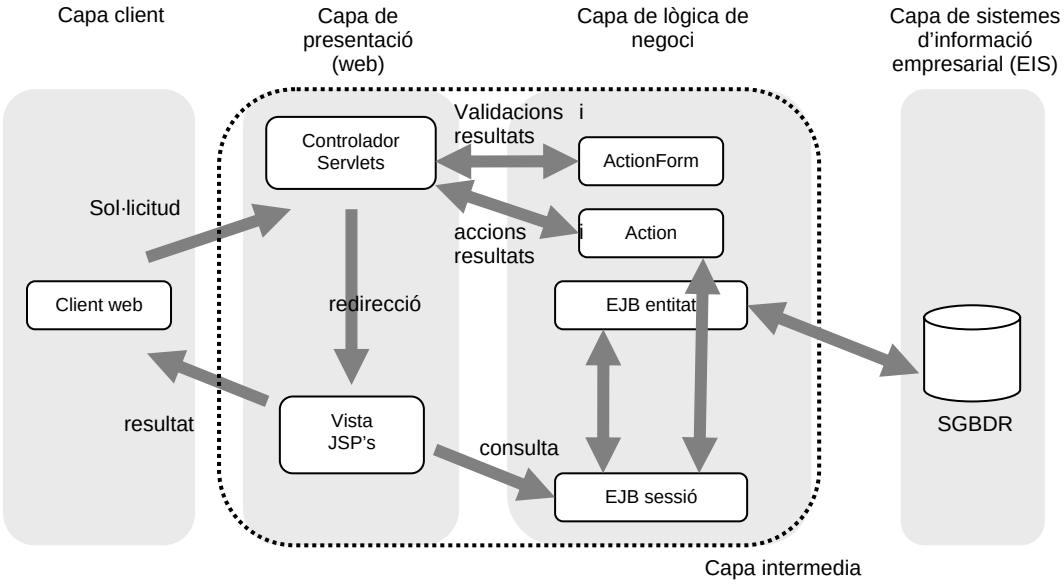


Figura 11. Diagrama de funcionament d'Struts.

Utilització de patrons.

Fonamentalment els patrons que utilitzarem en aquest projecte són els següents: *MVC, Front Controler, Bussines Delegate, HomeFactory i Façade.* Com hem vist anteriorment, el *framework* que utilitzarem implementa alguns d'aquests patrons.

Patró MVC (Model-View-Controller)

Aquest patró (que té els seus fonaments en la tecnologia *Smalltalk*) defineix un element complex en termes de tres subunitats lògiques amb la finalitat de desacoblar el codi d'accés a les dades, el codi de la lògica de negoci i el codi de presentació.

- Model. L'estat de l'element; s'ocupa dels canvis d'estat. És el component que conté una o més classes i interfícies que són les responsables de mantenir les dades del model o aplicació. Mostra la funcionalitat de l'aplicació. Correspondrà a la lògica del negoci (capa de la lògica de negoci).

- Vista. La representació de l'element. Representen una representació de les dades en el component del model. El component vista pot recuperar informació del model però no pot modificar-la (aquestes peticions es dirigeixen al component controlador).

- Controlador. Aquest component gestiona els canvis en el model. Les peticions de canvi poden venir d'un component vista. Processa les entrades dels usuaris, inicia canvis en l'estat del model i en el cas de treballar en web, és el responsable del control de flux.

Els avantatges[6] que s'obtenen de la utilització d'aquest patró són els següents:

- Proporciona una manera excel·lent de fer que un element sigui flexible i adaptable a diverses situacions. Es pot modificar la presentació de la

[6] Podeu veure més informació sobre arquitectures J2EE en el llibre JOHNSON, ROB. *J2EE Design and development*. Wrox Press

informació sense que això afecti, ni el model, ni la gestió de la interacció. També és possible canviar el model sense modificar el disseny.

- Permet un repartiment de tasques entre dissenyadors web i programadors J2EE. (separació de responsabilitats)
- Millora en la documentació.
- Independència de la lògica de negoci del protocol de comunicació emprat.
- Possibilitat de múltiples vistes.
- Possibilitat d'incorporar fàcilment tests de funcionalitats

L'inconvenient és determinar una separació adient que permeti la reutilització en diversos projectes i no la faci excessivament depenent de la solució proposada.

El següent diagrama de components il·lustra el seu funcionament.

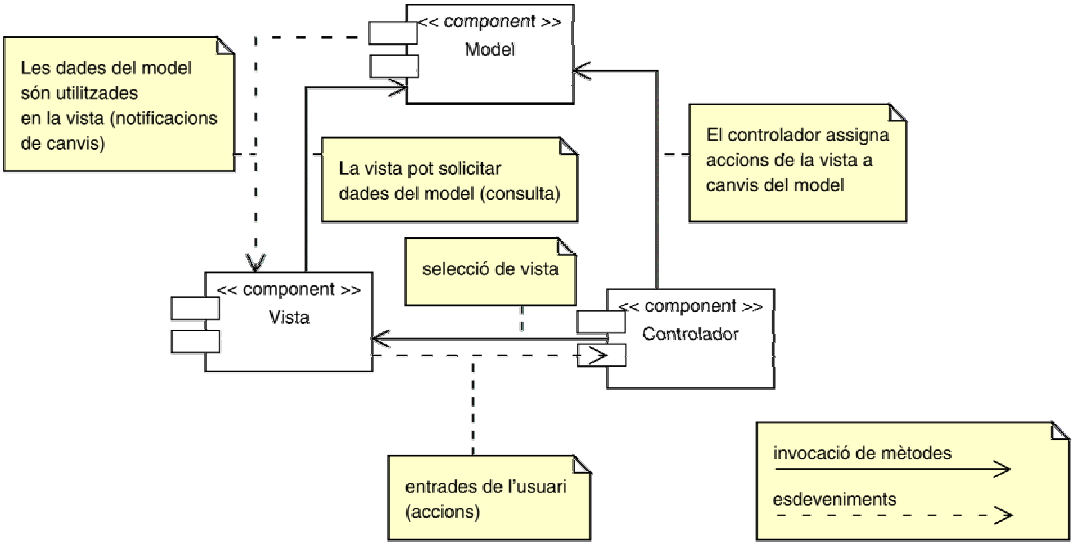


Figura 12. Diagrama de components MVC.

La implementació es pot fer de la següent manera: els components EJB s'utilitzen per a construir components del model. Les JSP i els *servlets* s'utilitzen per a crear components de la vista, mentre que com a components de controlador s'utilitzen, per exemple, EJB de sessió.

El patró Front Controller.[7]

Aquest patró permet que la capa de presentació controli i coordini el processament de cada usuari al llarg de múltiples sol·licituds, centralitzant aquests mecanismes de manera conjunta o distribuïda.

Els avantatges que ofereix la utilització d'aquest patró són:

[7] Per a més informació vegeu, ALUR, CRUPI i MALKS. *Core J2EE Patterns. Best practices and Design Strategies*. Sun Microsystems.

- La lògica es gestionada millor en un mòdul central, en comptes de duplicar el codi de la lògica de negocis a cadascuna de les vistes.
- S'utilitzen múltiples vistes per a respondre a les sol·licituds semblants.
- Permet realitzar el control i registre de l'evolució del client.
- Afavoreix la reusabilitat del codi al llarg de les sol·licituds.

Per implementar-ho utilitzem un controlador per gestionar les sol·licituds, què s'encarrega de les qüestions d'autentificació i autorització, seleccionant la vista adequada i delegant en la capa de la lògica de negocis quan sigui necessari.

Aquest patró ens permet reduir la quantitat de codi que contindran les diverses vistes (JSP's). La aplicació pot utilitzar diversos controladors.

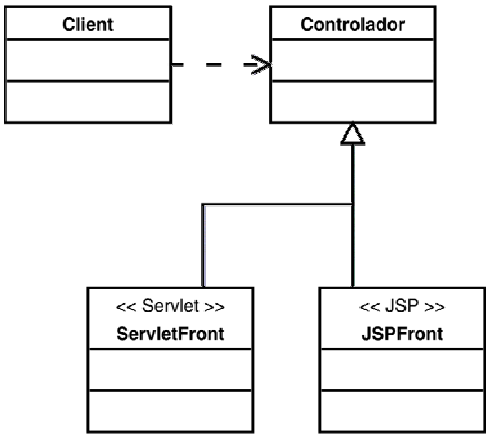


Figura 13. Diagrama de classes del patró Front Controller

El patró Sessión Façade.

Aquest patró és recomanable quan els clients han de cridar molts mètodes de EJB's. Si utilitzem un bean de sessió com a façana, podem agrupar les interaccions entre els objectes de negoci, proporcionant un servei uniforme als clients.

Entre els avantatges que suposa tenim,

- Evita el fort acoblament que provoca la dependència directa entre clients i objectes de negoci.
- Millora el rendiment, doncs redueix el nombre de crides a mètodes entre client i servidor.
- Ofereix una estratègia d'accés uniforme als clients.

El patró Bussines Delegate[8].

