# COESIG:

# Optimització d'estoc i generació de pressupostos amb comunicació xifrada

David Horta Domingo

**Resum**—*"COESIG"* és un projecte que pretén crear un sistema que pugui ser emprat en la realitat laboral de l'empresa *"Quadres Horta S.L."*, permetent per un costat el control i l'optimització de les motllures del magatzem i per l'altre la generació de pressupostos. Aquest sistema està compost per una aplicació que funciona sobre una base de dades relacional. A més, la comunicació entre l'aplicació i la base de dades està xifrada per mitjà de certificats digitals. Així doncs, la connexió a base de dades requereix *"SSL"* el que fa que el sistema global sigui segur davant de qualsevol amenaça externa.

**Paraules clau**—control d'estoc, optimització d'estoc, pressupost, seguretat, certificats digitals.

**Abstract**—*"COESIG"* is a project created which the aim of developing a system that can be used in the reality of the company *"Quadres Horta S.L."*, allowing control and optimization of the warehouse frames and automatizing budgets for customers. This system is composed by an application that operates on a relational database. In addition, communication between the application and the database is encrypted using digital certificates. That means that database connection requires *"SSL"* which makes the whole system secure against any external threat.

**Index Terms**— stock control, stock optimization, budget, security, digital certificates.

---


- *E-mail de contacte: hortalimon@gmail.com*
- *Menció realitzada: Tecnologies de la Informació.*
- *Treball tutoritzat per: Jaume Pujol Capdevila (dEIC)*
- *Curs 2013/2014*


## 1 INTRODUCCIÓ

*Quadres Horta S.L.* és una empresa especialitzada en marcs a mida que disposa de les motllures més actuals del mercat, ampliant i renovant constantment la gama per satisfer en tot moment les exigències del mercat.

No obstant, l'empresa no porta un control d'estoc sobre les motllures que hi ha als magatzems i que provenen d'arreu d'Europa. A més, els pressupostos pels clients es realitzen manualment, el que significa una inversió molt gran en temps i capital.

### 1.1 Motivació del treball

La motivació fundamental del projecte és automatitzar el procés de generació de pressupostos i optimitzar l'estoc per tal de fer més l'empresa més rentable.

### 1.2 Motivació personal

Per altre banda, personalment vam trobar molt interessant aquest projecte perquè suposava una inversió d'hores que produiria un resultat aprofitable en el món empresarial.

### 1.3 Objectius del projecte

El projecte gira al voltant de cinc objectius fonamentals:

1. Control d'estoc de les tires de motllures que hi ha al magatzem.
2. Optimització de les tires de motllures que hi ha en estoc alhora de muntar un quadre.
3. Automatització de la generació de pressupostos.
4. Comunicació segura entre el programa i la base de dades.

5. Disseny i creació d'una base de dades.

Posteriorment, es detallaran els requisits del sistema.

# 2 ESTUDI DE LA SITUACIÓ ACTUAL

## 2.1. Context

L'empresa "Quadres Horta S.L." es troba situada al polígon industrial de Can Roses (*Rubí*). Dins de la nau industrial, existeixen dues plantes. La planta inferior és el magatzem i la zona de fabricació, mentre que la planta superior és un despatx on es porten la comptabilitat i les comandes. Es disposa d'un ordinador al magatzem i d'un ordinador al despatx. Com s'ha introduït en la descripció inicial, el projecte vol permetre l'optimització i el control de l'estoc, juntament amb l'elaboració de pressupostos tot garantint una comunicació segura entre l'ordinador de planta i el del despatx.

## 2.2. Descripció física del sistema

L'ordinador del despatx serà emprat per l'administratiu de l'empresa i contindrà la BD mentre que l'ordinador de planta serà emprat pel treballador de planta i es connectarà a la base de dades local de l'altre ordinador. Ambdós ordinadors tindran instal·lada l'aplicació que es connectarà a la base de dades per mitjà de certificats digitals i comunicació xifrada (connexió SSL).

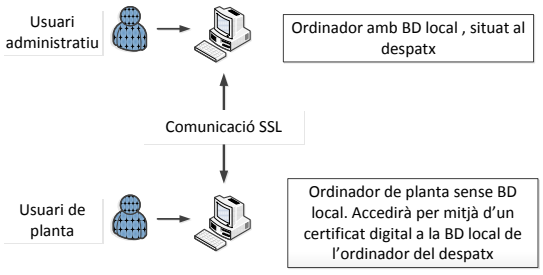



Fig. 1. Gràfic explicatiu sobre la descripció física del sistema

## 2.3. Descripció lògica del sistema

Per dissenyar la base de dades s'ha emprat "*MySQL*". Aquesta decisió no ha estat senzilla però es fonamenta en diversos aspectes.

