

Gestió d'un catàleg de productes

Maties Rebassa Frau
Enginyeria Tècnica Informàtica de Sistemes
Universitat Oberta de Catalunya
Consultor: David Gañan Jimenez
11 de juny de 2007

Resum

L'objectiu principal d'aquest projecte era utilitzar alguna de les tècniques de programació proposades pel seu caràcter novedós dintre del area de .NET, a més, es clar, de posar de manifest els coneixements adquirits al llarg de la carrera.

El projecte obté con a resultat la **Gestió d'un catàleg de productes** sobre la plataforma .NET de Microsoft, obtenint un executable per entorns d'escriptori Windows. Aquest catàleg de productes, podria servir per a qualsevol aplicació de gestió comercial o d'emmagatzematge, on els productes són un dels pilars de l'activitat de l'empresa.

Així doncs, prenent de partida la plataforma integrada .NET Framework de Microsoft, s'ha aprofundit amb la utilització d'un patró MVC per el desenvolupament d'aplicacions amb WinForms, el *User Interface Process Application Block* de Microsoft.

Amb la utilització d'aquest patró *Model-Vista-Controlador*, s'obté una programació multicapa on fàcilment es poden desenvolupar diferents vistes per a diferents entorns de treball, Windows, web, dispositius mòbils, etc., on només cal dissenyar les vistes segons l'entorn de treball i la resta de capes son reutilitzades i compartides per els diferents entorns.

En la capa de persistència, s'ha utilitzat Microsoft SQL server 2005, amb procediments emmagatzemats a la mateixa base de dades, per tal d'optimitzar les consultes i independitzar-les de l'aplicatiu, permetent així modificar i parametritzar aquestes consultes sense tenir que tocar el codi de l'aplicació.

Índex

Resum	2
Índex.....	3
Índex de figures	5
Introducció.....	7
<i>Justificació del TFC.....</i>	<i>7</i>
<i>Objectius del TFC.....</i>	<i>7</i>
<i>Enfocament i mètode seguit.....</i>	<i>8</i>
<i>Planificació del projecte</i>	<i>8</i>
<i>Diagrames de Gantt.....</i>	<i>9</i>
Pla de treball.....	9
PAC 2.....	9
PAC 3.....	10
Lliurament final i debat virtual	10
<i>Productes obtinguts</i>	<i>11</i>
<i>Descripció dels altres capítols de la memòria</i>	<i>11</i>
Infraestructura de l'aplicació.....	12
<i>Arquitectura Microsoft .NET Framework.....</i>	<i>12</i>
Descripció.....	12
Common Language Runtime (CLR)	13
Biblioteca de classe de .NET	13
<i>Patró MVC UIP Application Block</i>	<i>14</i>
<i>Detall del mètode seguit en el projecte.....</i>	<i>16</i>
<i>Descripció de funcionalitats.....</i>	<i>18</i>
Anàlisi de requeriments	18
Diagrama de casos d'ús	18
Detall dels casos d'ús principals	19
Cas d'ús Consulta Catàleg	19
Cas d'ús Alta Producte	19
Cas d'ús Modificació Producte	19
Cas d'ús Consulta Producte	20
Cas d'ús Baixa Producte.....	20
Cas d'ús Selecciona Producte.....	20
<i>Model conceptual de classes.....</i>	<i>21</i>
Diagrama de classes de les classes entitat del problema	21
Diagrama de classes de les classes gestores	22

Disseny	23
<i>Diagrames de seqüències</i>	23
<i>Disseny de la base de dades</i>	26
Model ER	26
Descripció de les taules.....	27
Taula usuaris	27
Taula Categories	27
Taula Marques	27
Taula Proveïdors	28
Taula Productes	28
Procediments emmagatzemats	29
Taula Usuaris.....	29
Taula Marques	29
Taula Categories	29
Taula Proveïdors	30
Taula Productes	30
Implementació	31
<i>Eines</i>	31
Visual Studio 2005	31
Microsoft SQL Server 2005.....	33
Microsoft SQL Server Management Studio Express.....	33
<i>Estructura</i>	34
El model.....	34
El controlador.....	34
La vista, disseny de la capa de presentació.....	34
<i>Disseny dels formularis</i>	35
Accés	35
Navegació del gestor	36
Formulari Menu.....	36
Gestió de Marques.....	38
Gestió de Categories.....	39
Gestió de Proveïdors.....	40
Gestió de Productes.....	41
Selector	44
Botonera.....	50
Navegació del Client.....	51
Conclusions	57
Glossari	58
Bibliografia	59
Annexos	60
<i>Annex A. Instruccions per a la instal·lació</i>	60
<i>Annex B. Instruccions d'ús</i>	60

Índex de figures

Figura 1. Fases de desenvolupament del projecte.....	8
Figura 2. Taula de dates claus.....	8
Figura 3. Diagrama de Gantt Pla de treball.....	9
Figura 4. Diagrama de Gantt PAC 2.....	9
Figura 5. Diagrama de Gantt PAC 3.....	10
Figura 6. Diagrama de Gantt Lliurament final.....	10
Figura 7. Diagrama de la estructura de .NET Framework.....	12
Figura 8. Biblioteca de classes de .NET.....	13
Figura 9. Capes de components de .NET amb UIP Application Block.....	15
Figura 10. Disseny de UIP Application Block.....	15
Figura 11. Esquema del patró MVC.....	16
Figura 12. Detall fitxer de configuració de l'aplicació.....	16
Figura 13. Fitxer de configuració de l'aplicació "TFCNet.exe.config".....	17
Figura 14. Casos d'us generals.....	18
Figura 15. Model conceptual de classes entitat del problema.....	21
Figura 16. Diagrama de classes de les entitats gestores.....	22
Figura 17. Codi de la classe <i>BaseCAD</i>	22
Figura 18. Diagrama de seqüència de Alta Producte.....	23
Figura 19. Diagrama de seqüència de Baixa Producte.....	24
Figura 20. Diagrama de seqüència Alta Marca.....	24
Figura 21. Diagrama seqüència Baixa Categoria.....	25
Figura 22. Diagrama ER de la base de dades.....	26
Figura 23. Explorador de solucions de Visual Studio 2005.....	32
Figura 24. Microsoft SQL Server Mangement Studio Express.....	33
Figura 25. Formulari d'accés a l'aplicació.....	35
Figura 26. Codi de la classe inicial de l'aplicació.....	35
Figura 27. Codi que inicia les dues possibles navegacions.....	36
Figura 28. Formulari Menú d'administració.....	37
Figura 29. Codi del formulari Menu.....	37
Figura 30. Codi del controlador que gestiona el menú de l'administrador.....	38
Figura 31. Formulari Gestió de Marques.....	39
Figura 32. Formulari Gestió de Categories.....	39
Figura 33. Formulari Gestió de Proveïdors.....	40
Figura 34. Formulari Gestió de Proveïdors amb notificació d'errors.....	41
Figura 35. Formulari Gestió de Productes.....	42
Figura 36. Codi per carregar un imatge des de arxiu.....	43
Figura 37. Ampliació de la imatge d'un producte.....	43
Figura 38. Formulari de selector des de la pròpia gestió.....	44
Figura 39. Fragment de codi del filtratge del selector.....	45
Figura 40. Formulari de selector amb filtratge.....	45
Figura 41. Formulari de selector de marques activat des de gestió de productes.....	46
Figura 42. Codi d'activació del selector de Marques amb altes.....	47
Figura 43. Codi d'activació del selector de Marques amb altes.....	47
Figura 44. Codi dels mètodes delegats del controlador.....	48
Figura 45. Codi de inicialització de la classe selector.....	49
Figura 46. Missatge de confirmació de baixes.....	50
Figura 47. Formulari del Catàleg de productes.....	51
Figura 48. Arbre de selecció.....	52
Figura 49. Llista de productes segons la selecció.....	52
Figura 50. Dades del producte.....	53

Figura 51. Catàleg de productes per Marca.....	54
Figura 52. Catàleg de productes d'una Categoria.....	55
Figura 53. Catàleg de productes d'una determinada Categoria i Marca.	56

Introducció

Justificació del TFC

En qualsevol aplicació comercial per a portar la gestió de les compres i vendes d'una empresa hi trobem que un dels punts principals és la gestió del catàleg de productes. Aquest productes o serveis, podran variar amb algunes característiques però el concepte de la seva gestió sempre serà el mateix.

Partint d'aquest context, on gairebé totes les empreses tenen la necessitat de tenir un catàleg dels productes o serveis que comercialitzen i/o ofereixen als seus clients, es pot veure clarament que aquest TFC tracta un tema àmpliament conegut, que si bé no té una gran quantitat de funcionalitats ni un sector específic d'aplicació, si que és un tema fàcilment adaptable en el món real de les aplicacions comercials.

En el desenvolupament d'aquest projecte, s'ha posat especial interès en potenciar el motor de cerca de les diferents objectes que implica el manteniment del catàleg, incorporant diferents sistemes per tal que les cerques siguin lo més intuïtives possible i potents.

Aquest TFC també ha servit per a aprofundir amb l'aprenentatge de la plataforma .NET de Microsoft amb el llenguatge de programació C#.

Objectius del TFC

L'objectiu principal del TFC és aprofundir en la plataforma de Microsoft .NET utilitzant diferents tecnologies, per tal d'aconseguir aquest objectiu s'ha vinculat a un objectiu secundari, la creació d'un catàleg de productes que constituïria una part de qualsevol aplicació comercial.

Mes concretament, l'eix principal del projecte serà la creació d'una aplicació amb WinForms amb la utilització del patró MVC de Microsoft *User Interface Process Application Block* versió 2.0 (a partir d'ara UIP), amb la utilització d'aquesta tecnologia de programació, es pretenen assolir els dos objectius aconseguint no només aprofundir en la plataforma NET, si no també amb la utilització i integració d'altres tecnologies que ens permetrà de desenvolupar unes solucions de més qualitat i de menor cost enfront de modificacions o ampliacions futures.

Un altre objectiu, en aquest cas personal, és el de aconseguir el domini suficient de la tecnologia .NET per a poder utilitzar-la en el camp professional desenvolupant solucions de qualitat amb una programació multicapa hi amb les possibilitats que ofereix una tecnologia robusta i aplicable a diferents entorns de treball com són els entorns Web, o els dispositius mòbils (pockets PC, Smartphones, etc).

Enfocament i mètode seguit

L'enfocament i el mètode seguit es basa amb l'estàndard del desenvolupament d'aplicacions, es a dir, en les fases d'anàlisi, disseny, implementació i proves. Cal remarcar que en tractar-se d'un TFC, fases com el manteniment no es duren a terme, com seria normal en una aplicació comercial.

Així, les fases en les que s'ha dividit el projecte es pot veure a la figura següent :

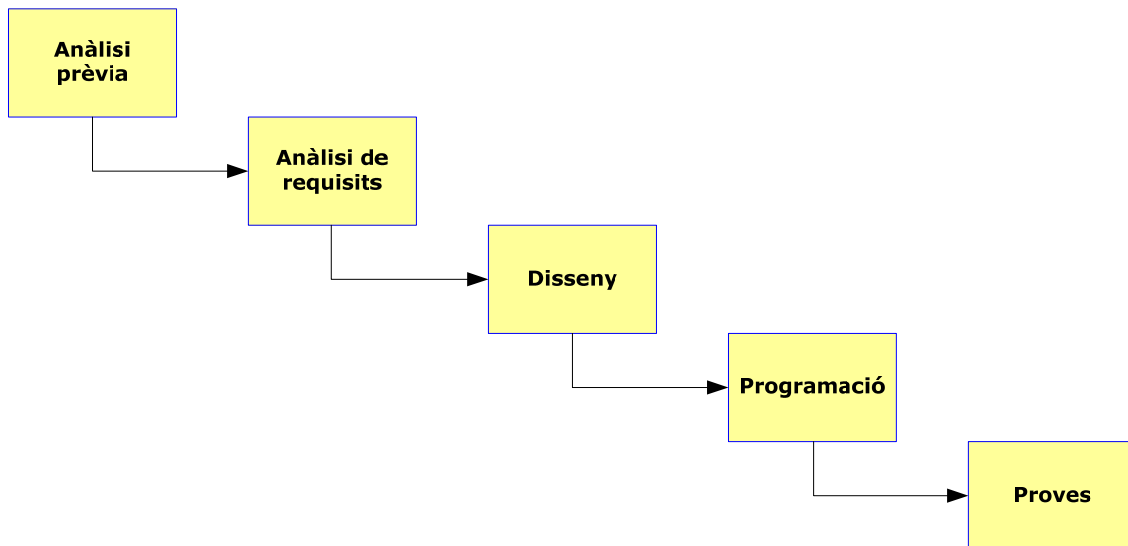


Figura 1. Fases de desenvolupament del projecte.

Planificació del projecte

Les dates d'entrega de les diferents fases incloses en el TFC són les següents

Títol	Data entrega
Esborrany pla de treball	11/03/2007
Pla de treball	13/03/2007
Esborrany PAC 2	31/03/2007
PAC 2	09/04/2007
Esborrany PAC 3	22/05/2007
PAC 3	28/05/2007
Esborrany lliurament final	05/06/2007
Lliurament final	11/06/2007

Figura 2. Taula de dates claus.

Diagrames de Gantt

La representació mitjançant els diagrames de Gantt s'ha dividit en apartats per tal d'augmentar la seva visibilitat.

Pla de treball

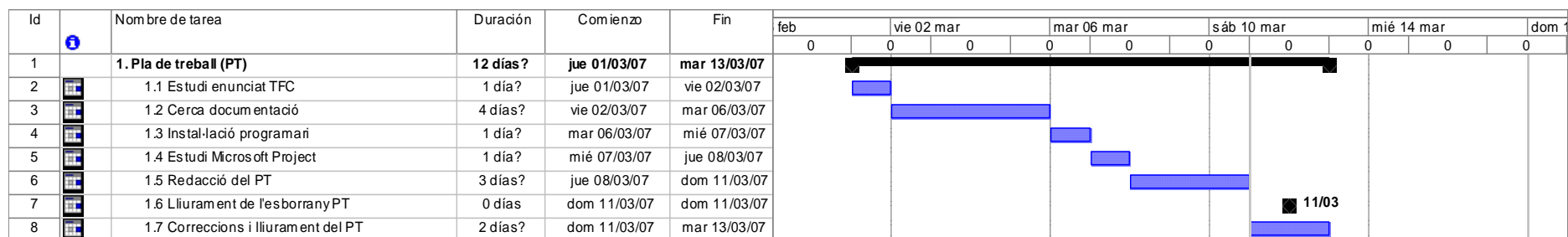


Figura 3. Diagrama de Gantt Pla de treball.

PAC 2

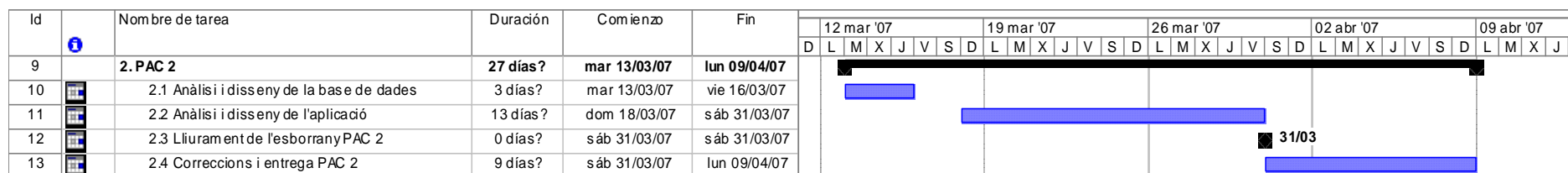


Figura 4. Diagrama de Gantt PAC 2.

PAC 3

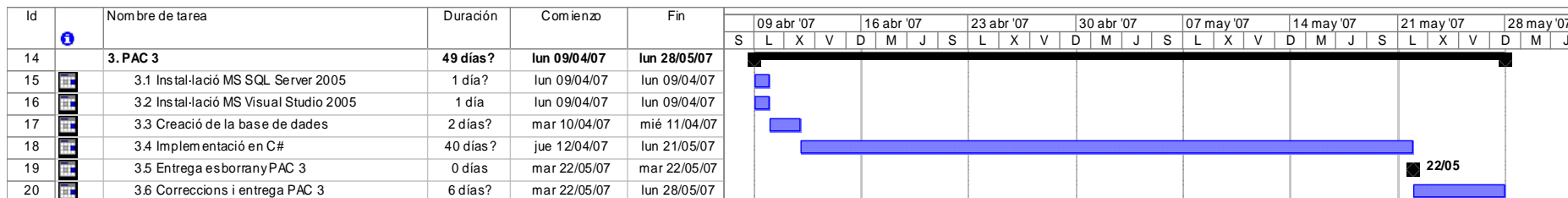


Figura 5. Diagrama de Gantt PAC 3.

Lliurament final i debat virtual

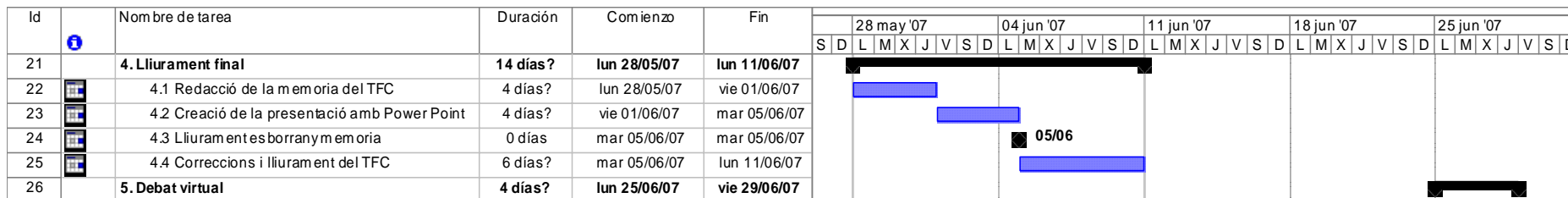


Figura 6. Diagrama de Gantt Lliurament final.

Productes obtinguts

Un cop finalitzat el projecte s'han obtingut els següents productes:

- Pla de treball. Breu descripció del projecte amb la planificació temporal prevista per a la realització del projecte.
- Anàlisi. Conjunt de requisits y funcionalitats representants mitjançant casos d'us.
- Disseny. Diagrama de classes i disseny de la base de dades.
- Implementació. Aplicació Windows totalment operativa amb totes les funcionalitats previstes implementades.
- Memòria del projecte. Document final que resumeix tota la feina realitzada.
- Presentació. Document multimèdia que complementa a la memòria del projecte i que destaca els punts més importants donant una visió global del projecte.

Descripció dels altres capítols de la memòria

Infraestructura de l'aplicació. En aquest capítol s'explica que és .NET Framework i el patró MVC utilitzat per el desenvolupament del TFC.

Disseny. En aquest capítol es veuen totes les passes de la fase de disseny de l'aplicació, es veuen les relacions entre els objectes de l'aplicació així com les relacions entre les taules definides a la base de dades.