La delegació consisteix en un medi d'estendre i reutilitzar la funcionalitat d'una classe mitjançant la creació d'altre classe que ens la proporcioni.

[8] Per a més informació vegeu, ALUR, CRUPI i MALKS. *Core J2EE Patterns. Best practices and Design Strategies*. Sun Microsystems.

La importància d'aquest patró radica en que ens proporciona un mecanisme per decidir quan deurem fer que una classe hereti d'altre, o quan afegir-li una referència per utilitzar els seus serveis.

El patró HomeFactory.

Aquest patró es defineix en el llibre EJB Design Pattern de Floyd Marinescu (Wiley & Sons). Implementar aquest patró permet crear i guardar en cau qualsevol referència propia EJB que necessiti l'aplicació. Donat que no depèn de ServletContext, aquest patró es pot implementar en aplicacions no WEB.

Dins el constructor d'aquest patró s'ha externalitzat els paràmetres provider i factory que es necessiten per accedir al servei de noms. Per fer-ho s'ha utilitzar el fitxer jndi.properties del JBOSS que inclou les següents entrades:

```
java.naming.factory.initial=org.jnp.interfaces.NamingContextFac
tory
java.naming.provider.url=localhost
```

Només canviant el localhost per l'adreça de la maquina on estiguin el components EJB s'accedirà remotament a ells.

3.4 Diagrama estàtic de disseny: classes i jerarquies

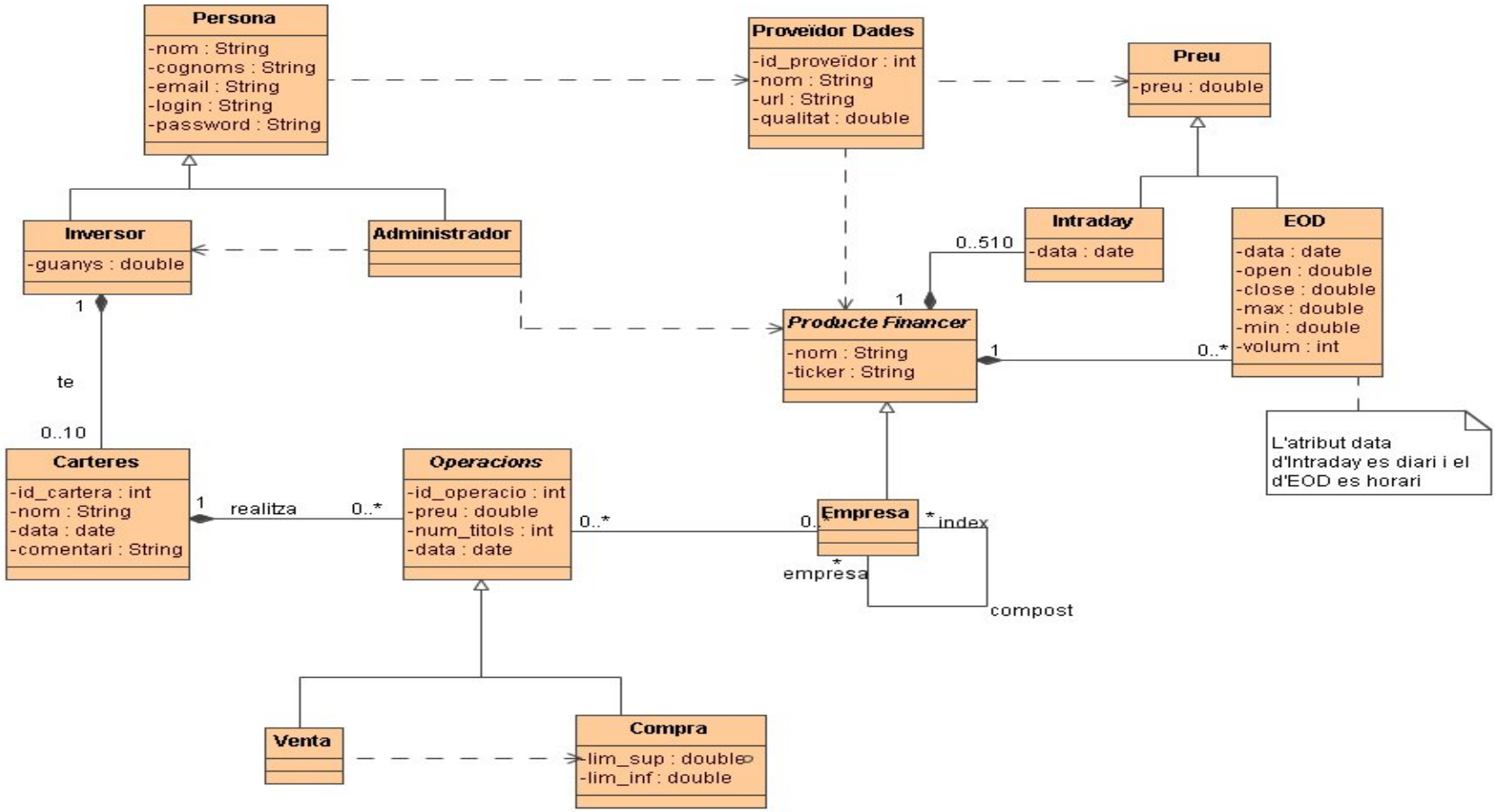


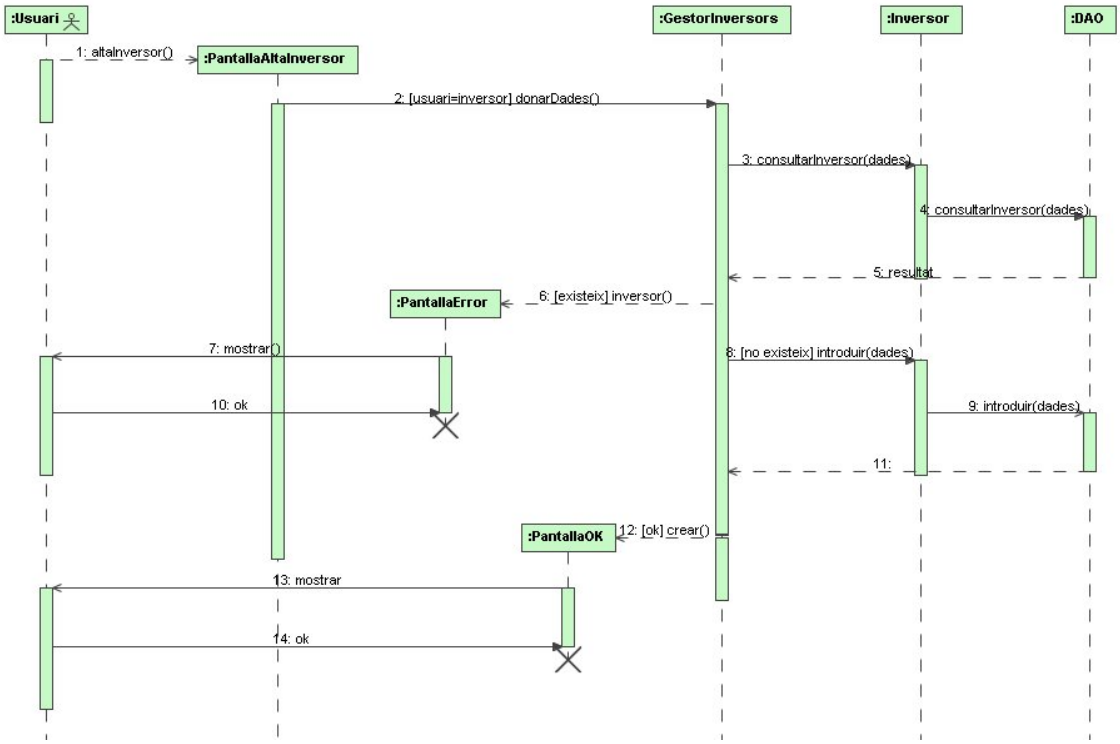
Diagrama 1. Diagrama de classes corresponents al model de negoci.