En un principi, es volia aprofitar la base de dades que l'empresa té allotjada a un servidor de 1and1. No obstant, aquesta opció va ser descartada perquè l'empresa 1and1 té un tallafocs que no permet a cap aplicació establir una connexió des de fora del propi servidor. Descartada aquesta opció, es va decidir muntar una base de dades local sobre la que existís un control total. Els coneixements sobre Oracle eren extensos perquè és el motor de base de dades que s'ha emprat al llarg del grau. Tot i això, al final es va decidir emprar "*MySQL*" perquè era completament gratuït i perquè per les necessitats de l'empresa, "*MySQL*" compleix amb escreixos tots els requisits i estalvia problemes de llicències.

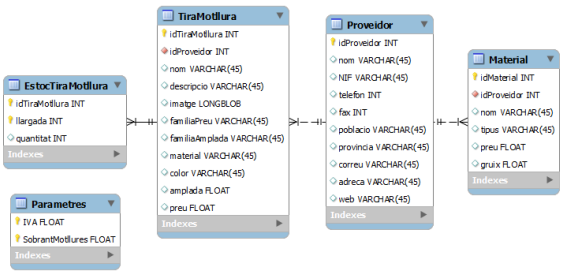



Fig. 2. Diagrama de la base de dades creat amb *"MySQL Workbench"*. El signes d'exclamació indiquen les claus primàries mentre els rombes rosats indiquen les claus foranies.

## 2.4. Aspectes legislatius

El sistema central està situat a l'empresa, és a dir, a Rubí. Per tant, alhora de tractar la informació de caràcter personal ens regirem per la *llei espanyola de protecció de dades (LOPD)*. [1]

# 3 REQUISITS DEL SISTEMA

## 3.1. Requisits funcionals

1. El programa permet introduir noves motllures a la base de dades i modificar o eliminar les existents.

2. Els programa permet veure la quantitat d'estoc que hi ha de cada una de les motllures que tenim a la base de dades.

3. El programa genera codis de barres a partir de l'identificador de la tira de motllura.

4. El programa permet la selecció d'imatges de motllures.

5. El programa permet la inserció de nous materials o la modificació dels mateixos.

6. El programa permet el control i l'optimització d'estoc.

7. El programa permet el càlcul de pressupostos, tot tenint en compte els preus de la motllura i els materials que es vulguin fer servir en l'elaboració d'un quadre.

8. El programa ha de permet l'ús d'un lector de codis de barres.

9. La comunicació entre el programa i la base de dades ha de ser xifrada (essencial).

## 3.2. Requisits no funcionals

1. Tots els paràmetres del programa són parametritzables (p.e. IVA).

2. El sistema central té una base de dades normalitzada.

### 3.3. Restriccions del sistema

1. Les consultes a la base de dades central no triguen més de 5 segons.

2. El software ha desenvolupar funciona sobre Windows 8.

## 4 METODOLOGIA

En aquest projecte es fa servir un desenvolupament iteratiu, ja que es considera el més adequat en projectes en els que només participa una persona. Com s'ha estudiat al llarg del grau, el model iteratiu és un procés de desenvolupament del software en el que els programes es van creant de manera incremental. És a dir, a cada etapa tenim un resultat, que s'itera i s'incrementa en la següent etapa. Això dóna una motivació extraordinària al programador alhora que evita les debilitats del model tradicional en cascada *("waterfall")*, ja que els errors es poden detectar i corregir molt abans.

El procés consisteix bàsicament en dues etapes:

1. En l'etapa inicial, es crea una primera versió del sistema.
2. En l'etapa d'iteració, es millora la idea corregint els problemes de la última versió que estigués disponible i s'augmenta la funcionalitat.

L'inconvenient destacable d'aquest mètode és la necessitat de col·laboració constant del client al llarg de tot el projecte. No obstant, al disposar d'un contacte directe amb el client això no resulta un inconvenient.