Implementació. S'expliquen les etapes de la implementació del disseny, explicant les eines utilitzades, l'estructura utilitzada en el desenvolupament de l'aplicació i tots el formularis de la interfície gràfica d'usuari.

Infraestructura de l'aplicació

Arquitectura Microsoft .NET Framework

Descripció

El framework de .NET es una infraestructura sobre la que es reuneix tot un conjunt de llenguatges de programació i serveis que simplifiquen enormement el desenvolupament d'aplicacions. Mitjançant aquesta eina, s'ofereix un entorn d'execució altament distribuït, que permet crear aplicacions robustes i escalables. Els principals components d'aquest entorn són:

- Llenguatges de compilació.
- Biblioteca de classes de .NET.
- CLR (Common Language Runtime)

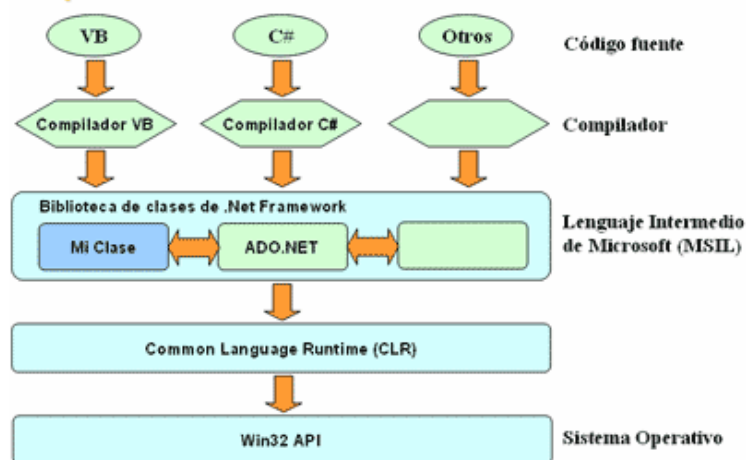


Figura 7. Diagrama de la estructura de .NET Framework.

.Net Framework suporta múltiples llenguatges de programació, tot i que cada llenguatge té les seves característiques pròpies, és possible desenvolupar qualsevol tipus d'aplicació en qualsevol d'aquests llenguatges. Existeixen més de 30 llenguatges adaptats a .NET, des de els més coneguts com C#, Visual Basic o C++ fins a llenguatges menys coneguts com Perl o Cobol.

Common Language Runtime (CLR)

El CLR és el vertader nucli del Framework .NET, ja que és l'entorn d'execució en el que es carregen les aplicacions desenvolupades amb els distints llenguatges, ampliant el conjunt de serveis que ofereix el sistema operatiu.

El codi font es compila i independentment del llenguatge utilitzat genera un mateix codi, denominat codi intermedi (MSIL, Microsoft Intermediate Language). Aquest codi intermedi no és codi màquina sinó que necessita d'una segona passa amb un compilador JIT "Just-In-Time" que genera el codi màquina real que s'executa en cada plataforma, aconseguint així certa independència de la plataforma de maquinari ja que pot tenir el seu propi compilador JIT i crear el codi màquina necessari a partir del codi MSIL.

Biblioteca de classe de .NET

Molts cops quan s'està programant una aplicació, es necessiten realitzar accions com manipular arxius, accés a dades, conèixer l'estat del sistema, implementar seguretat, etc. El Framework organitza tota la funcionalitat del sistema operatiu en un espai de noms jeràrquic de manera que a l'hora de programar resulti bastant senzill trobar el que es cerca.

Per això, .NET posseeix un sistema de tipus universals, denominats Common Type System (CTS). Aquest sistema permet que el programador pugui interactuar els tipus que s'inclouen en el propi Framework (biblioteca de classes de .NET) amb les creades per ell mateix (classes). D'aquesta manera s'aprofiten les avantatges pròpies de la programació orientada a objectes, com l'herència de classes predefinides per a crear-ne de noves, o com el polimorfisme de classes per a modifica o ampliar funcionalitats de les classes ja existents.

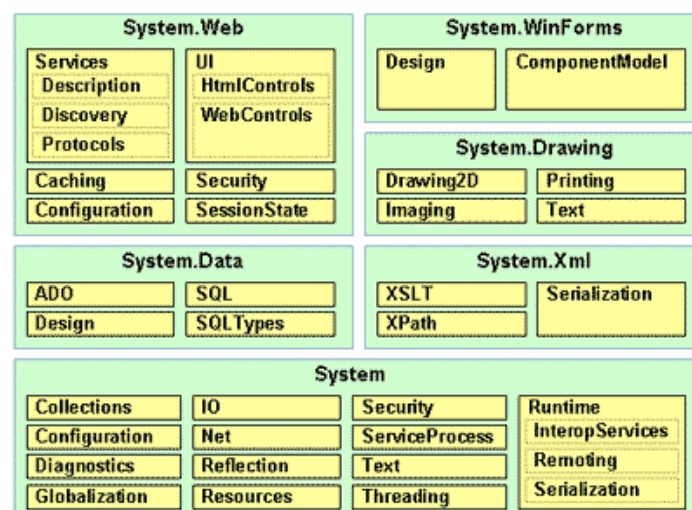


Figura 8. Biblioteca de classes de .NET.

Patró MVC UIP Application Block

En el moment de plantejar el desenvolupament del projecte es va decidir per realitzar el projecta en Window Forms i amb patrons MVC, després de cercar diferents opcions només es va trobar el User Interface Process Application Bloc (a partir d'ara UIP) versió 2.0, suportat per Microsoft.

Amb aquest patró s'aconsegueix un gran nivell d'independència de la plataforma ja que UIP proveeix un framework extensible que simplifica la tasca de separar el codi de la lògica de negoci de la interfície de l'usuari. Amb aquest patró es poden escriure complexes navegacions d'interfícies d'usuari i workflows que poden ser reutilitzats en múltiples escenaris i que poden ser ampliat a mesura que l'aplicació evolucioni.

L'UIP ofereix tot un set de funcionalitats entre les que s'inclouen:

- El control de la navegació i el workflow. Aquest control no està inclòs en la interfície de l'usuari, sinó que es basa en la lògica de negoci.
- Canvis en la navegació i workflow. Simplifica enormement el canviar l'ordre de les pàgines o formularis de la interfície d'usuari o afegir-ne de noves, ja que bassa aquesta navegació en un fitxer de configuració en format XML fàcilment modificable.
- Maneig de l'estat. Permet passar l'estat i mantenir-ne la consistència independentment de que l'aplicació estigui basada en Windows Forms o aplicacions Web.
- Guardar una instantània del procés actual. Es pot captura una instantània i recrear-la en qualsevol moment a través del temps, del maquinari o del usuari, permetent recuperar la mateixa sessió en el mateix punt on s'havia deixat.

Amb aquest framework ens permet d'abstreure la capa de presentació de l'aplicació en dues capes separades:

- User Interface Components (UIC). Aquest components configuren la interfície de l'usuari de l'aplicació. L'usuari veu i interacciona amb aquest components, són els responsables de adquirir les dades del usuari i de mostrar les dades cap a l'usuari.
- User Interface Process Components (UIP). Aquests components combinen els elements de la interfície d'usuari i coordinen per sota les activitats a seguir, com ara la navegació entre pàgines o el control del workflow. L'usuari no veu aquest components de procés, però són els qui conformen el suport vital als components de la interfície d'usuari (UIC).

El següent gràfic es poden veure aquestes dues capes en la arquitectura de .NET.

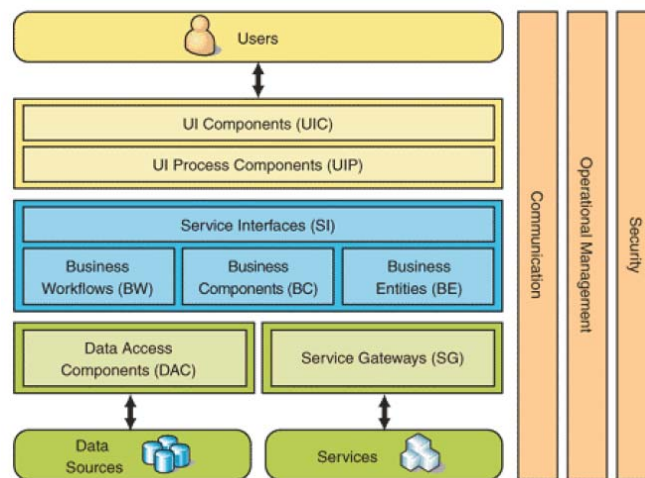


Figura 9. Capes de components de .NET amb UIP Application Block.

Amb la utilització del UIP és pot aconseguir que tot el codi de la navegació i del workflow són fora de la capa de presentació, això permet d'afegir noves vies, formularis o modificar les existents, i reutilitzar tota la resta de l'aplicació, simplificant d'aquesta manera les tasques de manteniment i/o ampliació de les aplicacions, y permetent una fàcil portabilitat de l'aplicació ja que només es necessari de dissenyar les noves vies en la nova plataforma i reutilitzant la resta de capes.

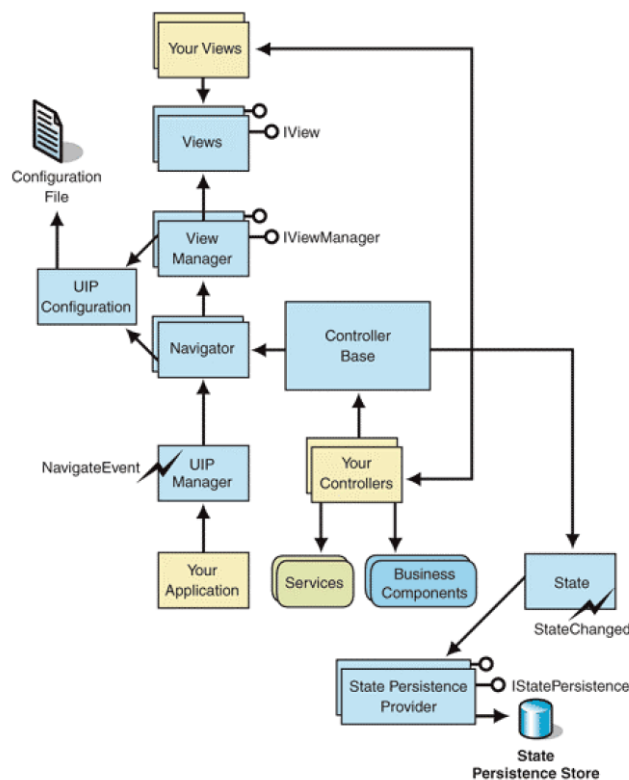


Figura 10. Disseny de UIP Application Block.

Detall del mètode seguit en el projecte

Tal i com ja s'ha comentat anteriorment, per tal de poder desenvolupar el projecte s'ha utilitzat una arquitectura *Model-Vista-Controlador* (MVC), utilitzat el patró User Interface Process Application Block versió 2.0 suportat per Microsoft (UIP).

L'objectiu ha estat separar al màxim possible la vista, el controlador i el model.

En primer lloc tindrem la *Vista* que correspon al disseny de la interfície gràfica que veurà l'usuari.

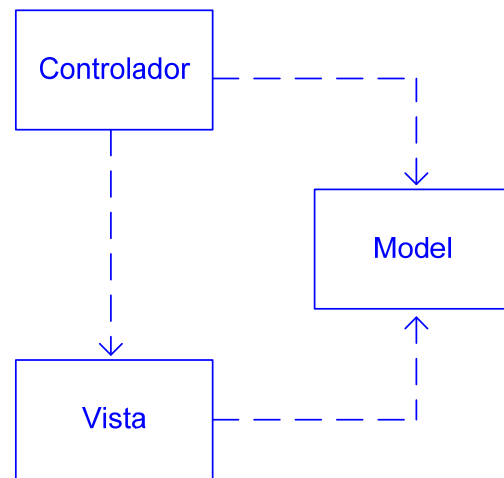


Figura 11. Esquema del patró MVC.

En segon lloc la programació del *Controlador*, és a dir, tot el codi que controla els esdeveniments quan l'usuari fa alguna acció sobre el formulari.

I finalment trobem el *Model*, que es pot anomenar el cor de l'aplicació, on s'emmagatzema tota la lògica de negoci, mètodes per accedir a les dades i altres procediments pel bon funcionament de l'aplicació.

En el nostre cas, al utilitzar l'UIP Application Block, mitjançant el fitxer de configuracions de l'aplicació en format XML, es pot indicar per a cada vista quin serà el seu controlador. Donades les poques funcionalitats a cobrir per el projecte, s'ha optat per un sol controlador per a totes les vistes al estar fortament relacionades entre elles.

```
<controller
  name="TfcControlador"
  type="UIProcessTFC.TfcControlador, UIProcessTFC.Comunes,
      Version=1.0.0.0,Culture=neutral,PublicKeyToken=null" />
.../...
</objectTypes>

<views>

  <view
    name="menu"
    type="UIProcessTFC.WinUI.menu, UIProcessTFC.WinUI,
        Version=1.0.1.0,Culture=neutral,PublicKeyToken=null"
    controller="TfcControlador"
    stayOpen="true"/>

  <view
    name="Marques"
    type="UIProcessTFC.WinUI.marcaView, UIProcessTFC.WinUI,
        Version=1.0.1.0,Culture=neutral,PublicKeyToken=null"
    controller="TfcControlador"
    openModal="true"/>
.../...
```

Figura 12. Detall fitxer de configuració de l'aplicació.

Com es pot observar en la figura anterior, per tal de que el framework del UIP pugui treballar, s'han de definir quines classes son els controladors i també cal, indicar a cada vista qui és el seu controlador.

Per a cada element que pren part en el procés, cal donar-li un nom i mitjançant la clau *type* indicar el nom de la classe, el projecte en el que es troba i característiques de versió i idioma. A més, a les vistes s'ha d'indicar qui es el seu controlador i es poden definir condicions de visualització diferents per a cada vista.

Aquest fitxer de configuracions pot ser fàcilment modificable per a qualsevol editor de text i així podem reconfigurar la navegació de les vistes o quina classe vista s'ha d'executar en cada moment, sense tenir que tocar el codi de l'aplicació.

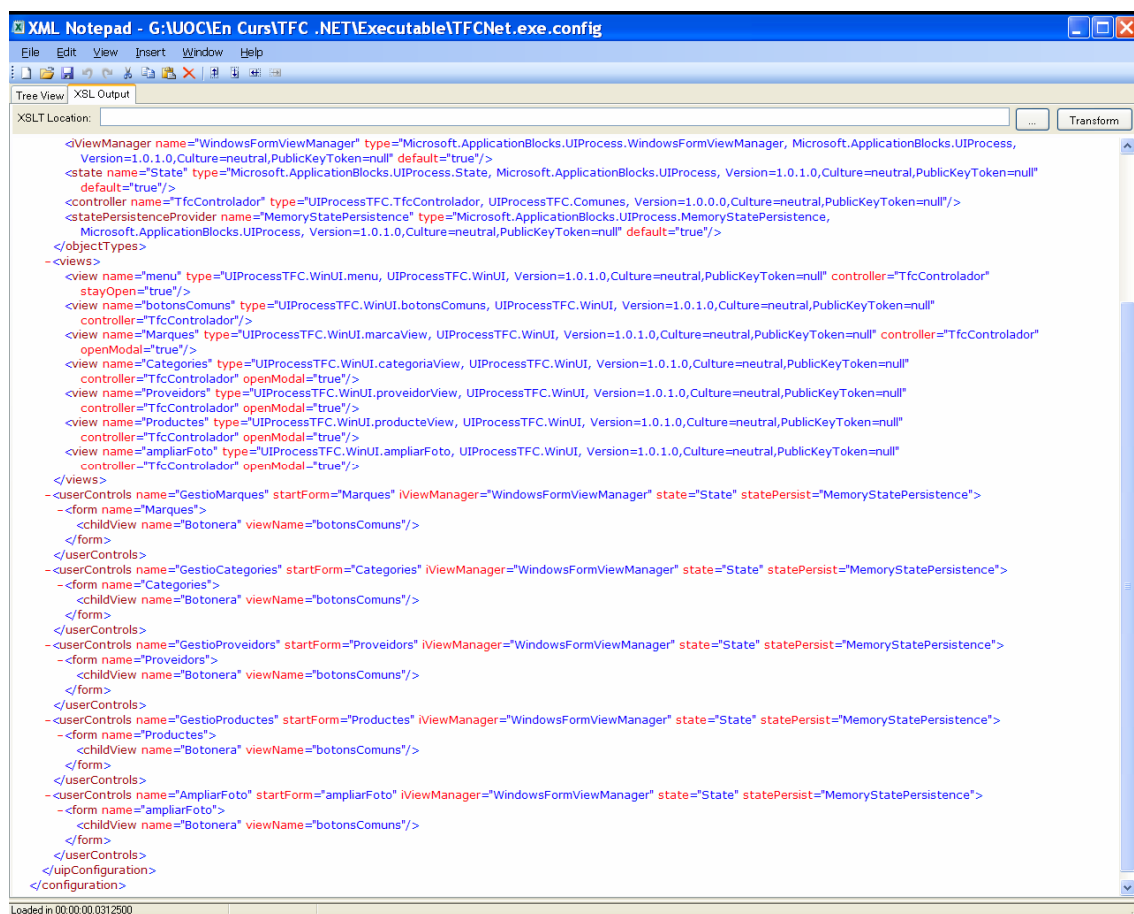


Figura 13. Fitxer de configuració de l'aplicació "TFCNet.exe.config".

Descripció de funcionalitats

Anàlisi de requeriments

Per portar a terme el manteniment del catàleg de productes hi haurà un *Gestor* que serà l'usuari que podrà realitzar totes les funcions necessàries.

El productes tindran un codi identificatiu i una sèrie de dades que caldrà emmagatzemar per a després poder ser consultades.

Les funcions sobre el productes són: donar d'alta un producte nou, modificar les dades d'un producte existent, consultar les dades d'un producte i donar de baixa un producte.

Els productes s'agruparan per diferents criteris, com són: per categories o famílies, per marca i per proveïdor. El *Gestor* portarà el manteniment d'aquestes fitxes. Per a cada fitxa es tindrà que poder donar d'alta, modificar, consultar i donar de baixa, només es podrà donar de baixa una d'aquestes fitxes si no hi ha cap producte que la faci servir.

Un altre servei que ha d'oferir l'aplicació, és la possibilitat que els clients consultin aquest catàleg, però aquesta consulta serà diferent a la del *Gestor* ja que només tindrà que mostrar les dades que es considerin d'interès pels clients.