3.5 Diagrames de seqüència per subsistema.

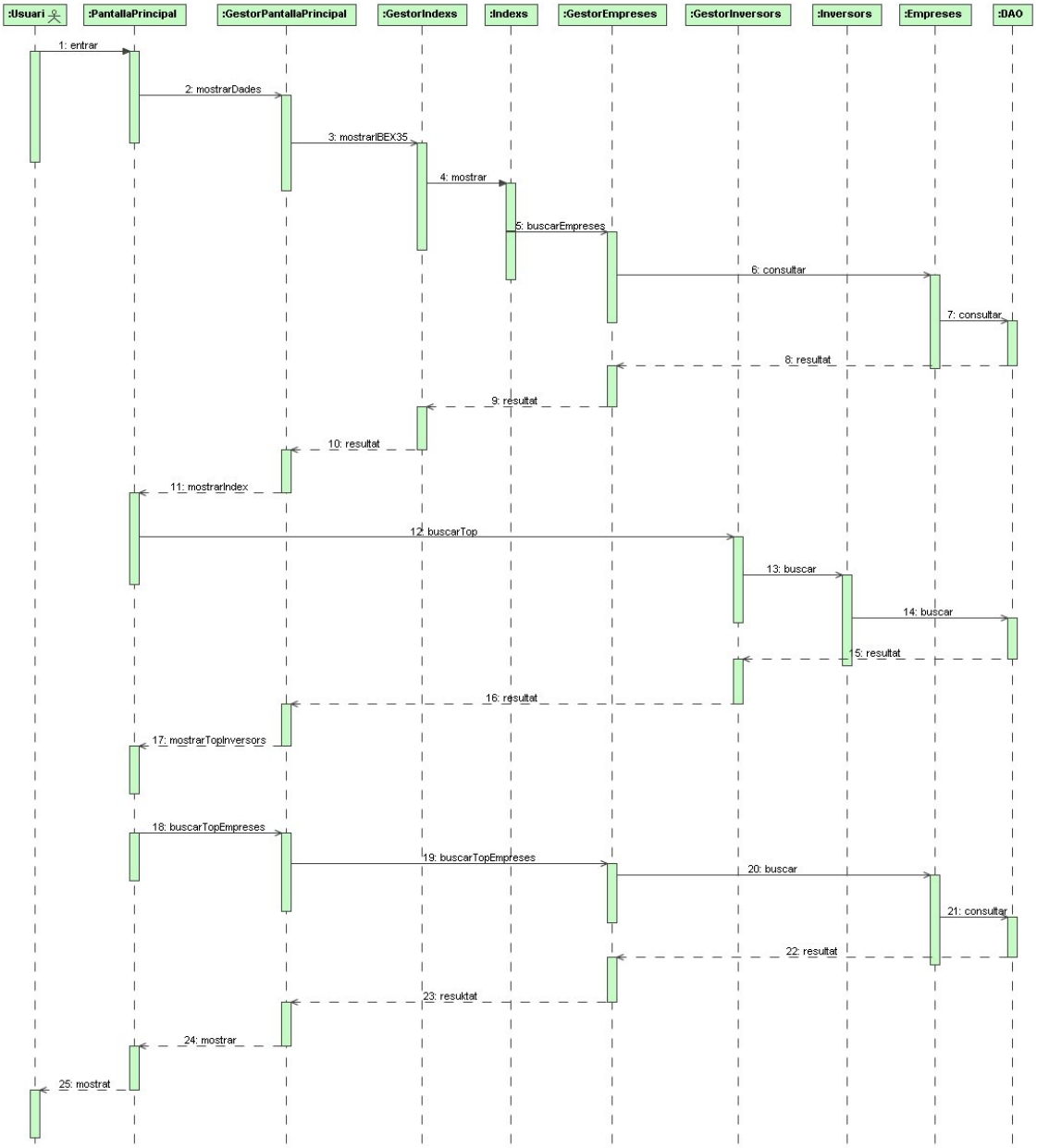
Els següents diagrames de seqüència representen les interaccions que es deriven de cadascun dels casos d'ús, (només es presenten els més representatius). En el disseny d'aquests diagrames apareixen classes fronteres, gestores i d'entitat. Fàcilment podrem assignar cadascuna d'elles a les diferents capes del disseny del sistema i distribuir-les segons les seves responsabilitats mitjançant la separació del patró MVC proposat.

Subsistema de informació bàsica.

1.1 Alta Inversor

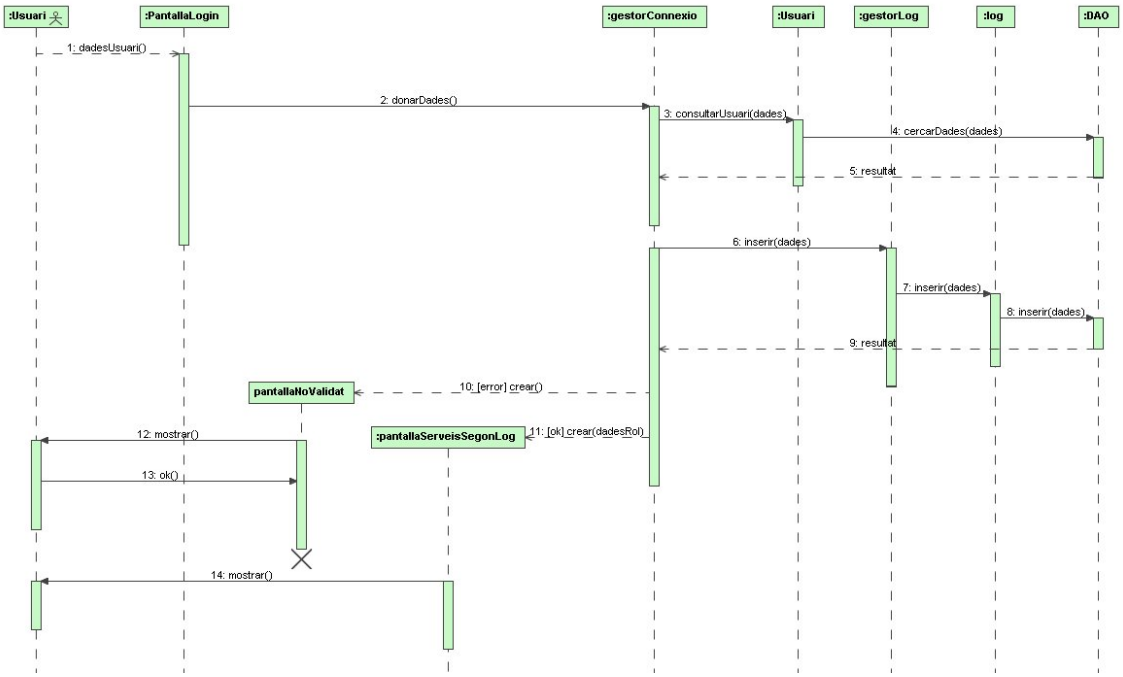


1.2 Consultar Dades

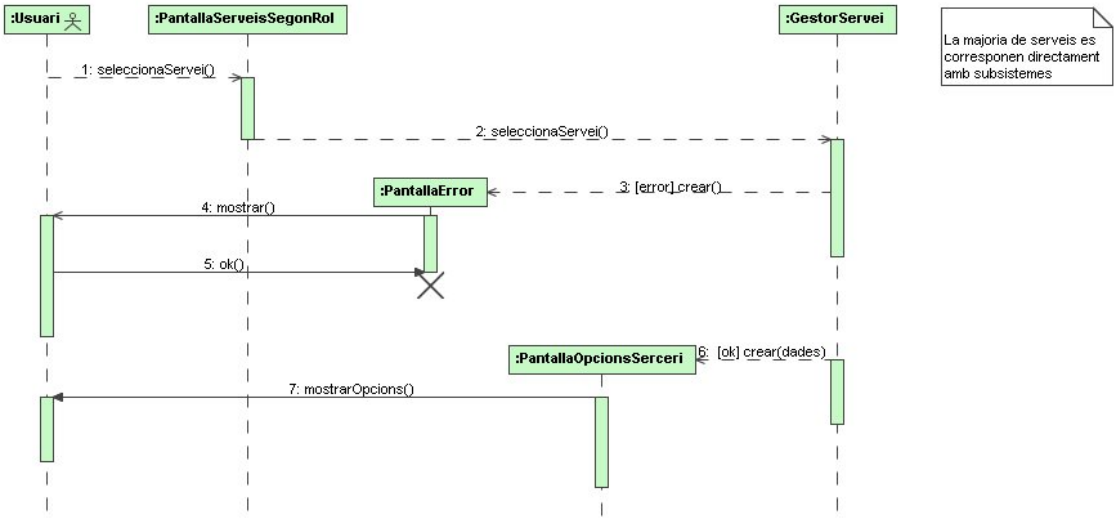


Subsistema de connexió.