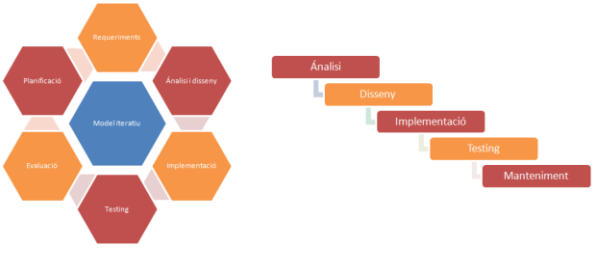
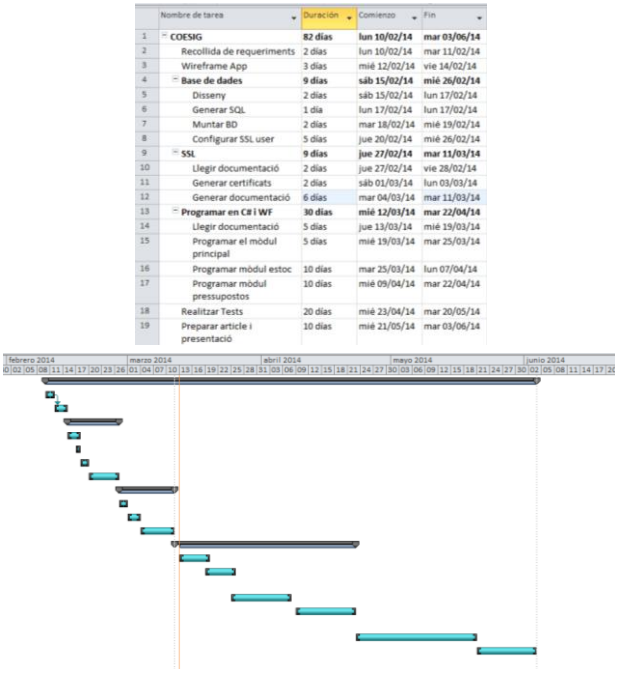



Fig. 3. Diagrames comparatius: a l'esquerra el model iteratiu i a la dreta el model en cascada.

Dins d'aquest tipus de desenvolupament iteratiu, trobem la tècnica SCRUM. Bàsicament, al principi del projecte s'escriuran en pòsits els objectius del projecte. Cada setmana es definiran una sèrie d'objectius (una sèrie de pòsits) per la següent setmana. Després, s'analitzaran els que s'han complert i es tornaran a definir nous objectius per la següent.

## 5 PLANIFICACIÓ

Tot i fer servir un model iteratiu, per tenir una millor visió global de l'estat del projecte, vam fer servir un diagrama de Gantt amb les principals fites.



| | Nombre de tarea | Duración | Comienzo | Fin |
|---|---|---|---|---|
| 1 | COESIG | 82 días | lun 10/02/14 | mar 03/06/14 |
| 2 | Recollida de requeriments | 2 días | lun 10/02/14 | mar 11/02/14 |
| 3 | Wireframe App | 3 días | mié 12/02/14 | vie 14/02/14 |
| 4 | Base de dades | 9 días | sáb 15/02/14 | mié 26/02/14 |
| 5 | Disseny | 2 días | sáb 15/02/14 | lun 17/02/14 |
| 6 | Generar SQL | 1 día | lun 17/02/14 | lun 17/02/14 |
| 7 | Muntar BD | 2 días | mar 18/02/14 | mié 19/02/14 |
| 8 | Configurar SSL user | 5 días | jue 20/02/14 | mié 26/02/14 |
| 9 | SSL | 9 días | jue 27/02/14 | mar 11/03/14 |
| 10 | Llegir documentació | 2 días | jue 27/02/14 | vie 28/02/14 |
| 11 | Generar certificats | 2 días | sáb 01/03/14 | lun 03/03/14 |
| 12 | Generar documentació | 6 días | mar 04/03/14 | mar 11/03/14 |
| 13 | Programar en C# i WF | 30 días | mié 12/03/14 | mar 22/04/14 |
| 14 | Llegir documentació | 5 días | jue 13/03/14 | mié 19/03/14 |
| 15 | Programar el mòdul principal | 5 días | mié 19/03/14 | mar 25/03/14 |
| 16 | Programar mòdul estoc | 10 días | mar 25/03/14 | lun 07/04/14 |
| 17 | Programar mòdul pressupostos | 10 días | mié 09/04/14 | mar 22/04/14 |
| 18 | Realitzar Tests | 20 días | mié 23/04/14 | mar 20/05/14 |
| 19 | Preparar article i presentació | 10 días | mié 21/05/14 | mar 03/06/14 |

Fig. 4. Fites i diagrama de Gantt corresponent.