Diagrama de casos d'us

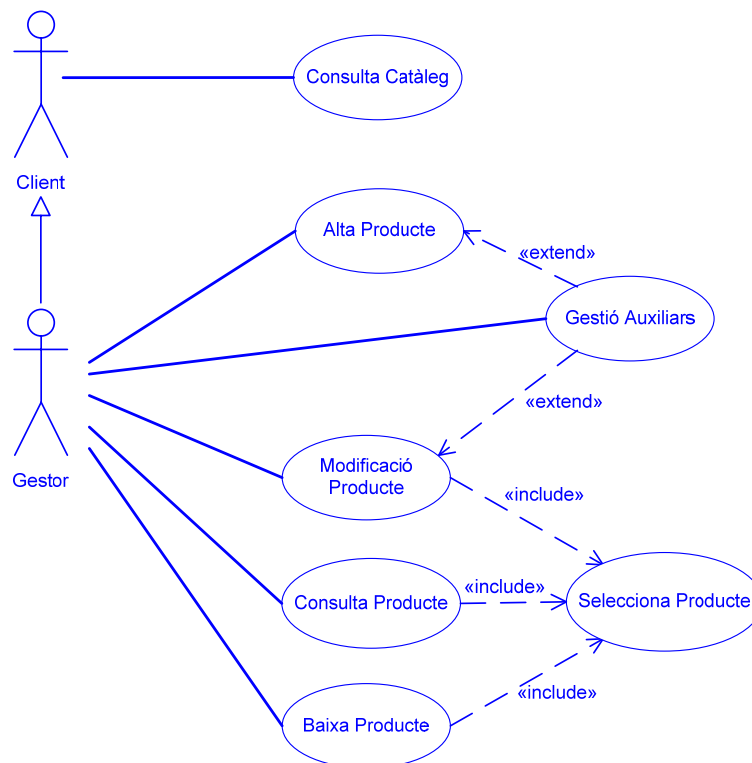


Figura 14. Casos d'us generals

Detall dels casos d'ús principals

Cas d'ús Consulta Catàleg

Resum de la funcionalitat: Realitza la consulta de les dades d'un producte.

Actors: **Client**, **Gestor**

Casos d'ús relacionats:

Precondició: el producte existeix

Postcondició:

Passos:

1. El sistema mostra una llista de productes.
2. L'usuari selecciona un producte de la llista del pas 1.
3. El sistema mostra les dades públiques del producte.

Cas d'ús Alta Producte

Resum de la funcionalitat: Dona d'alta un nou producte.

Actors: **Gestor**

Casos d'ús relacionats: Gestió Auxiliars

Precondició: el producte no existeix

Postcondició: el producte s'ha enregistrat

Passos:

1. El sistema demana les dades del producte..
2. L'usuari introdueix les dades indicades en el pas 1.

Cas d'ús Modificació Producte

Resum de la funcionalitat: Permet modificar les dades d'un producte.

Actors: **Gestor**

Casos d'ús relacionats: Gestió Auxiliars, Selecciona Producte

Precondició: el producte existeix

Postcondició: s'han enregistrat els canvis

Passos:

1. El sistema mostra una llista de productes.
2. L'usuari selecciona un producte de la llista del pas 1.
3. El sistema mostra les dades del producte.
4. L'usuari modifica les dades convenients.

Cas d'ús Consulta Producte

Resum de la funcionalitat: Permet visualitzar les dades d'un producte.

Actors: **Gestor**

Casos d'ús relacionats: Selecciona Producte

Precondició: el producte existeix

Postcondició:

Passos:

1. El sistema mostra una llista de productes.
2. L'usuari selecciona un producte de la llista del pas 1.
3. El sistema mostra les dades del producte.

Cas d'ús Baixa Producte

Resum de la funcionalitat: Permet modificar les dades d'un producte.

Actors: **Gestor**

Casos d'ús relacionats: Selecciona Producte

Precondició: el producte existeix

Postcondició: s'ha esborrat el producte

Passos:

1. El sistema mostra una llista de productes.
2. L'usuari selecciona un producte de la llista del pas 1.
3. El sistema mostra les dades del producte.
4. L'usuari esborra el producte del catàleg.

Cas d'ús Selecciona Producte

Resum de la funcionalitat: Selecciona un producte de la llista de productes.

Actors: **Gestor**

Casos d'ús relacionats: Modificació Producte, Consulta Producte, Baixa Producte

Precondició: el producte existeix

Postcondició: s'ha seleccionat un producte

Passos:

1. El sistema mostra una llista de productes donats d'alta en el sistema.
2. L'usuari selecciona un producte de la llista del pas 1.
3. El sistema mostra les dades del producte.

Model conceptual de classes

Les classes del model son totes aquelles que tenen que veure amb les dades del problema. Es divideixen en dos grups:

- Les classes que corresponen a les entitats del problema.
- Les classes gestores corresponents a les entitats que necessiten operacions en la base de dades. Aquestes classes s'anomenaran igual que les classes de l'entitat corresponent afegint CAD (capa d'accés a dades).

Diagrama de classes de les classes entitat del problema

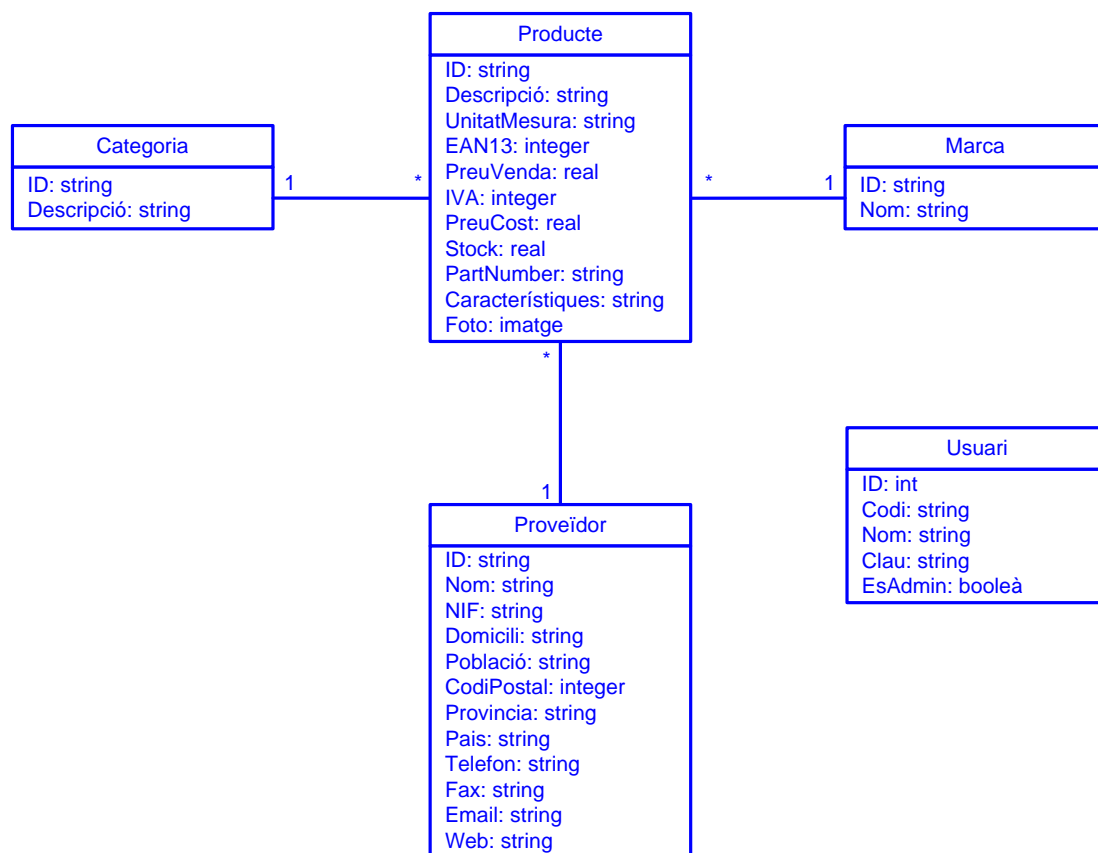


Figura 15. Model conceptual de classes entitat del problema

No s'han inclòs els constructors de classe per simplificar. Tots els atributs s'implementaran com a propietats.

Diagrama de classes de les classes gestores

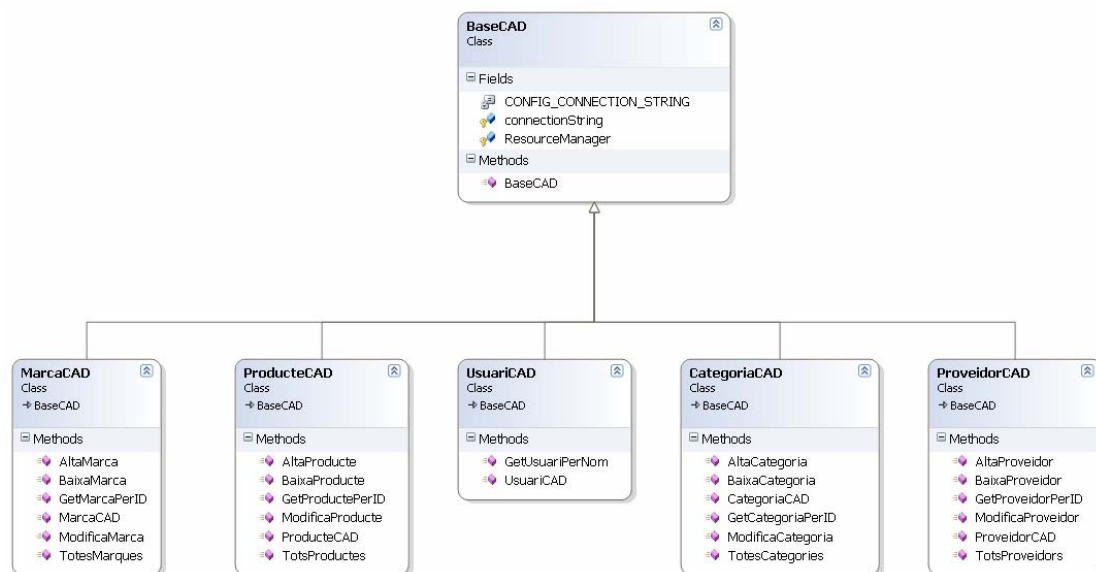


Figura 16. Diagrama de classes de les entitats gestores

Aquí es pot observar com totes les classes de la capa d'accés a dades, hereten de la classe BaseCAD, que implementa un conjunt d'atributs comuns necessaris pel funcionament del UIP, com pot ser el *connectionString* on conté la cadena de connexió amb la base de dades o el *ResourceManager*, fitxer de recursos on es defineixen els missatges d'error amb el idioma corresponent.

```

using System;
using System.Configuration;
using System.Reflection;
using System.Resources;
using System.Collections.Specialized;

namespace UIProcessTFC
{
    public class BaseCAD
    {
        private const string CONFIG_CONNECTION_STRING = "connectionString";

        /// <summary>
        /// Keep the connection string from the database
        /// </summary>
        protected string connectionString;

        protected static ResourceManager ResourceManager = new ResourceManager( typeof(
BaseCAD ).Namespace + ".CADText", Assembly.GetExecutingAssembly() );

        public BaseCAD()
        {
            NameValueCollection values =
(NameValueCollection)ConfigurationSettings.GetConfig( "appParams" );
            if (values == null)
                throw new ConfigurationException( ResourceManager.GetString(
"RES_ExceptionStoreConfigNotFound" ) );

            string connectionString = values[CONFIG_CONNECTION_STRING];
            if (connectionString == null)
                throw new ConfigurationException( ResourceManager.GetString(
"RES_ExceptionStoreConfigConnection" ) );

            this.connectionString = connectionString;
        }
    }
}

```

Figura 17. Codi de la classe BaseCAD.

Disseny

Diagrames de seqüències

A continuació es detallen alguns dels casos d'ús amb els seus corresponents diagrames de seqüències.

El cas d'ús més important és, sens dubte, l'alta del producte, que es detalla tot seguit.