2.1 Accés al sistema.

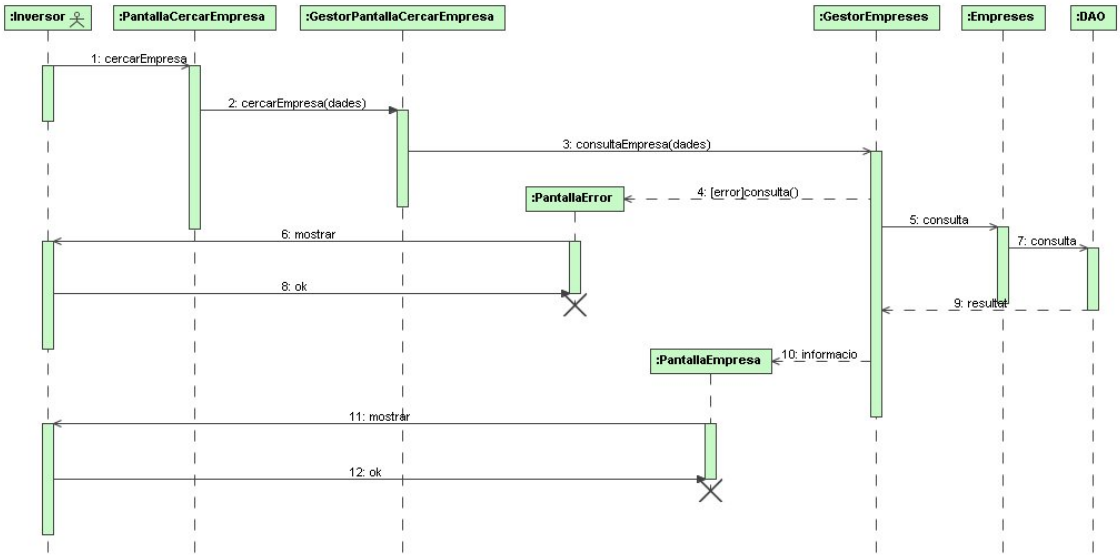


2.2 Accés a les funcionalitats segons rol.

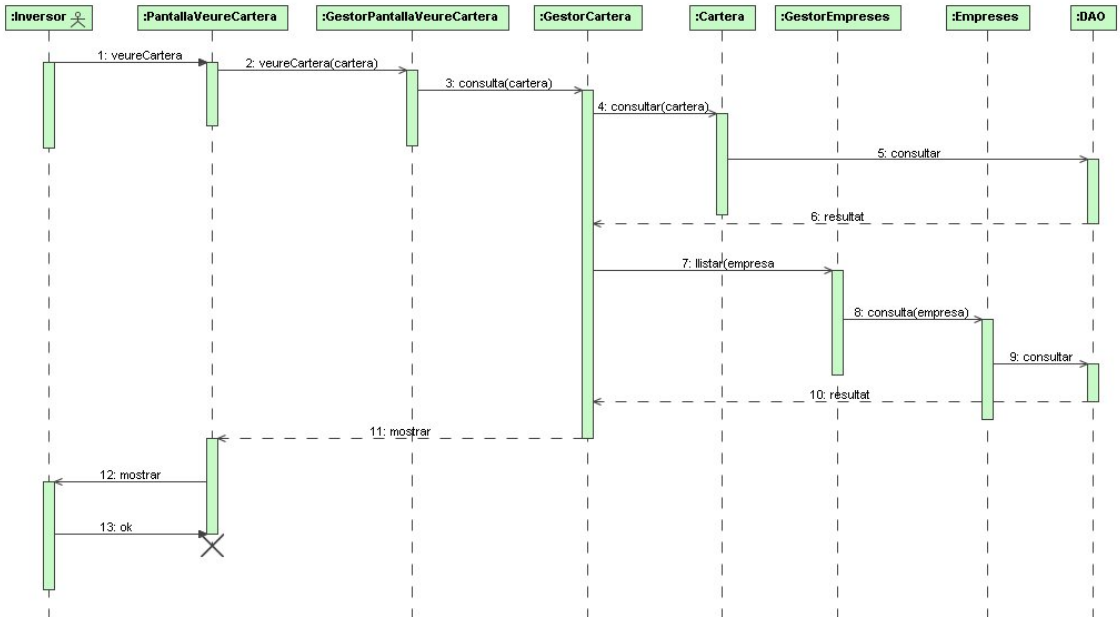


Subsistema de gestió de carteres.

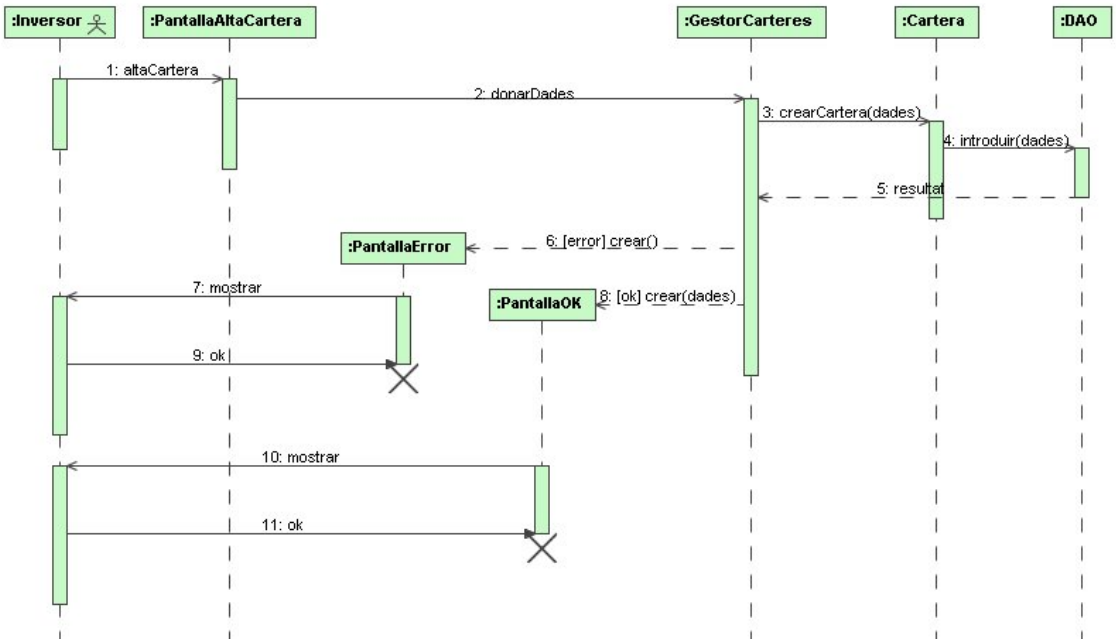
3.1 Cercar Empresa.



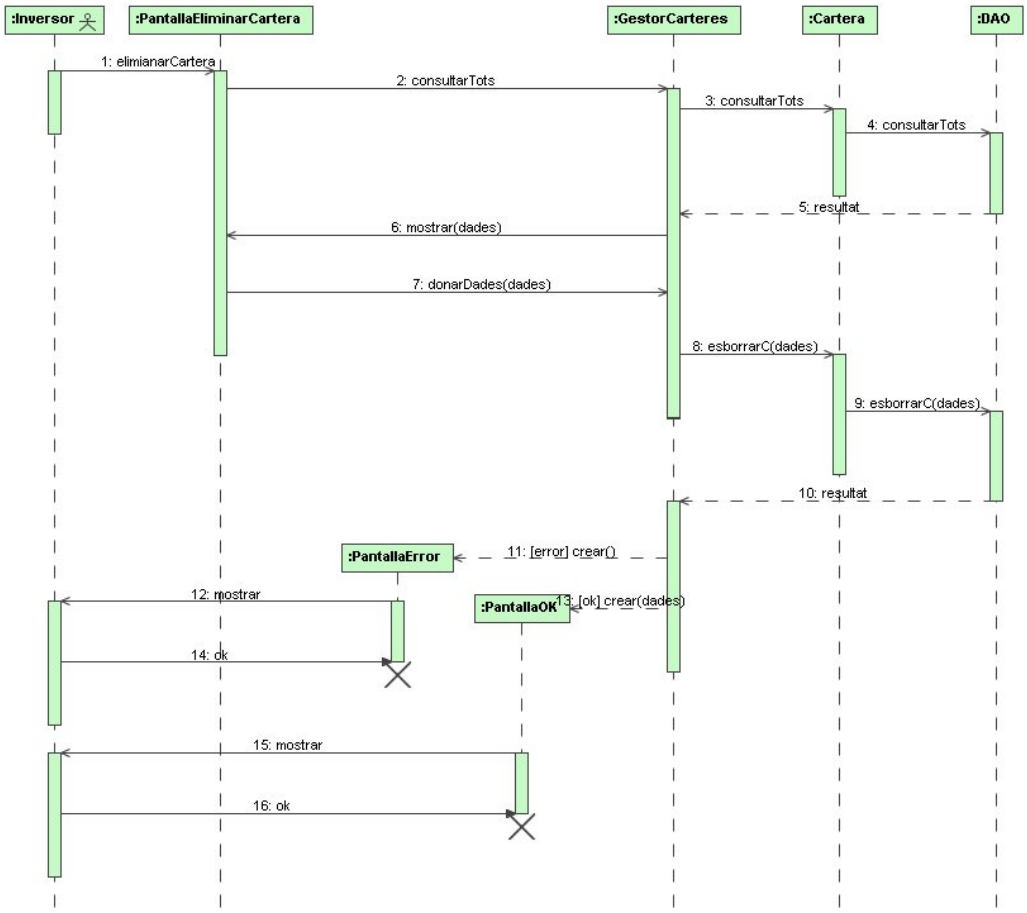
3.2 Veure Cartera.