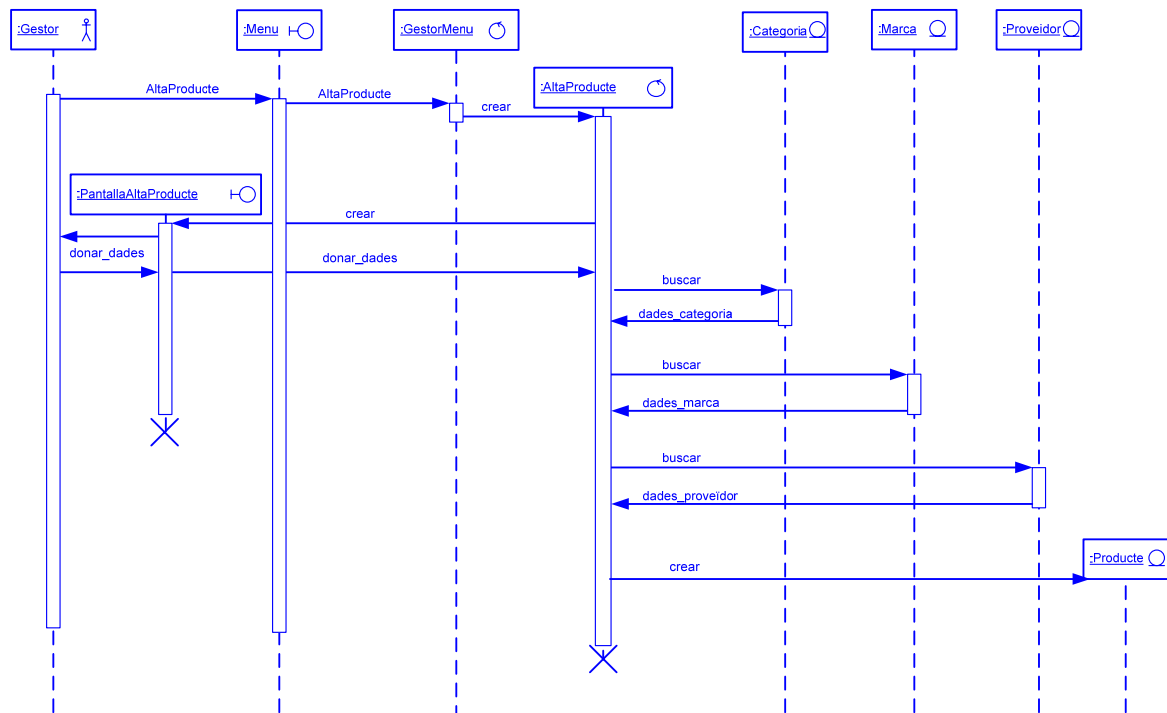


Figura 18. Diagrama de seqüència de Alta Producte

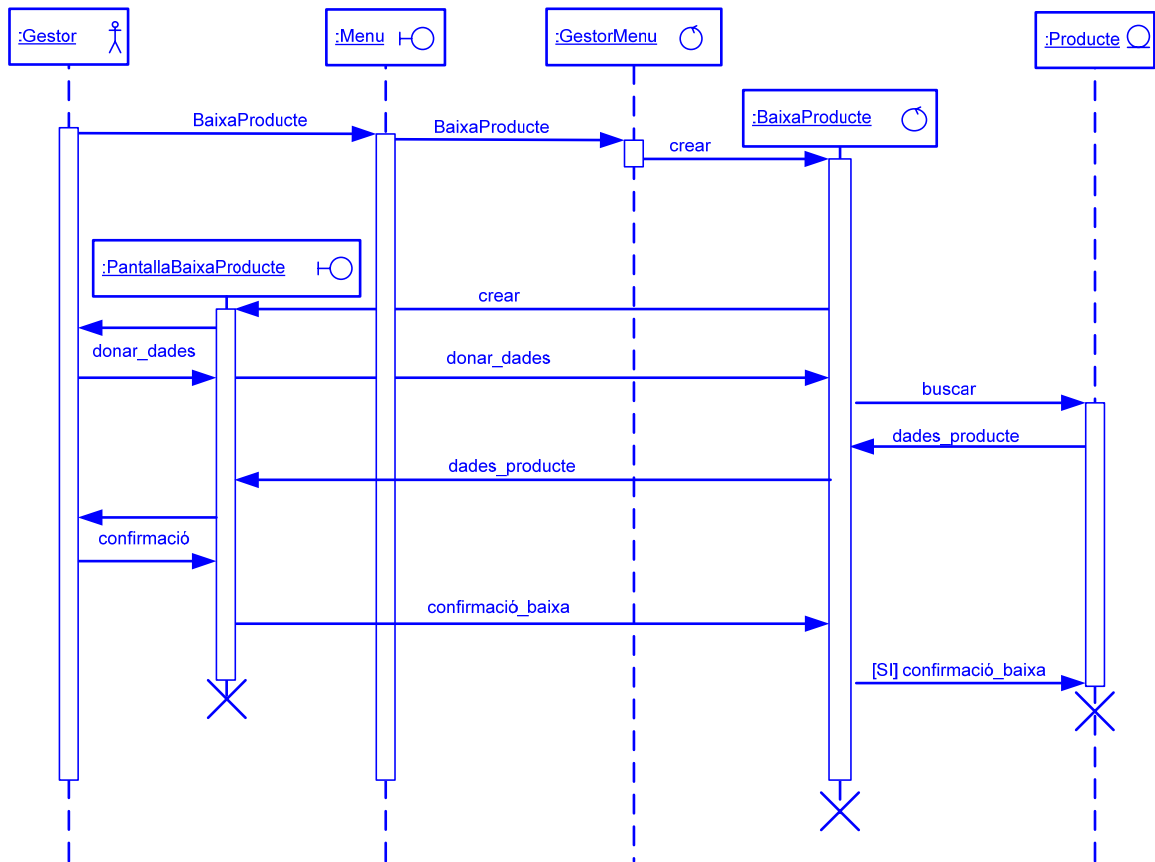


Figura 19. Diagrama de seqüència de Baixa Producte

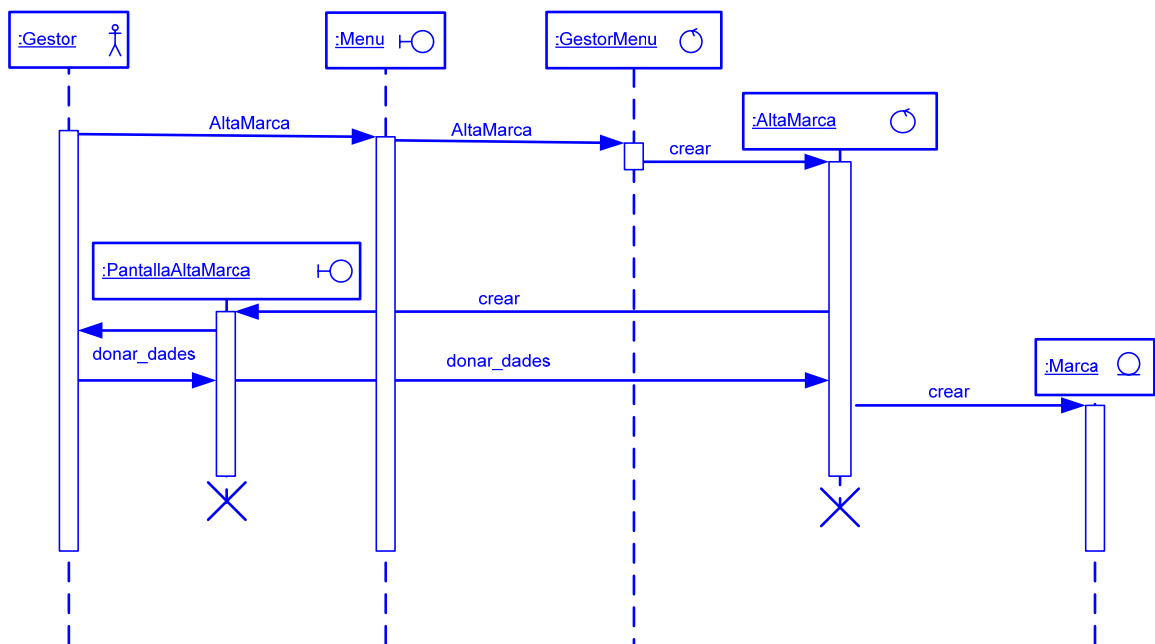


Figura 20. Diagrama de seqüència Alta Marca

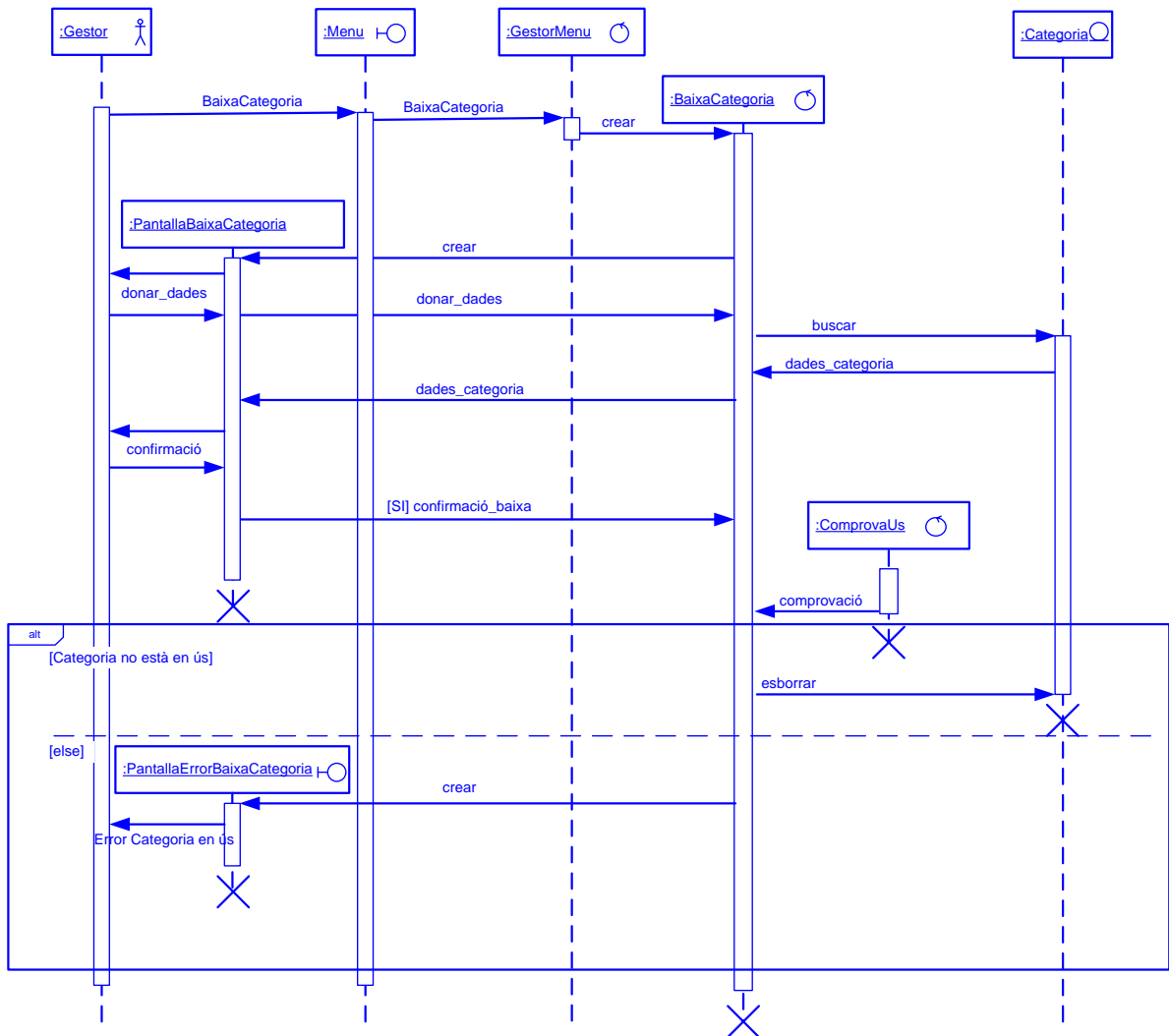


Figura 21. Diagrama seqüència Baixa Categoria

El casos d'ús mostrats són representatius de la resta de casos d'ús, ja que tant l'alta de categories o proveïdors com les baixes de cadascuna d'aquestes fitxes son iguals.

Disseny de la base de dades

Model ER

Per a crear la base de dades s'ha utilitzat el servidor de Base de Dades Microsoft SQL server 2005. S'ha optat per a crear una base de dades senzilla amb les taules necessàries pel funcionament de l'aplicació.

Tot seguit es mostra el diagrama entita-relació del model de dades obtingut en l'etapa de disseny.

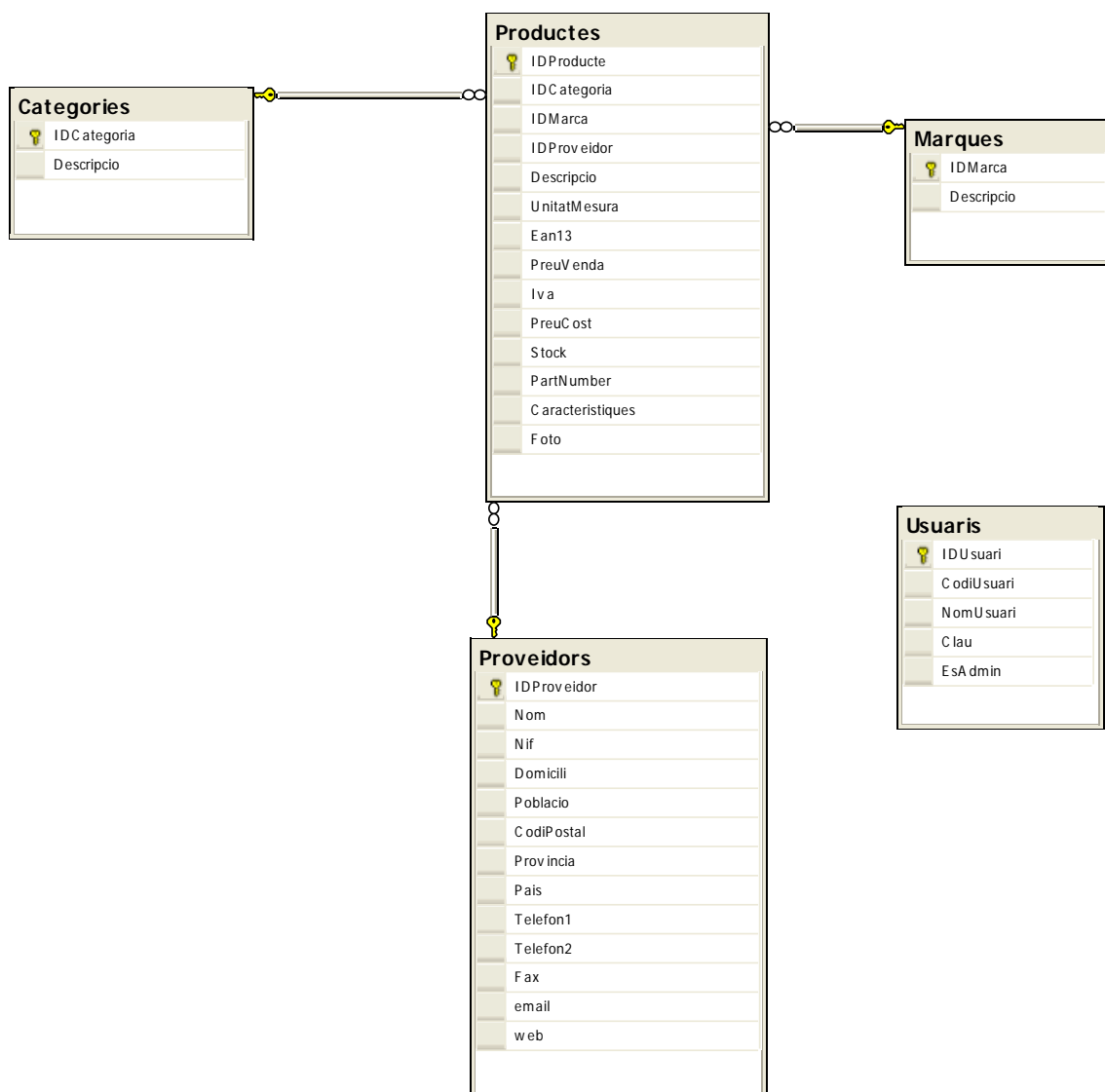


Figura 22. Diagrama ER de la base de dades.

Descripció de les taules

Tot seguit es passarà a mostrar en detall cadascuna de les taules amb la descripció de cada camp així com el tipus i si son claus primàries (PK) o claus foranes (FK) o si el camp pot ser nul (NULL).

Taula usuaris

La taula Usuaris és la que conté la informació dels usuaris del sistema, gestors i clients. Aquesta informació serà el nom del usuari, el seu codi d'accés, la clau y si l'usuari es un gestor o no.

Columna	Tipus	Clau/Nul	Descripció
IdUsuari	Int	PK	Identificador seqüencial del usuari
CodiUsuari	Varchar(50)		Permet identificar l'usuari en el sistema
NomUsuari	Varchar(100)		Nom real de l'usuari
Clau	Varchar(50)		Contrasenya de l'usuari per a poder accedir al sistema
EsAdmin	Bit		Booleà que permet saber si l'usuari es un gestor o un client

Taula Categories

En la taula Categories s'emmagatzemen les dades de les categories que serviran per poder classificar els productes.

Columna	Tipus	Clau/Nul	Descripció
IdCategoria	Varchar(10)	PK	Identificador de la categoria
Descripcio	Varchar(100)		Descripció de la categoria.

Taula Marques

En la taula Marques s'emmagatzemen les dades de les marques que serviran per poder classificar els productes.

Columna	Tipus	Clau/Nul	Descripció
IdMarca	Varchar(10)	PK	Identificador de la marca
Descripcio	Varchar(100)		Descripció de la marca.

Taula Proveïdors

En la taula Proveïdors s'emmagatzemen les dades dels proveïdors, aquest proveïdors serviran per a poder indicar en el producte quin es el proveïdor habitual del producte o bé el darrer proveïdor que .l'ha subministrat

Columna	Tipus	Clau/Nul	Descripció
IdProveïdor	Varchar(10)	PK	Identificador del proveïdor
Nom	Varchar(100)		Nom del proveïdor
Nif	Varchar(9)		Número d'identificació fiscal del proveïdor
Domicili	Varchar(50)	NULL	Domicili fiscal del proveïdor
Poblacio	Varchar(50)	NULL	Població del proveïdor
CodiPostal	Varchar(5)	NULL	Codi postal del proveïdor
Provincia	Varchar(50)	NULL	Província del proveïdor
Pais	Varchar(25)	NULL	País del proveïdor
Telefon1	Varchar(20)	NULL	Telèfon del proveïdor
Telefon2	Varchar(20)	NULL	Segon possible telèfon del proveïdor
Fax	Varchar(20)	NULL	Número del fax del proveïdor
email	Varchar(50)	NULL	Adreça de correu electrònic del proveïdor
web	Varchar(100)	NULL	Adreça URL de la pàgina Web del proveïdor

Taula Productes

En la taula Producte s'emmagatzemen les dades dels productes, taula principal de l'aplicació, el voltant de la qual es podran fer les cerques i ordenacions per diferents criteris.

Columna	Tipus	Clau/Nul	Descripció
IdProducte	Varchar(15)	PK	Identificador del producte
IDCategoria	Varchar(10)	FK	ID de la categoria a la que pertany el producte
IDMarca	Varchar(10)	FK	ID de la marca a la que pertany el producte
IDProveïdor	Varchar(10)	FK	ID del proveïdor, pot ser el proveïdor habitual del producte o bé el darrer proveïdor
Descripcio	Varchar(100)		Descripció del producte
UnitatMesura	Varchar(20)	NULL	Presentació del producte, pot ser unitats, kilos, metres, caixes, etc.
Ean13	Varchar(13)	NULL	Codi de barres del producte en format EAN13
PreuVenda	Money		Preu de venda del producte sense IVA
Iva	Int		Percentatge d'impost de valor afegit que suporta el producte
PreuCost	Money		Preu de cost del producte sense IVA
Stock	Numeric(10,2)	NULL	Quantitat de producte que tenim en els magatzems
PartNumber	Varchar(25)	NULL	Identificador del producte per part del fabricant
Caracteristiques	Varchar(8000)	NULL	Text lliure que permet de guardar les característiques principals del producte
Foto	Image	NULL	Foto del producte en format binari, permet guardar una imatge associada al producte

Procediments emmagatzemats

Aprofitant que la base de dades permet emmagatzemar procediments, s'han creat els mètodes necessaris per a accedir a les dades que empra el model de l'aplicació.

Així, també s'aconsegueix augmentar el rendiment de l'aplicació i s'allibera al programador de modificar el codi de l'aplicació per tal d'accedir a les dades de les diferents taules.

Els diferents procediments emmagatzemats són:

Taula Usuaris

- **SelectUsuariPerCodi.** Rep com a paràmetre el codi d'un usuari i retorna l'usuari si existeix.

Taula Marques

- **InsertMarca.** Rep com a paràmetres les dades d'una nova marca i l'afegeix a la taula.
- **UpdateMarca.** Rep com a paràmetres les dades d'una marca i les modifica a la taula.
- **DeleteMarca.** Rep com a paràmetre el ID d'una marca i l'esborra de la taula.
- **SelectMarcaPerID.** Rep com a paràmetre el ID d'una marca i retorna totes les dades de la marca.
- **SelectTotesMarques.** Retorna la taula marques complerta.

Taula Categories

- **InsertCategoria.** Rep com a paràmetres les dades d'una nova categoria i l'afegeix a la taula.
- **UpdateCategoria.** Rep com a paràmetres les dades d'una categoria i les modifica a la taula.
- **DeleteCategoria.** Rep com a paràmetre el ID d'una categoria i l'esborra de la taula.
- **SelectCategoriaPerID.** Rep com a paràmetre el ID d'una categoria i retorna totes les dades de la categoria.
- **SelectTotesCategories.** Retorna la taula categories complerta.

Taula Proveïdors

- **InsertProveïdor.** Rep com a paràmetres les dades d'un nou proveïdor i l'afegeix a la taula.
- **UpdateProveïdor.** Rep com a paràmetres les dades d'un proveïdor i les modifica a la taula.
- **DeleteProveïdor.** Rep com a paràmetre el ID d'un proveïdor i l'esborra de la taula.
- **SelectProveïdorPerID.** Rep com a paràmetre el ID d'un proveïdor i retorna totes les dades del proveïdor.
- **SelectTotsProveïdors.** Retorna la taula proveïdors complerta.

Taula Productes

- **InsertProducte.** Rep com a paràmetres les dades d'un nou producte i l'afegeix a la taula.
- **UpdateProducte.** Rep com a paràmetres les dades d'un producte i les modifica a la taula.
- **DeleteProducte.** Rep com a paràmetre el ID d'un producte i l'esborra de la taula.
- **SelectProductePerID.** Rep com a paràmetre el ID d'un producte i retorna totes les dades del producte.
- **SelectTotsProductes.** Retorna la taula productes complerta.

Implementació

Eines

Visual Studio 2005

L'eina principal pel desenvolupament de l'aplicació ha estat l'entorn de programació Microsoft Visual Studio 2005. Aquest entorn integrat serveix tant pel disseny com per a la programació, permet utilitzar diferents llenguatges de programació que a més a més poden conviure en la mateixa aplicació. Pel desenvolupament d'aquest projecte s'ha utilitzat el llenguatge de programació C#, per la seva potencia i per la seva similitud al llenguatge estudiat durant la carrera, el Java.

A més s'ha utilitzat el framework UIP Application Block, suportat per Microsoft i que és un framework del tipus MVC facilitant el procés de creació de l'aplicació utilitzant una programació en capes, aïllant totalment la capa de presentació de les capes de negoci i d'accés a dades. Aquest framework aprofita el fitxer de configuracions de l'aplicació en format XML, per a definir les navegacions i les transicions entre vistes.

Gracies a les facilitats que dona l'entorn de programació de Visual Studio 2005, només cal afegir les classes subministrades per l'UIP dintre de la solució com un projecte més, el codi font de UIP està desenvolupat en C#, però podia estar en qualsevol del llenguatges suportats per Visual Studio.

En la figura 23 es pot observar l'explorador de solucions de Visual Studio 2005 on es poden veure els diferents projectes que són:

- **Microsoft.ApplicationBlock.UIProcess**, subministrada pel framework UIP, que conté les classes necessàries per la navegació de les vistes i l'enllaç amb el controlador de cadascuna d'elles.
- **Microsoft.ApplicacionBlock.Data**, subministrada pel framework UIP, conté les classes per facilitar l'accés a les dades, aquest accés només es possible amb SQL Server de Microsoft.
- **CapaAccesDades**. Capa creada per accedir a les dades de cadascuna de les taules de la base de dades, en aquesta capa hi ha totes les crides necessàries per poder manipular les dades de les entitats de negoci.
- **EntitatsNegoci**. En aquesta capa es defineixen les entitats de negoci amb els seus atributs, s'han definit en format visual XSD Schema XML, on automàticament el propi Visual Studio 2005 genera el codi necessari per poder enllaçar amb les dades creant els DataSet necessaris.
- **ObjectesNegoci**. Aquesta capa enllaça les entitats de negoci amb la capa de dades.
- **Comunes**. En aquest projecte s'hi troba inclòs el controlador de l'aplicació així com classes auxiliars com la classe Imatge, que proporciona funcionalitats per a convertir imatges a array de bytes i viceversa.
- **WinUI**. Es la capa de presentació, aquí hi ha definides tots els formularis i components necessaris per a crear la interfície amb l'usuari.
- **TFCNet**. És el projecte inicial, conté l'executable de l'aplicació i el fitxer de configuracions en format XML on hi ha reflectides totes les característiques de les vistes i les seves navegacions.

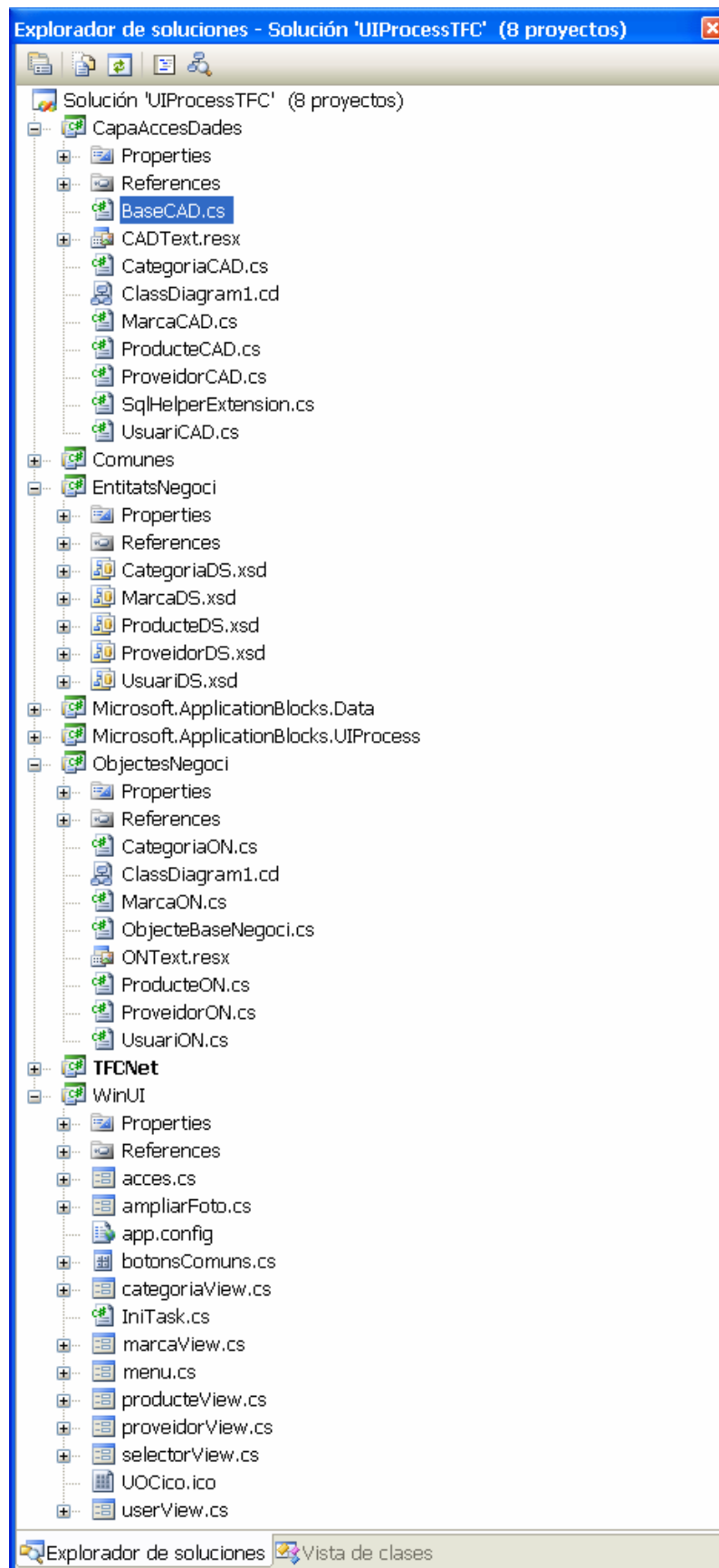


Figura 23. Explorador de soluciones de Visual Studio 2005.

Microsoft SQL Server 2005

Per a la capa de persistència s'ha utilitzat Microsoft SQL Server 2005 Express Edition, si bé la versió 2 del UIP estava pensada per utilitzar la versió 2000 del SQL Server, gracies a la seva compatibilitat no hi ha hagut cap problema a l'hora de treballar amb aquest nou motor de base de dades de Microsoft.

Microsoft SQL Server Management Studio Express

Com a complement al motor del SQL, s'ha utilitzat la utilitat Microsoft SQL Server Management Studio Express, que proporciona un potent entorn per a poder gestionar el SQL Server de Microsoft.

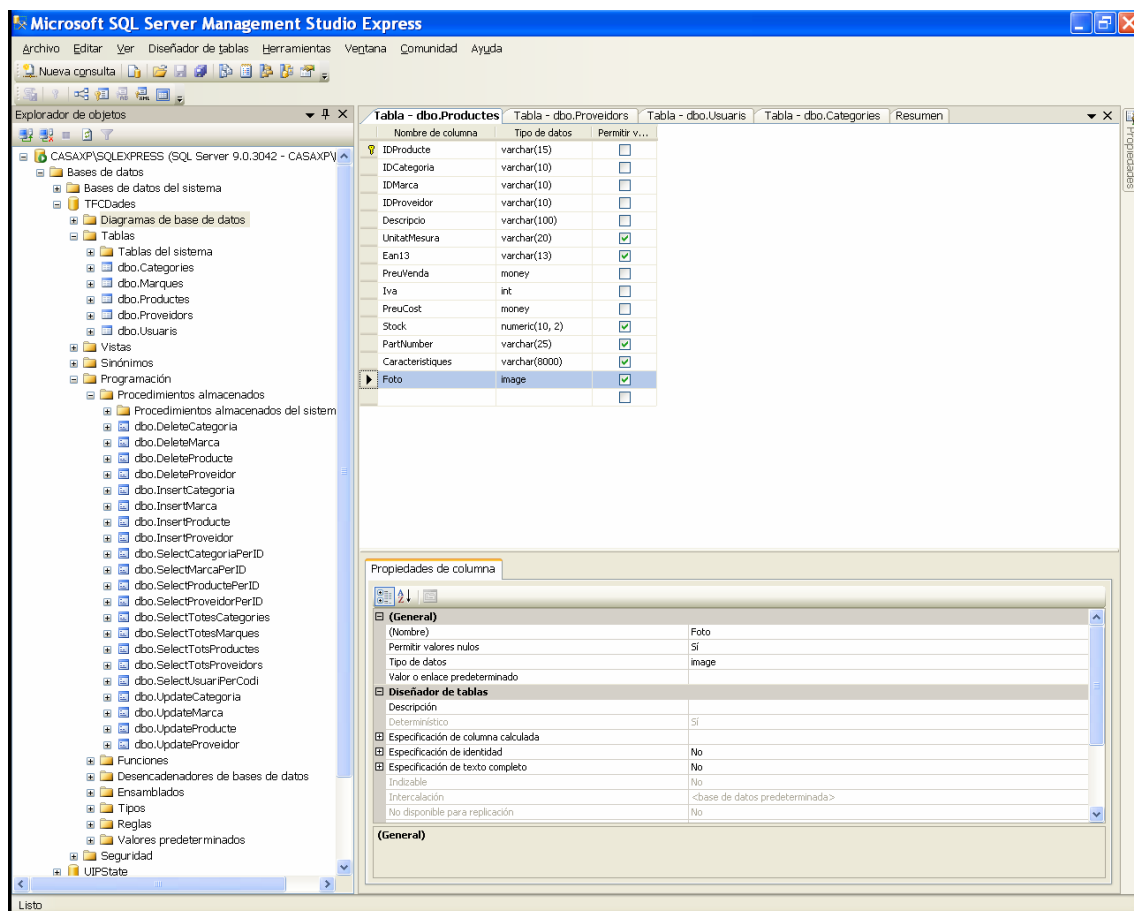


Figura 24. Microsoft SQL Server Mangement Studio Express.

Com es pot observar en la figura anterior, des de aquesta utilitat s'han pogut crear la base de dades de l'aplicació TFCdades, les seves taules i els procediments emmagatzemats necessaris pel funcionament correcte de l'aplicació.

A més aquesta utilitat també permet la manipulació de les dades, permetent inserir, modificar o esborrar files a les taules, així com poder comprovar el correcte funcionament del procediments emmagatzemats mostrant el resultats de les consultes.

Estructura

El model

El model està representat per dos espais de nom separats, *ObjectesNegoci* i *EntitatsNegoci*, en aquests projectes s'han definit les classes necessàries per gestionar les entitats de negoci i el seu enllaç amb la capa d'accés a dades.

En cadascuna de les entitats hi ha definits els mètodes necessaris per accedir a les dades i les relacions entre els diferents objectes de negoci. També en aquestes classes s'incorporen els controls necessaris per poder capturar les possibles excepcions que pugui produir l'accés a les dades, de manera que es pugui controlar qualsevol possible anomalia en el funcionament de l'aplicació.

El controlador

En aquesta aplicació només s'ha definit un controlador, ja que hi ha definides poques funcionalitats no s'ha cregut necessari utilitzar-ne més. Tal i com es pot observar en la figura 12, en el detall del fitxer de configuracions de l'aplicació, el controlador té el nom de *TFCControlador*, està inclòs dintre del projecte Comunes de la solució del TFC.

En aquest controlador es defineixen tots els procediments a seguir des de les diferents vistes de l'aplicació, permetent així que les vistes puguin rebre les dades necessàries per mostrar-les al usuari o enviar les dades introduïdes per l'usuari i emmagatzemar-les en la base de dades. El controlador es qui té la informació necessària per saber quina vista i com s'ha d'executar, a més hereta de *ControllerBase*, classe base subministrada pel framework UIP que permet que per a cada nova navegació s'estableixi un entorn aïllat permetent el pas de paràmetres entre les diferents vistes.

La vista, disseny de la capa de presentació

Definida en l'espai de noms *WinUI*, està integrada per un grup de Windows Forms i controls d'usuari, ja que alguns components es repeteixen en algunes de les vistes.

Totes aquestes vistes hereten de *WindowsFormView*, classe proporcionada per el framework UIP per tal de poder gestionar les vistes i els seus components, cada formulari només conté els elements que han de mostrar o adquirir les dades i els processos de validació d'aquestes dades. El control dels esdeveniments el marca el controlador de manera que les vistes interactuen amb el controlador sol·licitant d'aquestes les accions que cal prendre davant dels esdeveniments.

En aquest punt, cal comentar que es un control d'usuari. El que en l'entorn de .NET s'anomena un control d'usuari, no és més que un formulari convertit en un control i que per tant forma part d'un altre formulari i que es pot utilitzar per tants formularis com es vulgui, la qual cosa facilita la programació evitant tenir que repetir el contingut o la lògica de l'aplicació.

Disseny dels formularis

Accés

El primer formulari que apareix en l'aplicació i des de la que s'inicia tot el procés es el formulari d'accés, en aquest formulari es demana l'usuari i la clau de pas d'aquest per tal de poder esbrinar si l'usuari està autoritzat o no a l'accés a l'aplicació i quin tipus d'usuari tenim.

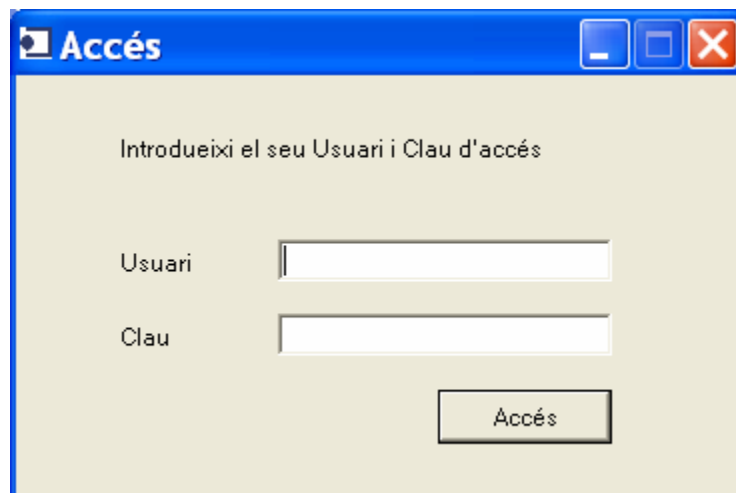


Figura 25. Formulari d'accés a l'aplicació.

Per tal que aparegui aquest formulari, des de el *Main* de l'aplicació es fa un crida a aquesta classe, a més també s'intercepten totes les possibles excepcions no controlades que puguin sorgir en el transcurs de l'aplicació de manera que aparegui un missatge d'error indicant-nos quin esdeveniment a provocat la sortida de l'aplicació.

```
public sealed class TFCNetStart
{
    [STAThread]
    public static void Main(string[] args)
    {
        Application.ThreadException += new
        System.Threading.ThreadExceptionHandler(Application_ThreadException);
        Application.Run( new acces() );
    }

    public static void Application_ThreadException(object source,
    System.Threading.ThreadExceptionEventArgs e)
    {
        string errorMessage = "";
        for( Exception tempException = e.Exception; tempException != null ; tempException =
        tempException.InnerException )
        {
            errorMessage += tempException.Message + Environment.NewLine + Environment.NewLine;
        }
        MessageBox.Show( string.Format( "Hi ha problemes al intentar emprar UIP Application
        block, per favor examini els següents missatges d'error: {0}"
        + Environment.NewLine + "Ha d'estar segur que UIP database scripts
        s'executen emprant sql server", errorMessage ),
        "Error d'Aplicació", MessageBoxButtons.OK, MessageBoxIcon.Error );
    }
}
```

Figura 26. Codi de la classe inicial de l'aplicació.

Un cop s'ha introduït l'usuari i la clau de pas, cal autenticar aquest usuari, aquest procés es porta a terme en la mateixa classe *accés* cridant als servis que brinda el controlador. Cal remarcar que la classe *accés* és l'única classe que no hereta de *WindowsFormView* de l'UIP ja que és el punt de partida, però sí que implementa la interface *IShutdownUIP* que permet la finalització de l'aplicació, ja que també serà el lloc per on sortir de l'aplicació.

Un cop autenticat l'usuari es poden donar dos camins, o bé és un usuari gestor o bé es un client, en aquest punt s'iniciaran dos navegacions diferents definides en el fitxer de configuracions del programa (figura 13), el la següent figura es pot observar el tros de codi que inicia aquestes navegacions, cridant als serveis oferts pel framework UIP.

```
private void okButton_Click(object sender, System.EventArgs e)
{
    //Demana al controlador si l'usuari és valid.
    if (TfcControlador.UsuariValid(usuariText.Text, clauText.Text))
    {
        // Usuari correcte.
        _startingTask = new IniTask();

        if (TfcControlador.EsUsuariAdministrador)
            UIPManager.StartOpenNavigationTask("administrador", "menu");
        else
            UIPManager.StartOpenNavigationTask("usuari", "userView");

        this.Visible = false;
    }
    else
    {
        MessageBox.Show("Usuari o Clau d'accés incorrectes", "Accés denegat",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

Figura 27. Codi que inicia les dues possibles navegacions.

Navegació del gestor

Un cop l'usuari s'ha autenticat i s'ha esbrinat que es tracta d'un gestor, s'inicia la navegació per un seguit de formularis per tal de cobrir els casos d'ús relacionat amb les tasques del gestor, que es poden portar a terme les tasques de manteniment de totes les taules implicades en la gestió del catàleg de productes.

Formulari Menu

A través d'aquest formulari el gestor pot triar quina acció vol prendre, quina taula vol actualitzar. Hi ha quatre possibles accions que es la gestió de cadascuna de les taules afectades en el manteniment del catàleg de productes.

En la següent figura es pot observar el formulari on hi apareixen quatre botons per iniciar cadascuna de les gestions i també figura el nom de l'usuari gestor.

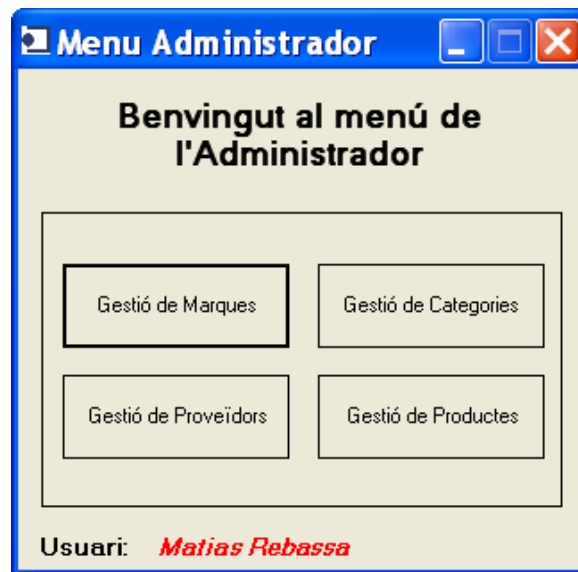


Figura 28. Formulari Menú d'administració.

En clicar un botó el formulari ho notifica al controlador i aquest pren les corresponents accions, tal i com es pot observar en el següent codi.

```
public partial class menu : WindowsFormView
{
    public menu()
    {
        InitializeComponent();
    }

    private void btnMarques_Click(object sender, EventArgs e)
    {
        TfcControlador.GestioMarques();
    }

    private void btnCategories_Click(object sender, EventArgs e)
    {
        TfcControlador.GestioCategories();
    }

    private void btnProveïdors_Click(object sender, EventArgs e)
    {
        TfcControlador.GestioProveïdors();
    }

    private void btnProductes_Click(object sender, EventArgs e)
    {
        TfcControlador.GestioProductes();
    }

    private void menu_Load(object sender, EventArgs e)
    {
        labelNomUsuari.Text = TfcControlador.NomUsuari;
    }

    private TfcControlador TfcControlador
    {
        get { return (TfcControlador)this.Controller; }
    }
}
```

Figura 29. Codi del formulari Menu.

Des de el controlador per a cada acció el codi es molt semblant i posa en marxa la vista corresponent, en el següent fragment de codi del controlador es veu la simplicitat de la crida als servis del framework UIP.