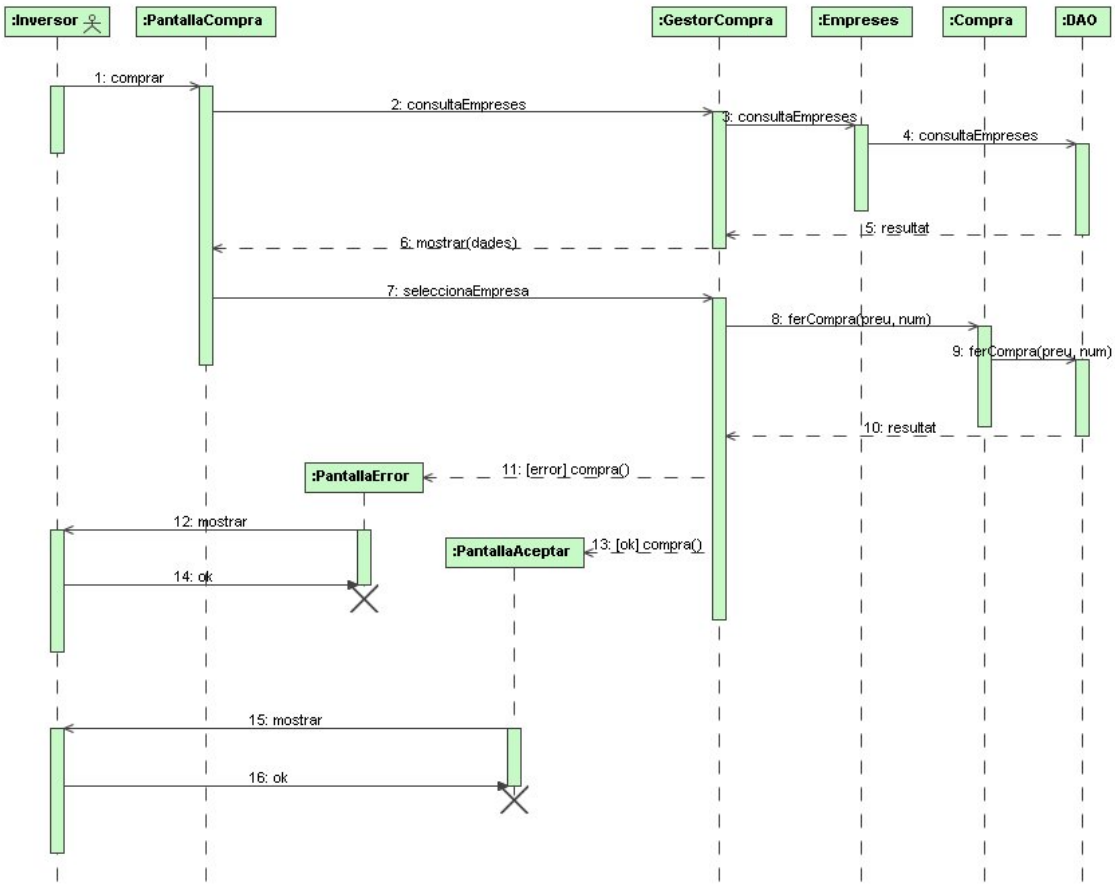
3.3 Alta Cartera



3.4 Eliminar Cartera

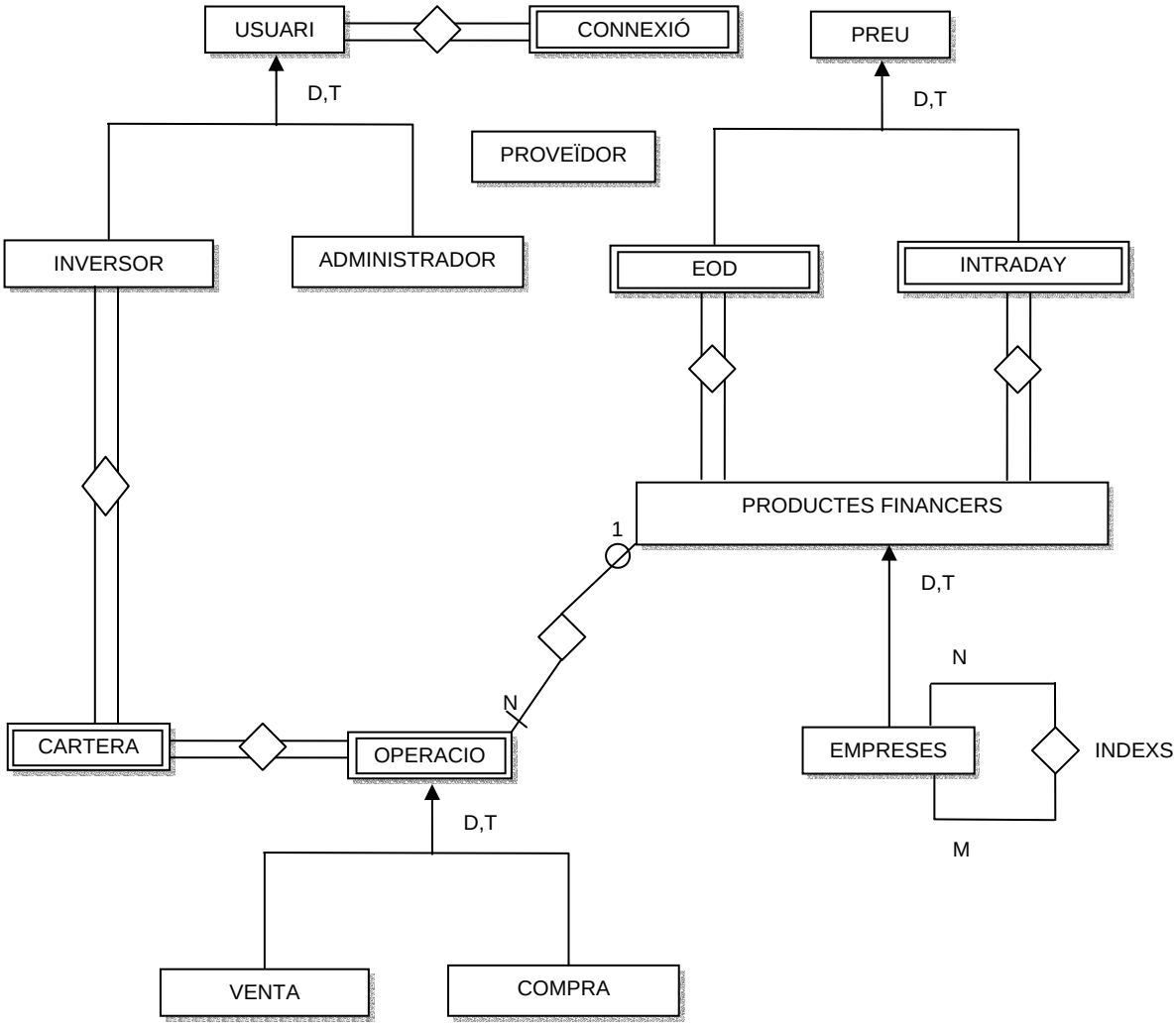


3.4 Compra


:Inversor
:PantallaCompra
:GestorCompra
:Empreses
:Compra
:DAO
1: comprar
2: consultaEmpreses
3: consultaEmpreses
4: consultaEmpreses
5: resultat
6: mostrar(dades)
7: seleccionaEmpresa
8: ferCompra(preu, num)
9: ferCompra(preu, num)
10: resultat
:PantallaError
11: [error] compra()
12: mostrar
14: ok
:PantallaAceptar
13: [ok] compra()
15: mostrar
16: ok

3.6 Disseny de la persistència.

Utilitzem com a enfocament de modelització el **model ER** (model entitat- interrelació), doncs ens ofereix una conceptualització independent del tipus de base de dades que finalment utilitzem.



Els atributs de les entitats que figuren en el diagrama són:

USUARI
<u>login, password</u>, nom, cognoms email

INVERSOR (subclasse d'usuari)
<u>login, password, guanys</u>

ADMINISTRADOR (subclasse d'usuari)
<u>login, password</u>

CARTERA (entitat dèbil, id_cartera l'identifica parcialment, s'identifica completament amb el inversor)
id_cartera, nom, data, comentari

OPERACIO (entitat dèbil, id_operacion l'identifica parcialment, s'identifica completament amb la cartera)
id_operacio, nom_empresa, preu, num_titols, data

COMPRA (subclasse d'operacio)
id_operacio, lim_sup, lim_inf

VENTA (subclase d'operacio)
id_operacio

PRODUCTES FINANCERS
ticker, nom

EMPRESA (subclasse de productes financers)
ticker

PREU
id_preu, preu

EOD (subclasse de preu i entitat dèbil, id_preu l'identifica parcialment, s'identifica completament amb productes financers)
id_preu, data

INTRADAY (subclasse de preu i entitat dèbil, id_preu l'identifica parcialment, s'identifica completament amb productes financers)
id_preu, data, open, close, max, min, volum

PROVEÏDOR
Id_proveïdor, nom, url, qualitat

CONNEXIO(entitat dèbil, data_hora l'identifica parcialment, s'identifica completament amb l'usuari)
data_hora

Disseny Lògic, característiques de la transformació al model relacional.

A partir del disseny conceptual fem la transformació al model relacional. Atès que volem deslligar el disseny lògic del SGBD que utilitzarem, considerarem la supressió de l'herència. Per fer-ho aplicarem la tècnica que consisteix en crear una taula per a la superclasse i una de complementària per a cada subclasse que conté els atributs propis. Cada taula complementària contindrà els atributs propis de la subclasse més els atributs que identifiquen cada fila dins la taula de la superclasse.