```
#region Funcions del menu de l'Administrador

///<summary>
///Activa la navegació per a la gestió de les marques
///</summary>
public void GestioMarques()
{
    UIPManager.StartUserControlsTask("GestioMarques");
}

///<summary>
///Activa la navegació per a la gestió de les categories
///</summary>
public void GestioCategories()
{
    UIPManager.StartUserControlsTask("GestioCategories");
}

///<summary>
///Activa la navegació per a la gestió dels proveïdors
///</summary>
public void GestioProveïdors()
{
    UIPManager.StartUserControlsTask("GestioProveïdors");
}

///<summary>
///Activa la navegació per a la gestió dels productes
///</summary>
public void GestioProductes()
{
    UIPManager.StartUserControlsTask("GestioProductes");
}
#endregion
```

Figura 30. Codi del controlador que gestiona el menú de l'administrador.

Gestió de Marques

Des de aquest formulari es portarà a terme la gestió completa de la taula de Marques, en aquest formulari s'integren tant les altes com les modificacions com les possibles baixes de marques.

Per tal de poder realitzar aquest funcions s'ha de controlar les dades introduïdes en el *textbox* del Identificador de la marca, cas d'existir el formulari mostrarà les dades al usuari i activarà el boto de *Esborrar* permetent així eliminar una marca de la taula. Cas de modificar alguna de les dades del formulari s'activarà el botó de Acceptar cobrint així la funcionalitat de modificació. Cas que l'identificador introduït no existeixi serà considerada una alta d'una nova marca.

Per tal de facilitar les cerques s'ha implementat un botó com el ressaltat en la figura següent que permet realitzar les cerques d'una manera ràpida i còmode i que es comú en tots el formularis de la gestió del catàleg.

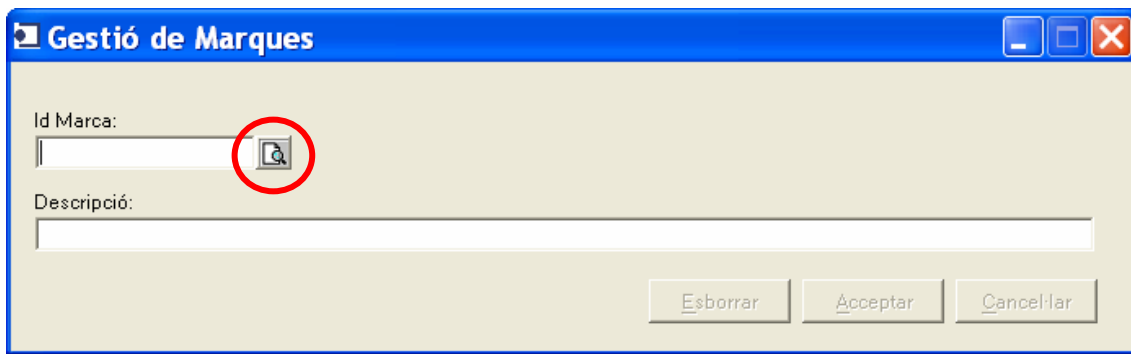


Figura 31. Formulari Gestió de Marques.

Com es pot observar en la figura anterior, aquest formulari té una botonera associada per tal de poder donar servei a les diferents accions que es puguin dur a terme, aquesta botonera és comú en tots els formularis de la gestió del gestor i s'ha creat com a control d'usuari.

El formulari de selecció per tal de dur a terme les cerques, vinculat al botó ressaltat té un tractament especial que es detallarà posteriorment.

Cada cop que es finalitza una acció amb una marca el formulari torna a sol·licitar al usuari un nou identificador de marca per tal de tornar a començar un nou cicle, si es surt del formulari l'usuari retorna al menú de l'administrador.

Gestió de Categories

El tractament, aparença i funcions d'aquesta gestió són gairebé idèntics al de la gestió de marques, exceptuant que actua sobre la taula de Categories.

Tot i que semblen gairebé idèntiques i es podia haver plantejat un sol formulari per a les dues gestions sobrecarregant alguns mètodes, es va optar per crear dos *Winforms* diferents per no reduir en accés les vistes a tractar.

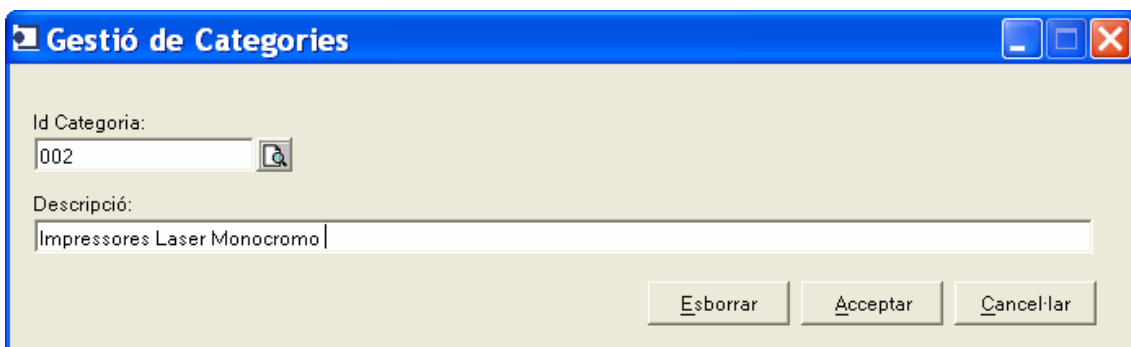
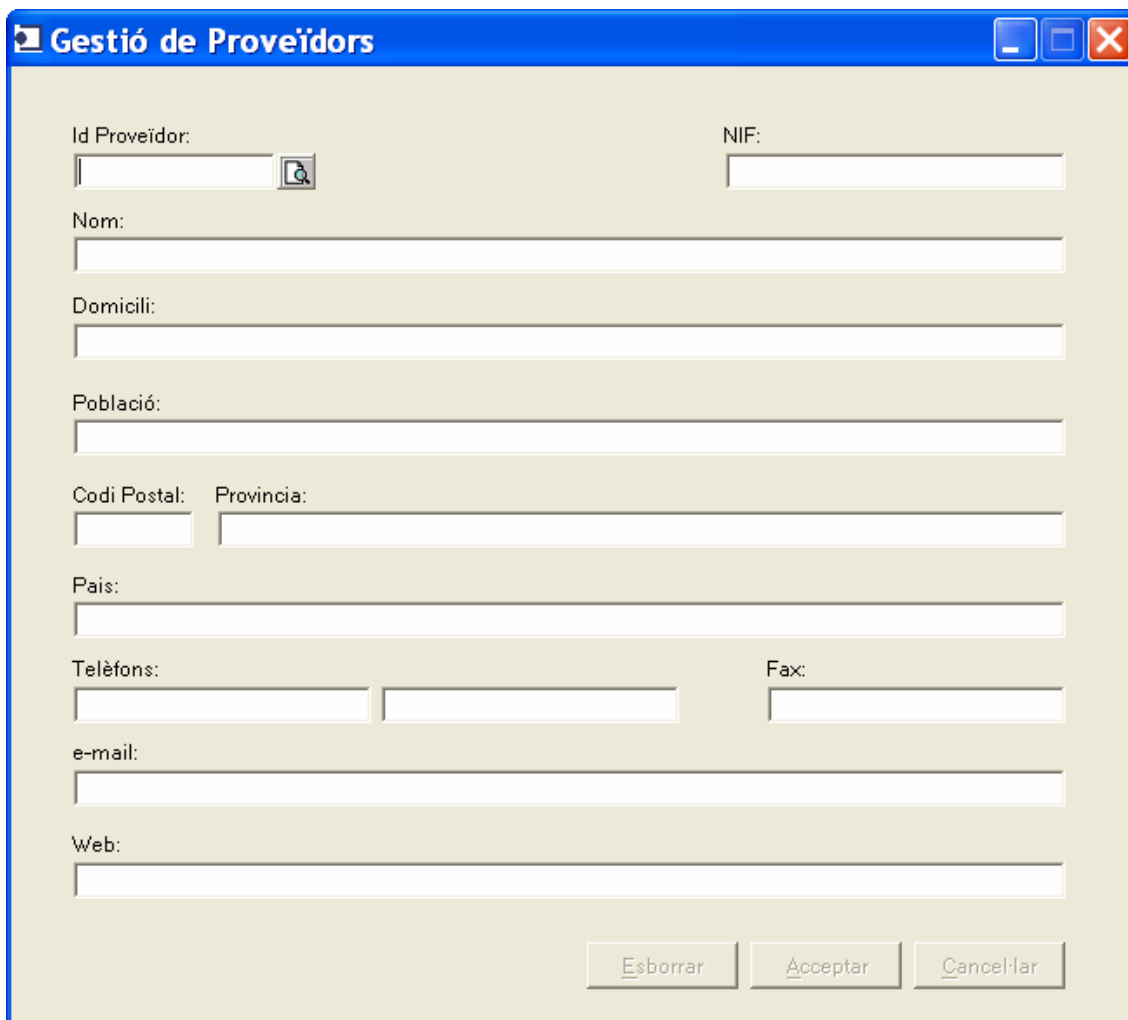


Figura 32. Formulari Gestió de Categories.

Gestió de Proveïdors



The screenshot shows a Windows-style application window titled "Gestió de Proveïdors". The window contains a form with the following fields and controls:

- Id Proveïdor:** A text input field with a search icon to its right.
- NIF:** A text input field.
- Nom:** A text input field.
- Domicili:** A text input field.
- Població:** A text input field.
- Codi Postal:** A text input field.
- Provincia:** A text input field.
- Pais:** A text input field.
- Telèfons:** Two text input fields.
- Fax:** A text input field.
- e-mail:** A text input field.
- Web:** A text input field.

At the bottom of the form, there are three buttons: "Esborrar", "Acceptar", and "Cancel·lar".

Figura 33. Formulari Gestió de Proveïdors.

Aquesta vista ofereix les mateixes funcionalitats que les dues anteriors, però a més, al tenir més camps, tal i com es pot observar en la figura anterior, alguns d'ells de caràcter obligatori en la definició de la base de dades, s'han afegit processos de verificació de les dades, de manera que en el moment de acceptar una alta o una modificació d'un proveïdor en indica si el formulari es correcte o no per tal de poder ser guardat a la base de dades i evitar així errors de validació per tal del motor del SQL provocant excepcions en l'aplicació.

En el procés de validació s'ha utilitzat un component de .NET *ErrorProvider* de manera que en el moment que es comprova si els camps contenen la informació correcte es pot activar aquest component mostrant en el formulari un símbol d'avís i indicant la causa del problema.

En la figura següent es mostra un exemple d'aquest avisos d'error en lloc de notificar-ho amb un *MessageBox*, quedant més elegant i amb una menor pèrdua de la navegació per part del usuari.

The screenshot shows a Windows-style window titled "Gestió de Proveïdors". The form contains the following fields and values:

- Id Proveïdor:** 001 (with a search icon)
- NIF:** (empty, with a red warning triangle icon)
- Nom:** Proveïdor 001
- Domicili:** Domicili 001, 12525
- Població:** Població 001
- Codi Postal:** 07008
- Provincia:** Illes Balears
- Pais:** Espanya
- Telèfons:** +34 971 444 444, +34 971 555 555
- Fax:** +34 971 666 666
- e-mail:** email@proveidor.com
- Web:** www.proveidor.com

At the bottom are three buttons: "Esborrar", "Acceptar", and "Cancel·lar". A tooltip box on the right side of the NIF field contains the text: "Es obligatori indicar e NIF del Proveïdor".

Figura 34. Formulari Gestió de Proveïdors amb notificació d'errors.

Gestió de Productes

En aquest formulari al igual que en els anterior hi ha la botonera comú, també he incorporat el sistema de validació dels camps obligatoris i del camps numèrics per tal de verificar que contenen la informació correcte.

A més, en aquest formulari també s'inclouen enllaços amb dades de les altres taules, són els camps definits com a claus foranes en la base de dades, aquest camps incorporen també l'opció de cerques sobre aquestes taules per tal de facilitar la feina al usuari.

També incorpora un component *PintureBox* per tal de poder mostrar o inserir una imatge corresponent al producte, aquesta imatge pot ser carregada des de un arxiu o bé pot ser copiada des del "Clipboard" de Windows, facilitant així la incorporació d'imatges que poden ser copiades des de les planes Web dels fabricants del producte sense tenir que emmagatzemar-les temporalment a l'ordinador.

També s'ha dotat al component de la imatge de la possibilitat de ampliar la foto al clicar amb el ratolí a sobre, permetent així veure amb més detall la imatge.

Gestió de Productes

Id Producte: 002 Categoria: 004 Plaques Mare

Marca: INTEL Intel Corporation

Descripció: Intel® Desktop Board D975XBX2

Proveïdor Habitual: 002 Nom proveïdor 002

Unitat Messura: Unitat Codi EAN 13: 1111111111111 Part Number: 4464654-4546

Preu de Cost: 180,2500 Preu de Venda: 215,5000 IVA: 16 Stock: 2

Característiques:

- Form Factor ATX (12.00 inches by 9.60 inches [304.80 millimeters by 243.84 millimeters])
- Processor
- Support for Intel® Core™2 Quad processor in an LGA775 socket with a 1066 MHz system bus
- Support for Intel® Core™2 Extreme processor in an LGA775 socket with a 1066 MHz system bus
- Support for an Intel® Core™2 Duo processor in an LGA775 socket with a 1066 MHz system bus
- Support for an Intel® Pentium® Processor Extreme Edition in an LGA775 socket with a 1066 or 800 MHz system bus
- Support for an Intel® Pentium® 4 Processor Extreme Edition supporting Hyper-Threading Technology† (in an LGA775 socket with a 1066 MHz system bus)
- Support for an Intel® Pentium® D processor in an LGA775 socket with an 800 MHz system bus
- Support for an Intel Intel® Pentium® 4 processor supporting Hyper-Threading Technology† (in an LGA775 socket with an 800 MHz

Foto:

Carregar Enganxar Esborrar

Esborrar Acceptar Cancel·lar

Figura 35. Formulari Gestió de Productes.

En la figura anterior es pot observar com apareixen les opcions de cerca a cadascun del identificadors de Producte, Categoria, Marca i Proveïdor, que camps claus de les seves corresponents taules. Aquestes cerques es realitzen mitjançant una única vista comuna que s'explicarà més extensament a continuació.

Cada component *TextBox* que fa referència a un camp enllaçat amb una taula auxiliar, provoca que a la sortida del component es verifiqui aquest valor i només s'accepten aquells valors que pertanyen a objectes vàlids, es a dir, a elements existents dins de la seva corresponent taula.

També es pot observar el component de la foto amb els botons associats per tal de poder carregar la foto des de un arxiu. Aquesta funcionalitat es fa aprofitant una de les moltes facilitats que incorpora la plataforma .NET amb quadres de diàleg ja assemblats i només cal fer una crida per disposar del servei de carregar un arxiu des de qualsevol ubicació del nostre ordinador.

En la figura següent es pot veure el codi que permet carregar una imatge des de un arxiu, sol·licitat els serveis que ens ofereix la classe *OpenFileDialog*.

```
private void btnCarregar_Click(object sender, EventArgs e)
{
    // Seleccionar una imatge
    OpenFileDialog oFD = new OpenFileDialog();
    oFD.Title = "Selecccionar la imatge";
    oFD.Filter = "Tots (*.*)|*.*|Imatges|.jpg|.gif|.png|.bmp";
    if (oFD.ShowDialog() == DialogResult.OK)
    {
        this.foto.Image = Image.FromFile(oFD.FileName);
        activarModificacions(sender, e);
    }
    this.btnLlevar.Enabled = this.foto.Image != null;
}
```

Figura 36. Codi per carregar un imatge des de arxiu

En la figura següent es mostra l'ampliació que es produeix en clicar amb el ratolí sobre la imatge del producte.



Figura 37. Ampliació de la imatge d'un producte.

S'ha controlat que l'opció de esborrar la imatge del producte, amb el botó definit a tal efecte (figura 35, botó *Esborrar*), només estigui actiu en el cas de que el producte tingui alguna imatge carregada, com es pot comprovar en el fragment de codi mostrat en la figura 36, el botó *Esborrar* s'activa en funció del valor que conté el component *PictureBox*.

Selector

Aquest formulari és el que permet fer una cerca dintre de cadascuna de les taules que intervenen en la gestió del catàleg.

S'ha definit un sol formulari Windows per a donar servei a totes les peticions, i aprofitat la possibilitat que ofereix el framework UIP de passar paràmetres ha permès el modelar aquesta situació, simplificant molt la tasca de programació, estalviant esforços a l'hora de fer la implementació.

El selector, mitjançant els paràmetres que rep en el moment de ser instanciat, pot tenir activades unes funcionalitats o no, en la figura següent es mostra el selector de Marques cridat des de la gestió de marques.

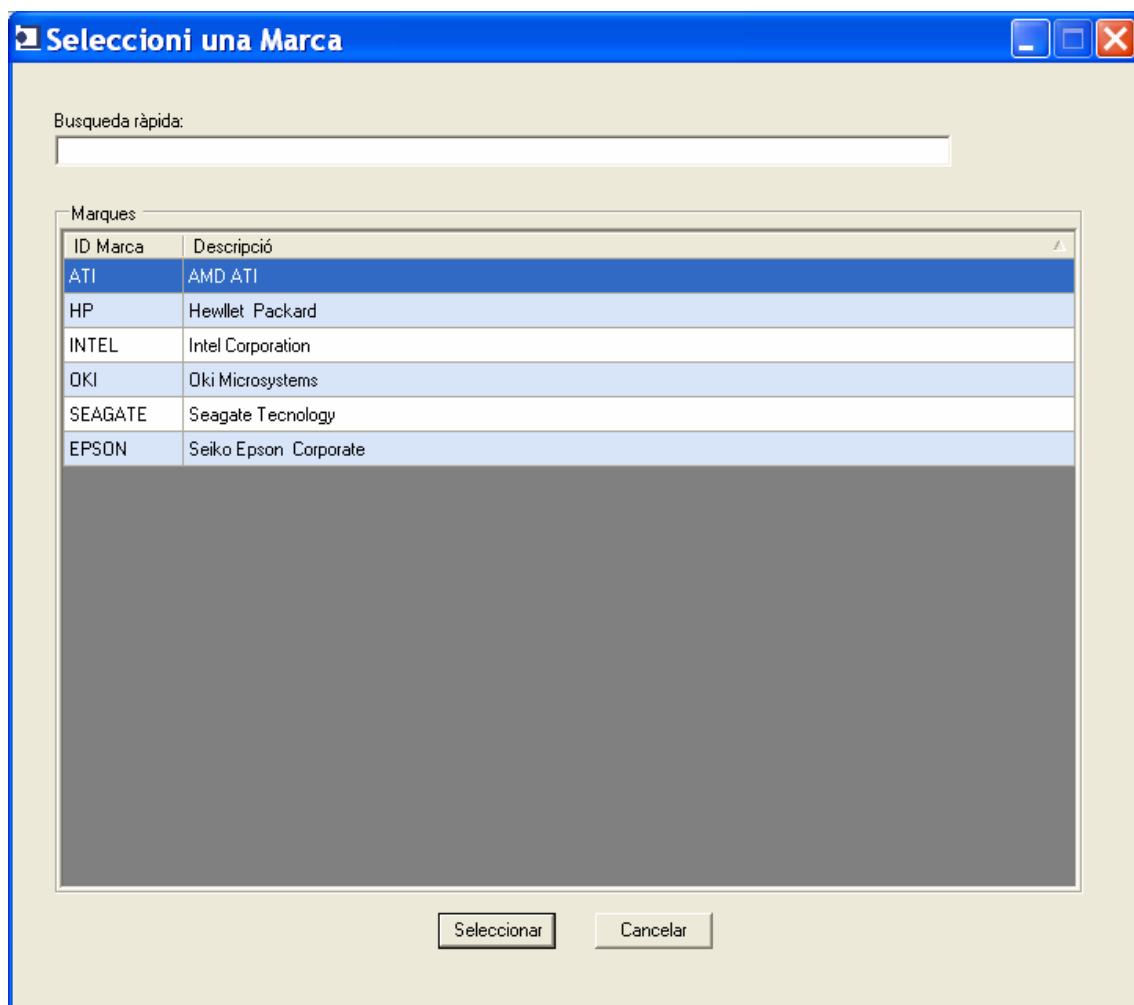


Figura 38. Formulari de selector des de la pròpia gestió.

Com es pot observar en aquest cas el selector ens permet triar una marca ensenyant una component *DataGridView* on hi ha carregades totes les dades de la taula corresponent, a més el selector incorpora un component *TextBox* per tal de poder realitzar un filtratge de les dades que apareixen en el *Grid*. Per tal de donar velocitat a les cerques aquest filtratge es realitza a cada modificació del contingut

del *TextBox* de filtratge, en la figura següent es mostra el codi que intercepta l'esdeveniment de la modificació del *TextBox*.