Podeu veure les taules que s'obtenen com a conseqüència d'aquesta traducció (en llenguatge SQL) a l'annex sentències SQL.

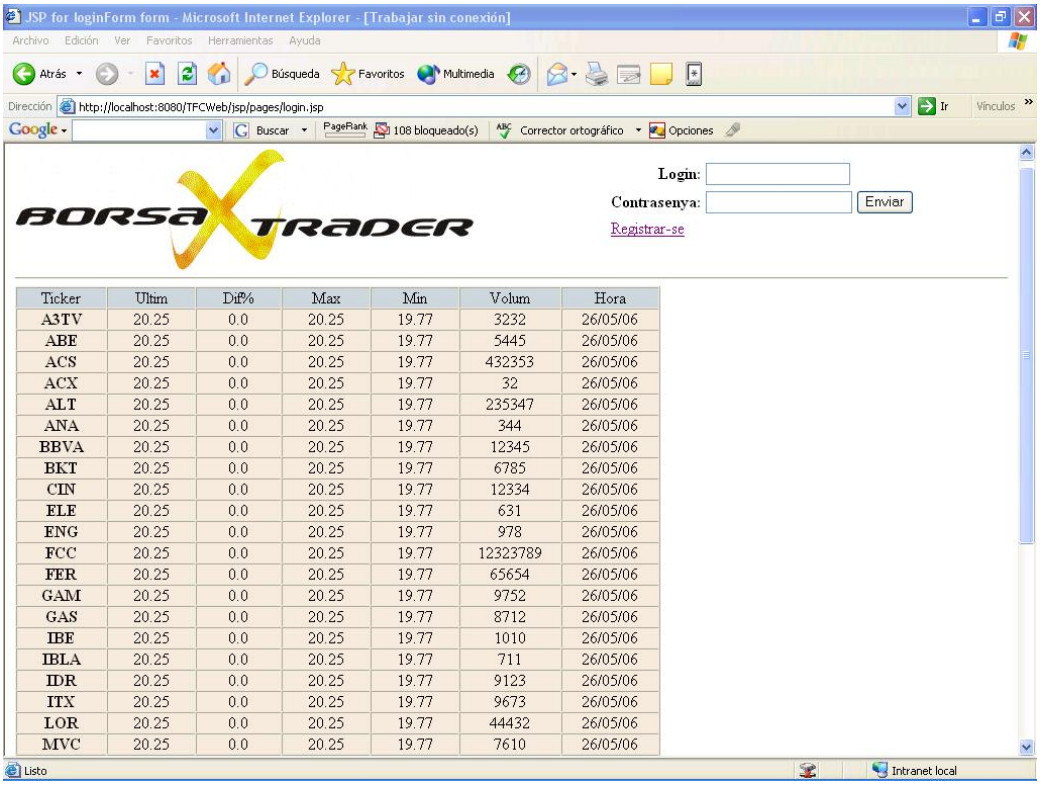
Detalls de la implementació.

J2EE ens ofereix dues maneres de controlar la persistència a través de beans d'entitat: BMP i CMP. En el els BMP (persistència gestionada per bean) el codi de la persistència es implementat pel desenvolupador. Utilitzarem els EJB amb persistència gestionada pel contenidor, els avantatges que presenta són:

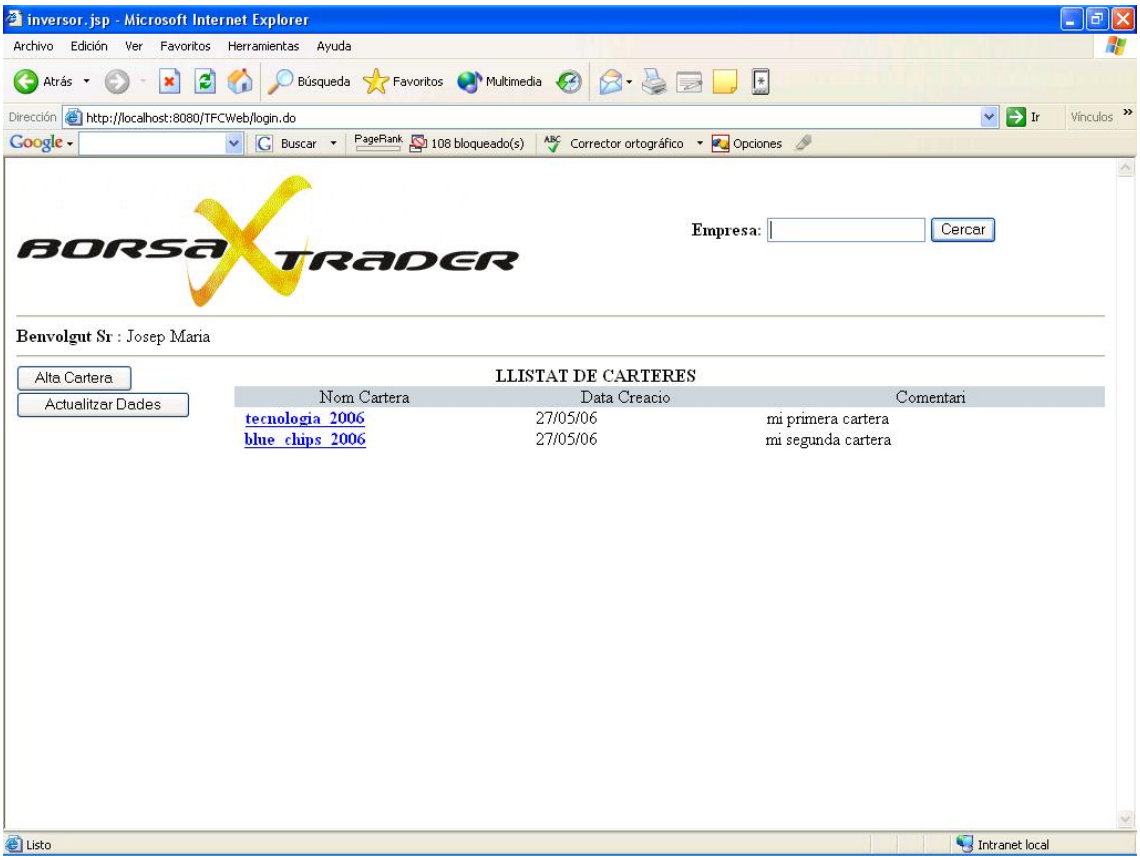
- El codi a desenvolupar és molt més senzill
- El codi desenvolupat està deslligat de la base de dades concreta que utilitzem.
- Teòricament el component ha de tenir un rendiment millor ja que la implementació l'optimitzarà el contenidor al màxim.

3.7 Disseny de la interfície d'usuari.

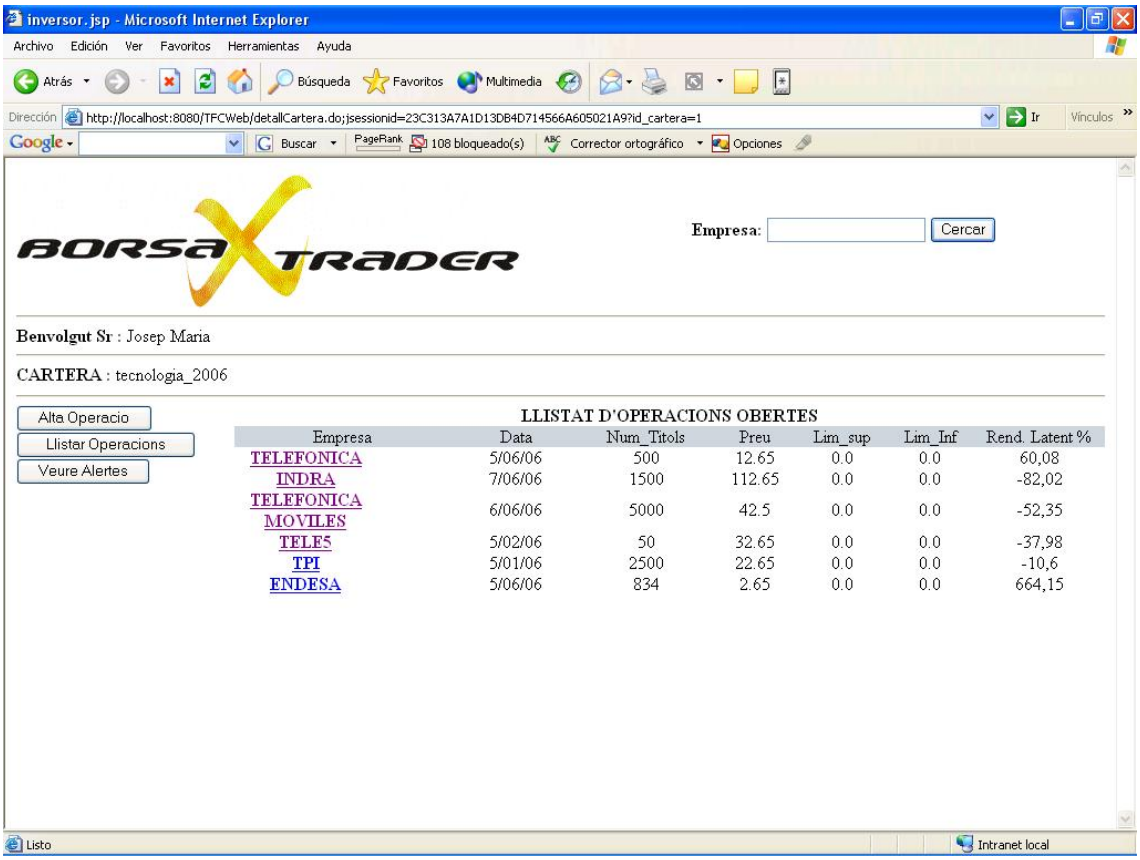
La interfície d'usuari s'ha dissenyat amb planes JSP. Veiem un petit circuit de navegació per l'aplicació:



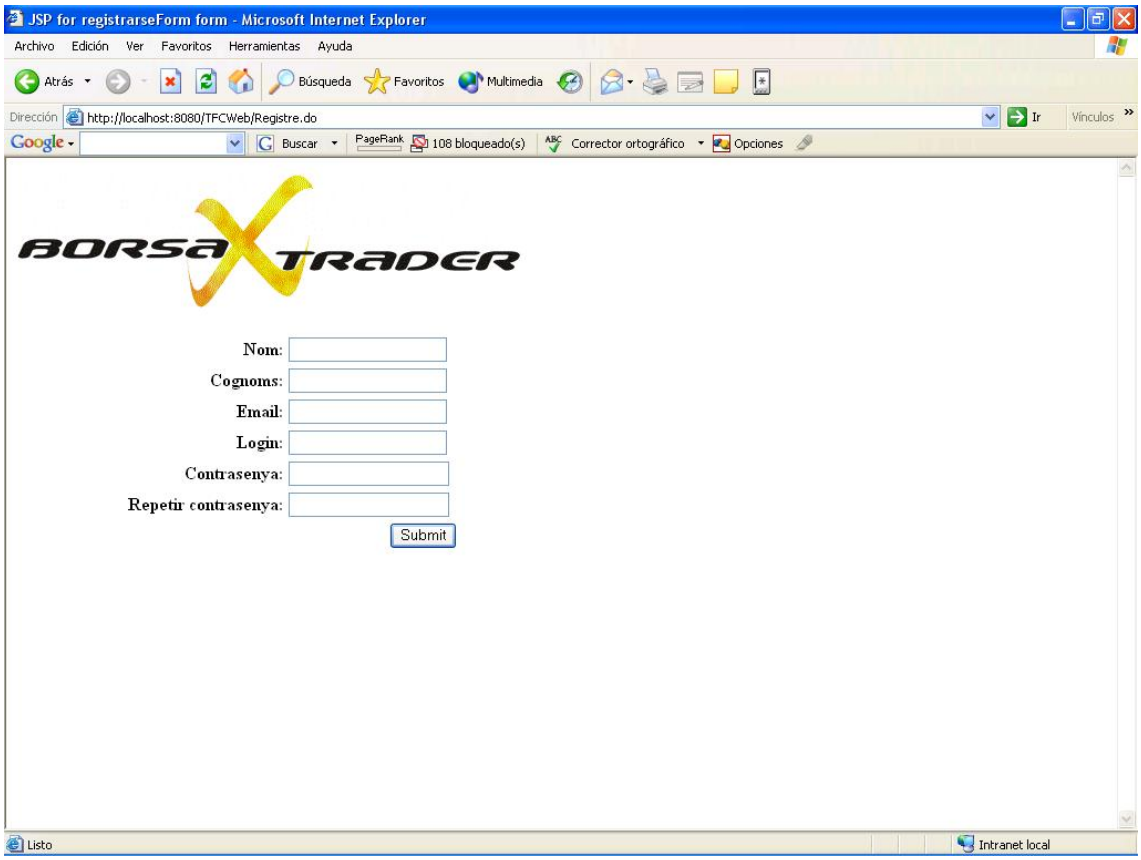
Il·lustració 1. Pantalla de login



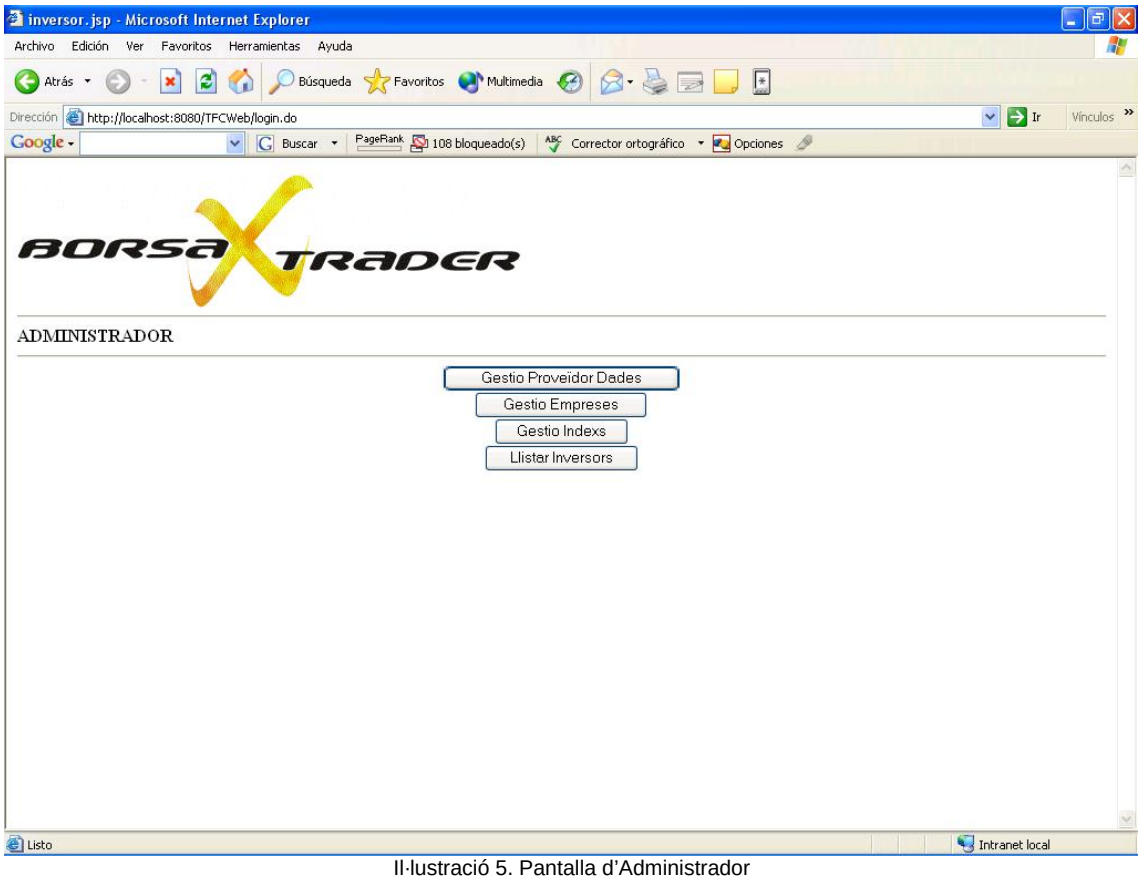
Il·lustració 2. Pantalla d'Inversor



Il·lustració 3. Pantalla Cartera d'Inversor



Il·lustració 4. Pantalla Registre



Il·lustració 5. Pantalla d'Administrador

4. IMPLEMENTACIÓ.

4.1 Requeriments de programari.

Per implementar el sistema utilitzarem el següent programari i configuracions:

- Els navegadors web que s'utilitzin han de complir les especificacions d'HTML 4.01 i permetre l'execució de codi en JavaScript (només per gestionar aspectes de la interfície).
- Plataforma J2SE 1.5 update 3 (SDK).
- Servidor d'aplicacions que compleixi les especificacions J2EE 1.4 Utilitzarem *JBOSS* com a servidor d'aplicacions (versió 4.0.1sp1).
- Contenidor web Apache *Tomcat* 5.0.28 (inclòs amb JBoss)
- Com a Base de Dades s'utilitzarà MySQL
- Framework Apache *struts* (MVC) per al desenvolupament de la capa web. Struts 1.2.4
- Jakarta Ant. Per automatitzar el procés de desplegament i generar els fitxers jar, war i ear de l'aplicació.

4.2 Eines de desenvolupament.

- Utilitzem com a IDE, Eclipse 3.0.2. (programari lliure i open source)
- Plugin JBoss JBossIDE-1.4.1-e30. (utilitzem aquets versió, doncs la darrera versió d'aquest plugin dóna problemes de compatibilitat amb l'eclipse 3.0.X)
- Plugin myeclipse 3.8.4, per gestionar el manteniment dels descriptor del framework Struts (struts-config.xml, etc.)