```
private void selectorText_TextChanged(object sender, EventArgs e)
{
    baseDadesBS.Filter = campsBD[0,0] + " Like '%" + selectorText.Text
+ '%" or " + campsBD[1,0] + " Like '%" + selectorText.Text + "%'";
}
```

Figura 39. Fragment de codi del filtratge del selector.

Com es pot observar en el codi, basta modificar la propietat *Filter* del component *BindingSource* amb una cadena SQL per tal d'obtenir un filtratge que afecta als dos camps mostrats al *Grid*, permetent així al usuari reduir els elements mostrats d'una manera àgil i còmode.

En la figura següent es mostra aquesta reducció que es produeix en el mateix moment que es van prement les tecles, en aquest cas, ens mostra els elements que contenen la cadena "se" tant en el identificador com en la descripció.

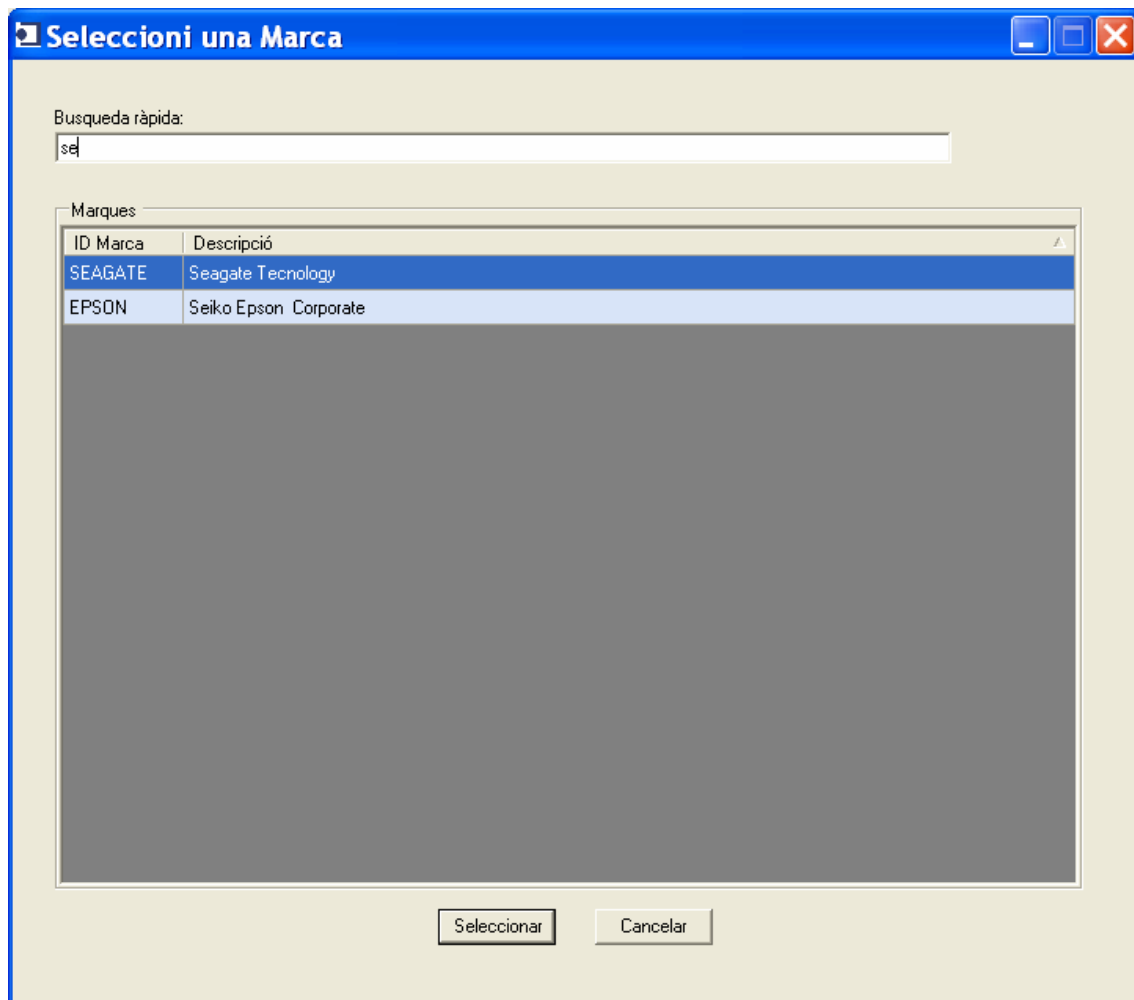


Figura 40. Formulari de selector amb filtratge.

A continuació en la figura següent vorem el mateix selector activat des de el formulari de la gestió dels productes.

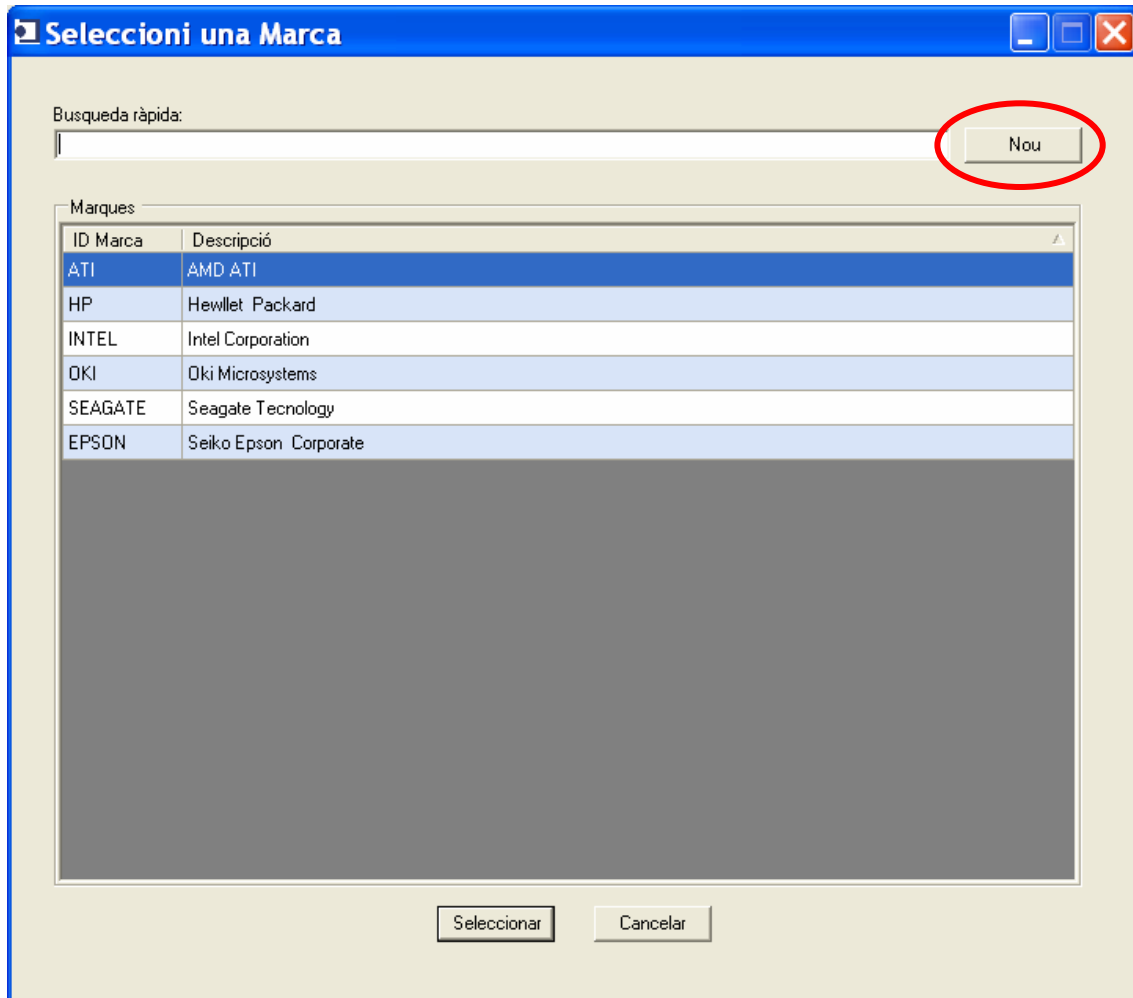


Figura 41. Formulari de selector de marques activat des de gestió de productes.

Com es pot observar apareix un botó que ens dona la possibilitat de donar d'alta un nou element en la taula de marques, sense tenir que sortir de la gestió de productes i anant a la opció de la gestió de marques, amb aquesta opció s'ha simplificat la tasca de afegir nous elements a les taules auxiliars permetent dur tota la gestió des de la pròpia gestió de productes.

A continuació explicarem com s'ha aconseguit implementar aquesta solució, ja que des de el formulari que llança la crida al controlador, s'ha de passar una sèrie de paràmetres per tal d'indicar si s'ha d'activar la possibilitat o no de donar altes, ja que si des de el mateix formulari de la gestió de Marques es dones aquesta possibilitat es podria produir un bucle cridant-se a el formulari a si mateix, efecte no desitjat en cap cas.

```
private void btnSelectorMarca_Click(object sender, EventArgs e)
{
    if (idProducteText.Text.Trim() != "")
    {
        TfcControlador.SeleccionarMarques(true);
        if ((string)TfcControlador.Stat["Sortida"] != "")
        {
            idMarcaText.Text = (string)TfcControlador.Stat["Sortida"];
            idMarcaText_Leave(sender, e);
            descripcioText.Focus();
        }
        else
        {
            idMarcaText.Focus();
        }
    }
    else
        System.Media.SystemSounds.Beep.Play();
}
```

Figura 42. Codi d'activació del selector de Marques amb altes.

Com es pot observar, es reclama el mètode *SeleccionarMarques* del controlador amb un paràmetre booleà activat a vertader per tal que es puguin donar d'altes des de el selector. A més s'observa que en tornar del selector es comprova si s'ha seleccionat algun element comprovant si el element *Stat* per a la clau "Sortida" te un valor nul o bé conté el valor del identificador seleccionat.

L'element *Stat* es un element gestionat per el framework UIP que incorpora la possibilitat de passar valor d'una vista a un altre, en aquest mateix component *Stat* guarda la informació corresponent al estat en el que es troba el procés, de quina vista ve i cap a quina vista va.

```
public void SeleccionarMarques(bool Altes)
{
    Stat["NomForm"] = "Seleccioni una Marca";
    Stat["NomTaula"] = "marca";
    Stat["NomTitol"] = "Marques";
    Stat["CampsBD"] = new string[,] { { "IDMarca", "ID Marca" }, {
"Descripcio", "Descripció" } };
    Stat["Altes"] = Altes;
    carregarDS += new CarregarDS(TotesMarques);
    altaGenerica += new Alta(GestioMarques);
    UIPManager.StartUserControlsTask("Selector", new IniTask(), new
TaskArgumentsHolder(Stat.TaskId, "Productes", Stat));
    carregarDS -= new CarregarDS(TotesMarques);
    altaGenerica -= new Alta(GestioMarques);
}
```

Figura 43. Codi d'activació del selector de Marques amb altes.

En el fragment de codi anterior, es mostra com es prepara la crida del selector per tal d'indicar amb quina taula es treballarà, els textos que han de apareixia en el formulari i sobrecarregat uns mètodes delegats que permetran que es carregin les dades de la taula corresponent fent servir un sol mètode des de el selector.

El detall del codi d'aquest mètodes sobrecarregats es mostra a continuació.

```
.../...

public delegate DataSet CarregarDS();
public static CarregarDS carregarDS;

public delegate void Alta();
public static Alta altaGenerica;

.../...

///<summary>
///Activa la navegació per a la gestió de les marques
///</summary>
public void GestioMarques()
{
    UIPManager.StartUserControlsTask("GestioMarques");
}

.../...

public DataSet TotesMarques()
{
    return new MarcaON().TotesMarques();
}

.../...
```

Figura 44. Codi dels mètodes delegats del controlador.

Com es pot observar, el mètode *GestioMarques* és el mateix que s'havia mostrat en la figura 29, que correspon al grup de mètodes per a activar la navegació des de el menú de l'administrador.

El mètode delegat *CarregarDS* es sobrecarregat en cada cas per tal de cridar al mètode de cada objecte de negoci per a que retorni totes les dades de la taula.

En el moment de instanciar el Selector, s'han de llegir els arguments passats des de el controlador, això s'aconsegueix sobrecarregant el mètode *Initializate* definit a la classe *WindowsFormView* del UIP.

En la figura 45 es pot observar aquest codi, on s'extreuen els arguments passat a la classe per tal de poder carregar les dades i mostrar els texts corresponents segons la taula amb la que s'està treballant, també cal remarcar que s'han aprofitat la potencia d'alguns components que ofereix .NET Framework 2.0 com son els *BindingSource* i el *DataGridView*.

Aquest punt va ser un dels que més problemes donaren a l'hora de fer la implementació de l'aplicació, ja que incorporava tècniques que eren totalment noves com els delegats, i a més, es va tenir que estudiar en profunditat la manera de treballar del framework UIP per tal de poder passar arguments entre vistes.


```
public partial class selectorView : WindowsFormView
{
    private BindingSource baseDadesBS;
    private DataSet baseDadesDS;
    private string[,] campsBD;
    private State stat;

    public selectorView()
    {
        InitializeComponent();
    }

    public override void Initialize(TaskArgumentsHolder args, ViewSettings
settings)
    {
        this.stat = (State)args.TaskArguments;
        this.Text = (string)this.stat["NomForm"];
        this.campsBD = (string[,])this.stat["CampsBD"];
        this.groupBox.Text = (string)this.stat["NomTitol"];
        btnAlta.Visible = (bool)this.stat["Altes"];
        this.stat["Sortida"] = "";

        dadesGrid.AutoGenerateColumns = false;

        CarregarBaseDades();

        dadesGrid.AllowUserToOrderColumns = false;
        dadesGrid.AllowUserToDeleteRows = false;
        dadesGrid.AllowUserToAddRows = false;
        // posar un color diferent alternatiu a les files de la rejilla
        dadesGrid.AlternatingRowsDefaultCellStyle.BackColor =
SystemColors.InactiveCaptionText;
        dadesGrid.MultiSelect = false;
        dadesGrid.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
        dadesGrid.VirtualMode = true;

        //
        // 1º Columna: ID
        //
        DataGridViewTextBoxColumn colId = new DataGridViewTextBoxColumn();
        colId.AutoSizeMode =
DataGridViewAutoSizeColumnMode.DisplayedCells;
        colId.DataPropertyName = campsBD[0,0];
        colId.HeaderText = campsBD[0, 1];
        colId.Name = campsBD[0,0];
        colId.ReadOnly = true;
        dadesGrid.Columns.Add(colId);

        //
        // 2º Columna: Descripció per a Productes, Categories i Marques.
Nom per a Proveïdors
        //
        DataGridViewTextBoxColumn colDescripccio = new
DataGridViewTextBoxColumn();
        colDescripccio.AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;
        colDescripccio.DataPropertyName = campsBD[1,0];
        colDescripccio.HeaderText = campsBD[1,1];
        colDescripccio.Name = campsBD[1,0];
        colDescripccio.ReadOnly = true;
        dadesGrid.Columns.Add(colDescripccio);
    }
}
```

Figura 45. Codi de inicialització de la classe selector.

Botonera

La botonera es va crear con un control d'usuari, els botons s'activen en funció del tractament que es facin de les dades, si es tracta d'una alta, inicialment no s'activa cap botó, en el moment que s'insereix alguna dada en el formulari s'activen els botons de *Acceptar* i *Cancel·lar*.

Si es tracta d'un element existent, inicialment s'activa el botó *Esborrar*, i si es modifica alguna dada del formulari s'activen els altres dos botons.

El botó *Acceptar* verifica les dades en els formularis de Proveïdors i de Productes i actua de manera diferent segons si es una alta o si es una modificació, cridant al mètode corresponent del controlador.

El botó *Cancel·lar* descarta qualsevol dada o canvi introduït en el formulari.

El botó *Esborrar* elimina de la taula l'element mostrat en el formulari, però abans de esborrar l'element mostra un *MessageBox* demanant la confirmació per part de l'usuari, tal i com es mostra a la figura següent.

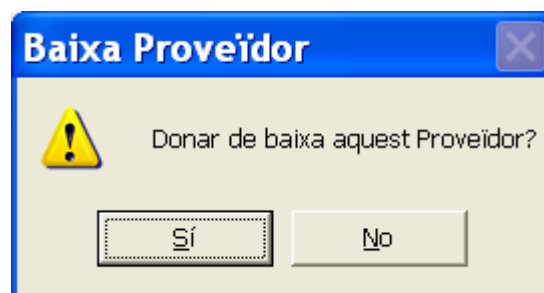


Figura 46. Missatge de confirmació de baixes.

Navegació del Client

La navegació del client només consta d'un formulari, la visualització del catàleg dels productes, amb les dades d'interès per el client, es a dir, que no es mostren totes les dades, ja que algunes d'aquestes dades son de caràcter privat i només hi han de tenir accés els gestors.

En aquest apartat, s'ha intentat optimitzar el sistema de cerques de la informació, permetent al usuari de classificar els productes segons la seva categoria i marca, o per només un camp, permetent filtratge sobre les dades mostrades.

S'ha dissenyat un formulari diferent a la majoria de sistemes de cerca de productes, on es solen incorporar sistemes de classificació mitjançant components del tipus *ComboBox* on es poden seleccionar el elements per el que es vol fer la classificació.

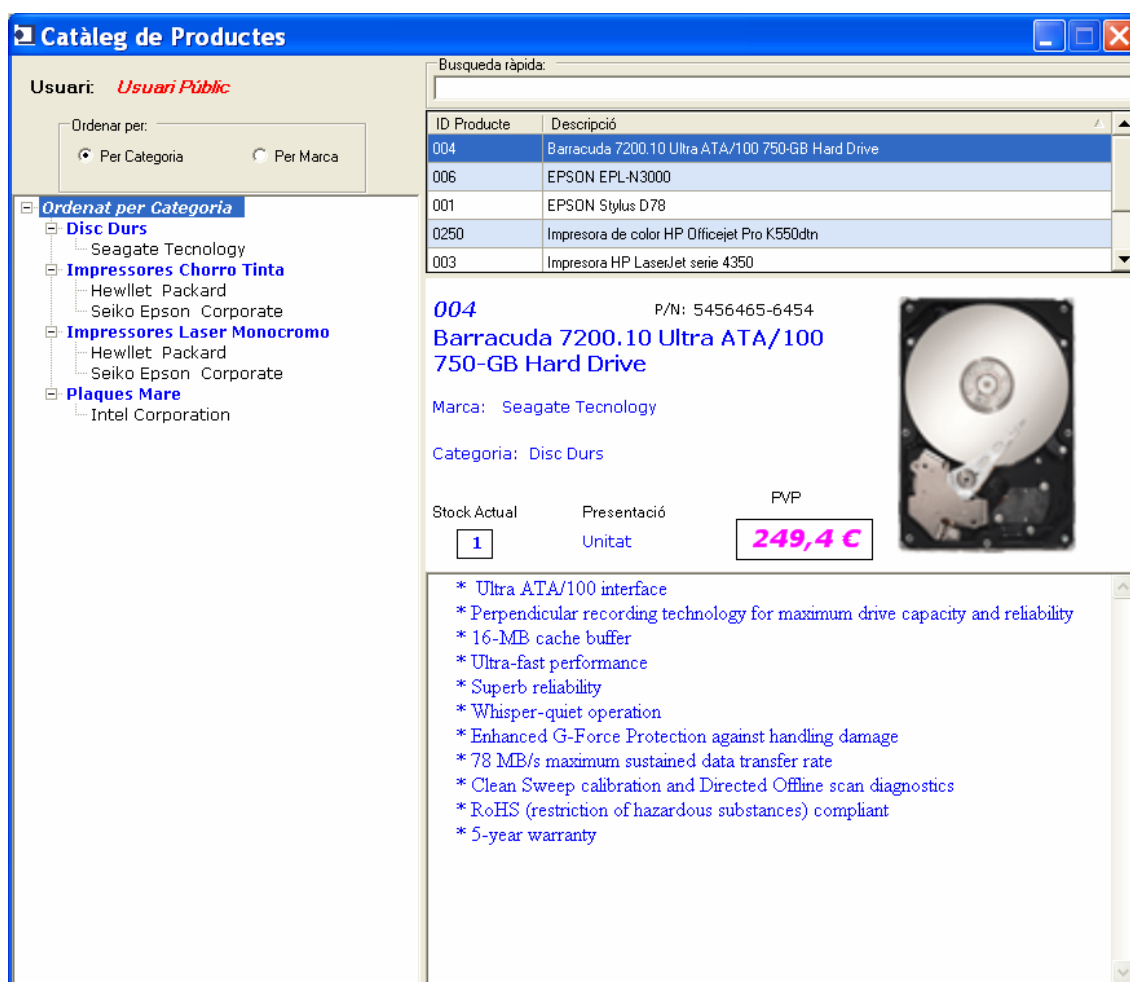


Figura 47. Formulari del Catàleg de productes.

Com es pot observar en la figura anterior el formulari està separant en tres blocs diferenciats:

Primer trobem a la part esquerra, un component *TreeView* amb associat amb uns component *RadioButton* que permeten triar al usuari l'ordre de com muntar l'arbre de selecció.



Figura 48. Arbre de selecció.

A la dreta trobem un component *DataGridView* que mostra els elements del node seleccionat en l'arbre, a més també permet filtratge simultani com l'explicat en el cas del selector.

Busqueda ràpida:

ID Producte	Descripció
004	Barracuda 7200.10 Ultra ATA/100 750-GB Hard Drive
006	EPSON EPL-N3000
001	EPSON Stylus D78
0250	Impresora de color HP Officejet Pro K550dtn
003	Impresora HP LaserJet serie 4350

Figura 49. Llista de productes segons la selecció.

I per acabar davall del *DataGridView* es troba un panell on es mostra tota la informació rellevant del producte, mostrant la foto i permetent fer ampliació de la mateixa.



Figura 50. Dades del producte.

En aquest panell es mostra la informació rellevant per un usuari client, mostrant l'stock actual del producte, el preu amb l'IVA inclòs i remarcant les característiques del producte.

El funcionament d'aquest formulari pretén que sigui molt intuïtiu per l'usuari, pot canviar l'ordre de selecció del arbre activant un o altre *RadioButton*, d'aquesta manera es munta l'arbre amb l'ordre Categoria→Marca, si s'activa el botó Categoria, on apareixeran com a branques les categories i d'elles i penjaran les marques, tal i com es veu en la figura 47.

Cas d'activar el botó de Marca, l'ordre de muntatge de l'arbre es inversa a l'anterior es a dir Marca→Categoria, a les branques hi apareixen les marques i a les fulles hi ha les categories que engloba cada marca, com es pot apreciar a la figura 51.

Cal destacar que en el muntatge de l'arbre només hi apareixen les categories i marques que tenen productes que les referencien, no és un producte cartesià de les dues taules de on sortirien gran quantitat de fulles sense cap producte.

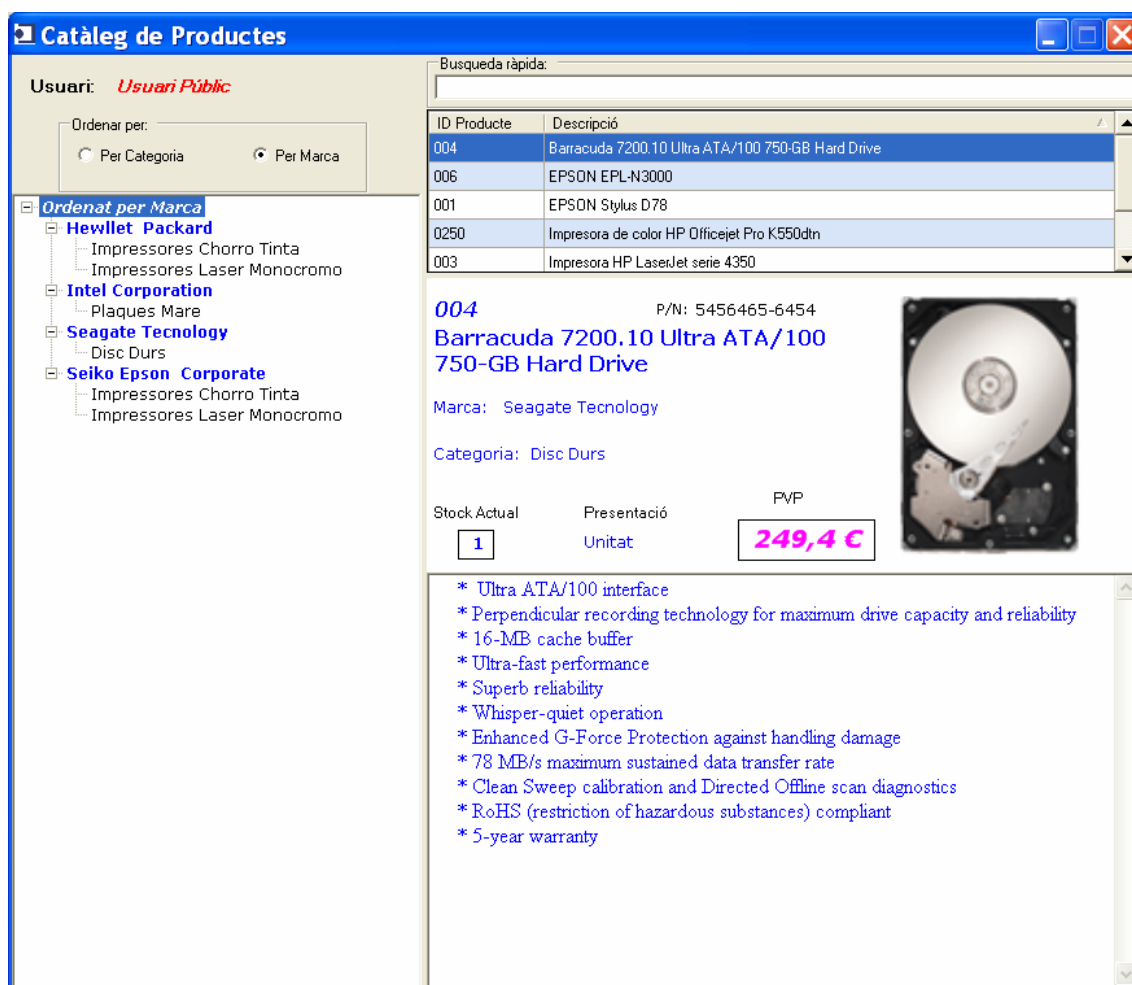


Figura 51. Catàleg de productes per Marca.

Quan l'usuari selecciona un dels nodes de l'arbre, el sistema visualitza els productes afectats per el node, en el cas del node arrel es mostra el total dels productes, com és el cas de les figures 47 i figura 51. Si el node es una branca es mostraran tots els articles que pertanyen a una categoria o a una marca segons l'opció que estigui activada com a ordenació principal, es pot observar en la figura 52, on apareixen tots el productes de la categoria "Impressores Chorro Tinta", on es pot observar que hi apareixen barrejades les de diferents marques.

Si en canvi, l'usuari selecciona una fulla, només apareixen els productes que compleixen les dues condicions, que son de la categoria i de la marca a la que fa referència la fulla (Figura 53).

A més, pensant que un catàleg de productes pot tenir una gran quantitat d'elements d'una determinada categoria i marca, els productes apareixen ordenats descendentment per la descripció del producte i es pot fer un filtratge per text, que al igual que en el selector, filtre tots aquells productes que contenen la cadena de text tant en el identificador com en la descripció.

The screenshot shows a web application window titled "Catàleg de Productes". At the top left, it indicates the user is "Usuari: *Usuari Públic*". Below this, there are sorting options: "Ordenar per:" with radio buttons for "Per Categoria" (selected) and "Per Marca".

A tree view on the left side shows the product hierarchy under "Ordenat per Categoria":

- Disc Durs
 - Seagate Technology
- Impressores Chorro Tinta
 - Hewlett Packard
 - Seiko Epson Corporate
- Impressores Laser Monocromo
 - Hewlett Packard
 - Seiko Epson Corporate
- Plaques Mare
 - Intel Corporation

The main content area features a search bar "Busqueda rápida:" and a table of products:

ID Producte	Descripció
001	EPSON Stylus D78
0250	Impresora de color HP Officejet Pro K550dtn

The selected product, "001 EPSON Stylus D78", is shown in detail. It includes the P/N: 111-562-558, a small image of the printer, and the following information:

- Marca: Seiko Epson Corporate
- Categoria: Impressores Chorro Tinta
- Stock Actual: 10
- Presentació: Unitat
- PVP: 58,9976 €

Technical specifications are listed below:

- Método de impresión: Cabezal avanzado Epson MicroPiezo™
- Configuración de los inyectores: 90 inyectores para negro/29 x 3 inyectores para color
- Tamaño de gota: 4 picolitros
- Resolución: 5760 x 1440 ppp* optimizados
- Velocidad de impresión:
 - Texto en negro en formato A4: 22 ppm
 - Texto en color en formato A4: 12 ppm
 - Fotografía 10 x 15: Aprox. 91 segundos
- Interfaz: USB 1.1
- Manejo del papel: Capacidad 80 hojas A4

Figura 52. Catàleg de productes d'una Categoria.

Catàleg de Productes

Usuari: *Usuari Públic*

Ordenar per:
 Per Categoria Per Marca

Ordenat per Categoria

- Disc Durs
 - Seagate Technology
- Impressores Chorro Tinta
 - Hewlett Packard
 - Seiko Epson Corporate
- Impressores Laser Monocromo
 - Hewlett Packard
 - Seiko Epson Corporate
- Plaques Mare
 - Intel Corporation

Busqueda ràpida:

ID Producte	Descripció
0250	Impresora de color HP Officejet Pro K550dtn

0250 P/N: C8158A
Impresora de color HP Officejet Pro K550dtn

Marca: Hewlett Packard

Categoria: Impressores Chorro Tinta

Stock Actual: Presentació: Unitat PVP: **249,4 €**

Velocidad de impresión (negro, calidad óptima, A4)
5 ppm

Velocidad de impresión (color, calidad óptima, A4)
5 ppm

Calidad de impresión (negro, calidad óptima)
Hasta 1.200 x 1.200 ppp

Calidad de impresión (color, calidad óptima)
Con una resolución optimizada de hasta 4.800 x 1.200 ppp con 1.200 de entrada

Memoria de serie
32 MB

Duty cycle (monthly, A4)
Hasta 7.500 páginas por mes

Opciones de impresión a doble cara
automática

Capacidad de entrada máxima (hojas)

Figura 53. Catàleg de productes d'una determinada Categoria i Marca.

Conclusions

Un cop finalitzat el projecte es pot afirmar que s'han assolit els objectius d'aquest TFC, que eren el aprofundir en el coneixement de la tecnologia .NET i concretament amb el desenvolupament d'una aplicació utilitzant un patró MVC.

El treball inicial ha estat dur degut al desconeixement d'aquesta plataforma i del llenguatge C#, la recerca d'informació de com tenir una estructuració i una programació acurada i intentant complir amb les nomenclatures establertes per aquest llenguatge de programació.

El més complicat va ser localitzar un patró MVC per a poder treballar amb aplicacions d'escriptori de Windows, ja que la majoria de patrons estan pensats per el desenvolupament d'aplicacions Web sobre ASP.NET i només es va trobar una referència a patrons MVC per a Windows Form, el UIP Application Block versió 2.0, suportat per Microsoft i desenvolupat per a la versió 1.1 de .NET Frameworks i Microsoft SQL Server 2000.

Tot i el desencís inicial al no trobar més alternativa, es va aprofundir en aquest framework, per tal de poder compatibilitzar-lo amb la versió 2.0 de .NET "només" va caler modificar l'esquema intern del UIP en un fitxer de configuracions XML, tasca que no va ser fàcil, ja que només calia eliminar una línia del fitxer, però davant la falta d'informació i de suport d'aquest framework i la poca experiència amb el maneig del *Debug* de Visual Studio 2005 va fer que aquesta tasca fos bastant dura.

Potser per aquesta mateixa raó, la necessitat de examinar el codi intern del framework del UIP, va permetre veure la potencia del llenguatge i de la plataforma, i entendre molt millor el funcionament del propi framework, no cal dir, que amb aquesta experiència ara es domina el maneig del depurador que porta integrat Visual Studio 2005.

Un altre aspecte que també s'ha aprofundit és la tecnologia de Microsoft SQL Server 2005, s'ha pogut experimentar amb una base de dades actual la potencia i la senzillesa de maneig d'aquesta basa de dades i de la integració amb la plataforma .NET.

L'experiència ha servit per comprovar la importància que té la programació en capes separant la lògica de negoci de la interfície d'usuari, cosa que en la majoria dels casos no es fa i complica molt el manteniment, ampliació i migració a d'altres entorns de treball de les aplicacions.

No cal dir que el projecte era de caire didàctic que només és una petita part de tot el que es podria incorporar a una gestió comercial, on el catàleg de productes només és una petita part.

A nivell personal estic molt satisfet pels coneixements assolits amb les diferents tecnologies i mètodes de programació, cosa que permetrà en un futur immediat posar-los en pràctica per obtenir uns productes de programari de qualitat amb menys esforços i alta portabilitat.

Glossari

Cas d'ús. Acció que duu a terme un actor sobre un objecte, un actor no ha de ser necessàriament una persona.

CLR. Common Language Runtime. Motor de .NET Framework, compilador JIT que permet que una aplicació es pugui executar sobre diferents plataformes.

Compilador. Programari que tradueix el codi font d'alt nivell en codi màquina que entén el computador.

Compilador JIT (Just-in-time). Genera el codi màquina real en el moment en el que s'executa l'aplicació a partir d'un codi intermedi.

Control d'usuari. Porció de codi que integra la vista i el controlador i que es pot reutilitzar fàcilment.

Diagrama ER. Entitat-Relació, diagrama que mostra les taules d'una base de dades i les relacions que hi ha entre elles.

Diagrama de Gantt. Diagrama habitualment utilitzat per a representar les etapes d'un projecte.

Framework .NET. Marc on corren les aplicacions independentment del sistema operatiu. Llançat per Microsoft com a resposta a la plataforma Java.

Model de dades. Estructura utilitzada per a guardar les dades amb un programari gestor de dades.

MVC. Model – Vista – Controlador. Patró concret amb la finalitat de separar la codificació de la presentació de l'aplicació.

Objectes del domini. Objectes del món real detectats en la fase d'anàlisi.

Objectes del negoci. Objectes que representen programari específic.

Patró. Estructura arquitectònica utilitzada en el disseny d'aplicacions.

XML. Format estàndard d'intercanvi de dades que permet la comunicació entre diferents sistemes.

WorkFlow. La traducció és flux de treball. S'utilitza per descriure els processos interns que es realitzen en les empreses o en el programari, on es realitzen unes tasques en base a unes regles definides.

Bibliografia

- Document "Indicacions per a la redacció de la memòria TFC" editat per la UOC.
- Document "Planificació de projectes" editat per la UOC.
- John Sharp. Aprenda ya Microsoft Visual C#.NET. Ed. Microsoft Press, 2002.
- Francisco Charte. Visual C#.NET. Ed. Anaya Multimedia, 2002.
- Bill Evjen i més. Professional ASP.NET 2.0. Ed. Wiley Publishing, Inc., 2006
- Simon Robinson i més. Professional C# 3rd Edition. Ed. Wiley Publishing, Inc. 2004.
- Matthew MacDonald i Mario Szpuszta. Pro ASP.NET 2.0 in C# 2005. Ed. Apress, 2005
- Diversos. Professional ADO.NET 2 Programming with SQL Server 2005, Oracle, and MySQL. Ed. Wiley Publishing, Inc. 2006.
- John Paul Mueller. Visual C#.NET Developer's Handbook. Ed. Sybex Inc., 2002.
- Enciclopèdia Wikipedia
<http://es.wikipedia.org/>
- Documentació on-line de Microsoft
<http://msdn.microsoft.com/>
- Comunitat .Net
<http://www.codeproject.net/>
- Microsoft MELL

Annexos

Annex A. Instruccions per a la instal·lació

La instal·lació de la aplicació es molt senzilla, ja que només cal copiar el fitxer executable TFCNET.exe i la resta de fitxers .DLL en una carpeta des de la que es vulgui executar l'aplicació. També farà falta copiar el fitxer TFCNet.exe.config, que es el fitxer de configuracions de l'aplicació, aquest fitxer és en format XML i és fàcilment modificable amb qualsevol editor de text.

Després ens caldrà instal·lar la base de dades en el motor del SQL Server 2005, per això cal crear la base dades amb nom TFCDades en el SQL Server emprant els fitxer subministrats, en aquest fitxers hi ha donades d'alta una seria de fitxes que han servit per a fer les proves.

Possiblement caldrà modificar el fitxer de configuracions per a indicar en el *connectionString* quin és el nom del servidor SQL Server 2005.

```
<appParams>  
  <add key="ConnectionString" value="Data Source=.\\sqlexpress;Initial  
Catalog=TFCDades;Integrated Security=True" />  
</appParams>
```

On tindrem que posar el nom del nostre servidor SQL Server.

Annex B. Instruccions d'ús

Si bé l'aplicació és molt simple i intuïtiva en el seu maneig, cal com a mínim donar les claus per a poder entrar.

Els usuaris definits són:

Usuari	Clau	Tipus
admin	admin	Administrador
matias	matias	Administrador
user	user	Client