5. VALORACIÓ ECONÒMICA.

Tot el programari utilitzat en aquest TFC és programari lliure, i les característiques particulars de cadascuna de les llicències les podeu consultar als espais webs dels fabricants. Val a dir que existeixen versions sobre plataformes UNIX o Windows, tant de les eines de desenvolupament triades com dels contenidor web (tomcat) i del servidor d'aplicacions (JBoss). Ara bé, inicialment està pensat per posar en producció sobre una plataforma Microsoft Windows. No és objectiu d'aquest TFC la quantificació econòmica del projecte, donada la gran quantitat d'hores dedicades a l'aprenentatge de l'arquitectura J2EE i dels frameworks Struts.

6. CONCLUSIONS.

Des d'un punt de vista **personal**, l'elecció de realitzar el TFC relacionat amb J2EE va ser motivada atenent el meu interès en el desenvolupament d'aplicacions web relacionades amb el mon bursàtil i encuriosit per aprendre com es podia aplicar el llenguatge Java en el desenvolupament distribuït d'aplicacions empresarials per Internet.

Així doncs, el TFC m'ha suposat aplicar els coneixements adquirits durant la carrera, especialment els relatius a enginyeria del programari, programació OO, i tècniques de desenvolupament d'aplicacions OO, però sobretot, m'ha permès adquirir nous coneixements sobre programació distribuïda i un nombre molt elevat de coneixements de l'arquitectura J2EE, així com implementar-los en un projecte real.

Donat que quan vaig començar no tenia cap coneixement sobre J2EE, penso que he assolit els objectius que apareixen en el pla docent relatius al TFC. La memòria, i sobretot el producte obtingut en forma d'aplicació web, són l'exemple més il·lustratiu.

Des d'un punt de vista **professional**, aquest TFC, tal i com reflexa el seu títol, presenta les etapes d'anàlisi, disseny i implementació, necessàries per a realitzar una aplicació de gestió de carteres i anàlisi tècnic de valors bursàtil. A més d'utilitzar com a arquitectura J2EE, un tret molt característic és la utilització del bastiment STRUTS. Com ha quedat exposat i demostrat, gràcies a Struts hem pogut aplicar el patró MVC/Model 2 d'una manera senzilla i uniforme, permetent la internacionalització, i separant clarament les responsabilitats dels nivell de vista, controlador i del model.

7. GLOSSARI.

- Cartera de valors: Conjunts d'accions agrupades sota un denominador comú, per exemple, d'alt risc, conservadora, biotecnològica,...

- Temps real: Modalitat de presentació de dades bursátils, per la qual, s'actualitza instantàniament (retard de xarxa) qualsevol variació que presenti el preu d'un valor en el mercat.

- Gràfica intradia: Es la representació de la evolució del preu en temps real a la sessió.

- Ticker: Abreviatura de tres o quatre lletres del nom de l'empresa.

- Obrir una posició: Acció per la qual un inversor decideix participar a un producte que cotitza a un mercat, pot ser, a l'alça o la baixa, cas de ser a l'alça es diu que ha obert una posició llarga i si es a la baixa que ha obert una posició curta.

- Tancar una posició: Acció per la qual un inversor decideix deixar de participar a un producte que cotitza a un mercat, pot ser, a l'alça o la baixa, cas de ser a l'alça es diu que ha tancat una posició llarga i si es a la baixa que ha tancat una posició curta.

- Barres de preus: Element que conforma el gràfic de preus d'un valor, la seva magnitud reflexa els preus màxim i mínim del espai temporal concret.

- Espelmes (candlesticks): Es un concepte força similar a la barra de preus, però que reforça l'análisis de l'orientació psicològica del mercat, especialment al curt plac. Te el seu origen el segle XVII el Japò.

- Ratlles de tendència: Son ratlles que dibuixa l'inversor, sobre o sota, la qual és mou el preu d'un valor.

- Ratlles de suport: Son ratlles que dibuixa l'inversor, on acostuma a rebotar a l'alça el preu d'un valor.

- Ratlles de resistència: Son ratlles que dibuixa l'inversor, on acostuma a rebotar a la baixa el preu d'un valor.

- Retrocessos fibonacci: Hi ha llibres sencers que expliquen aquest concepte, la idea bàsica es quantificar en tant percent, el retrocés que experimentarà el preu d'un valor desprès d'un rebot a l'alça important.

8. BIBLIOGRAFIA CONSULTADA

CHUCK CAVANES. Jakarta Struts. Anaya Multimedia O'Reilly

CAMPDERRICH, BENET. *Enginyeria del programari; Anàlisi orientada a objectes* . UOC. (material de l'assignatura).

ALISTAIR COCKBURN. Writing Effective Use Cases. Adison Wesley

ARMSTRONG; BALL; BODOFF;...The J2EE 1.4 Tutorial. Sun Microsystems.

BOOCH, G; RUMBAUGHT,J:JACOBSON,I. *El lenguaje de modelado unificado. Manual de referencia.* Adison Wesley

CAMPDERRICH, BENET. *Enginyeria del programari; Anàlisi orientada a objectes* . UOC. (material de l'assignatura).

GREGORY BRILL. J2EE EJB, JDBC, JSP and Servlets. Random House

FLOYD MARINESCU. *EJB Design Pattern.* Wiley & Sons

DEEPAK ALUR; JOHN CRUPI; DAN MALKS. Core J2EE Patterns. Sun Microsystems

JAMES MCGOVERN; RAHIM ADATIA; YAKOV FAIN;JASON GORDON; ETHAN HENRY; WALTER HURST; ASHISH JAIN; MARK LITTLE; VAIDYANATHAN NAGARAJAN; HARSHAD OAK; LEE ANNE PHILLIPS. Java™ 2 Enterprise Edition 1.4 Bible

Josep M. Ganyet. Interacció Humana amb els Ordinadors. Apunts assignatura. UOC (material de l'assignatura Interacció Humana amb els ordinadors).

HUSTED, DUMOULIN, FRANCISCUS, WINTERFELDT: *Struts in Action. Building web applications with the leading Java framework* Ed. Manning 2004

9. ANNEX 1. SENTÈNCIES SQL

```
create database BORSAXTRADER;

create table PERSONA
(
	login VARCHAR (8),
	contrasenya VARCHAR (8),
	nom VARCHAR (20),
	cognoms VARCHAR (30),
	email VARCHAR (30),
	rol VARCHAR (15) CHECK ('administrador', 'inversor'),
	guanys DECIMAL(4,2),
	PRIMARY KEY (login, contrasenya)
);

create table CONNEXIO
(
	login VARCHAR (8) references PERSONA (login),
	contrasenya VARCHAR (8) references PERSONA (contrasenya),
	data_hora DATE,
	PRIMARY KEY (login, contrasenya, data_hora)
);

create table PROVEIDOR
(
	id_proveidor INTEGER,
	nom VARCHAR (15),
	url TEXT,
	qualitat DECIMAL(5,2) CHECK (qualitat BETWEEN 0.00 AND 100.00),
	PRIMARY KEY (id_proveidor)
);

create table CARTERA
(
	login VARCHAR (8) references PERSONA (login),
	contrasenya VARCHAR (8) references PERSONA (contrasenya),
	id_cartera INTEGER,
	nom VARCHAR (15),
	data DATE,
	comentarI TEXT,
	PRIMARY KEY (login, contrasenya, id_cartera)
);

create table OPERACIO
(
	login VARCHAR (8) references PERSONA (login),
	contrasenya VARCHAR (8) references PERSONA (contrasenya),
	id_cartera INTEGER references CARTERA (id_cartera),
	id_operacio INTEGER,
	nom_empresa VARCHAR (15) references PRODUCTEFINANCER (nom),
	preu DECIMAL(5,2),
	num_titols INTEGER,
	data DATE,
	tipus VARCHAR (15) CHECK ('compra', 'venta'),
	lim_sup DECIMAL(5,2),
	lim_inf DECIMAL(5,2),
	PRIMARY KEY (login, contrasenya, id_cartera,id_operacio)
);
```

```
create table INTRADAY
(
	ticker VARCHAR(5) references PRODUCTEFINANCER (ticker),
	id_preu_itd INTEGER,
	preu DECIMAL(5,2),
	data DATE,
	PRIMARY KEY (ticker, id_preu_itd)
);

create table EOD
(
	ticker VARCHAR(5) references PRODUCTEFINANCER (ticker),
	id_preu_EOD INTEGER,
	preu DECIMAL(5,2),
	data DATE,
	open DECIMAL(5,2),
	close DECIMAL(5,2),
	max DECIMAL(5,2),
	min DECIMAL(5,2),
	volum INTEGER,
	PRIMARY KEY (ticker, id_preu_EOD)
);

create table PRODUCTEFINANCER
(
	nom VARCHAR (15),
	ticker VARCHAR (5),
	PRIMARY KEY (ticker)
);

create table INDEXS
(
	ticker_i VARCHAR (5) references PRODUCTEFINANCER (ticker),
	ticker_e VARCHAR (5) references PRODUCTEFINANCER (ticker),
	PRIMARY KEY (ticker_i, ticker_e)
);
```