## 6 ANÀLISI DEL SITEMA

### 6.1. Algorisme d'optimització

Com a exemple per poder explicar millor aquest algorisme, es suposará que el client vol fabricar un quadre de 50cm x 70cm (mida exterior). La motllura seleccionada serà la "566" i tindrà el següent estoc de tires: dues de 50cm, dues de 80cm i una de 100cm. Quan s'apreti el botó "calcular" es donaran les dues úniques solucions que hi ha en aquest cas (eliminant repetides).

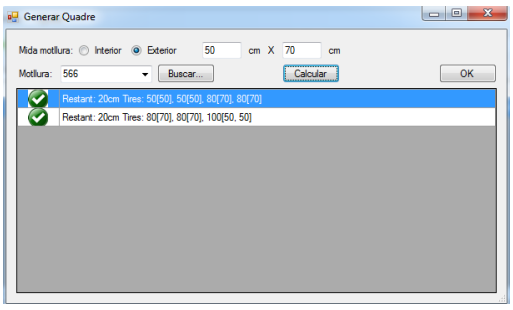



Fig. 5. Resultat final de l'exemple proposat

A continuació es detalla quin és l'algorisme que permet obtenir el que mostra la "fig. 5":

1. Es crea una llista amb les llargades de tot l'estoc de la motllura seleccionada per l'usuari. En l'exemple, la llista és (50,50,80,80,100).

2. Es crea una llista amb les llargades que introdueix l'usuari. En l'exemple l'usuari vol fer un quadre de 50cmx70cm (exterior) i la llista és (50,50,70,70).

3. Es crea una llista amb les llargades dels trossos agafats (quan s'està realitzant l'arbre d'expansió).

4. Es crea una llista amb les llargades dels trossos que falten per agafar (quan s'està realitzant l'arbre d'expansió).

5. Es comencen a expandir les combinacions que caben en les tires que hi ha a l'estoc.

6. S'agafa la tira amb menys tros restant.

7. Es treuen les llargades que ja s'han trobat i les tires que ja s'han utilitzat.

8. Quan s'han trobat totes les llargades, es té una solució.

9. Si encara queden tires per utilitzar, es busca la solució dels conjunts de llargades i de tires restants.

10.S'afegeixen les solucions i s'ordenen segons com d' òptimes siguin (s'agafen les 5 millors solucions).

1

| B = 70 | B = 70 | A = 50 | A = 50 | total |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 50 |
| 0 | 0 | 1 | 0 | 50 |
| 0 | 1 | 0 | 0 | 70 |
| 1 | 0 | 0 | 0 | 70 |
| 0 | 0 | 1 | 1 | 100 |
| 0 | 1 | 0 | 1 | 120 |
| 0 | 1 | 1 | 0 | 120 |
| 1 | 0 | 0 | 1 | 120 |
| 1 | 0 | 1 | 0 | 120 |
| 1 | 1 | 0 | 0 | 140 |
| 0 | 1 | 1 | 1 | 170 |
| 1 | 0 | 1 | 1 | 170 |
| 1 | 1 | 0 | 1 | 190 |
| 1 | 1 | 1 | 0 | 190 |
| 1 | 1 | 1 | 1 | 240 |

2

| B = 70 | B = 70 | A = 50 | total |
|---|---|---|---|
| 0 | 0 | 1 | 50 |
| 0 | 1 | 0 | 70 |
| 1 | 0 | 0 | 70 |
| 0 | 1 | 1 | 120 |
| 1 | 0 | 1 | 120 |
| 1 | 1 | 0 | 140 |
| 1 | 1 | 1 | 190 |

3

| B = 70 | B = 70 | total |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 70 |
| 1 | 0 | 70 |
| 1 | 1 | 140 |

Taula 1: Taules que mostren la primera passada en l'arbre d'expansió de l'exemple. En la taula 1, es tenen totes les combinacions. En la taula 2, les combinacions després d'agafar primer el tros de motllura de 50. En la taula 3, les combinacions després d'agafar l'altre de 50. Després s'agafarà el de 70, el que reduirà l'última opció a 70.

La resta de passades es fan exactament igual fins a fer totes les combinacions possibles (15 passades ja que el zero no es compta).

L'algoritme no expandeix les combinacions que, basant-se en l'estoc, no són possibles i evita també els resultats repetits.

L'últim pas com ja s'ha explicat és ordenar les tires de més a menys òptimes i fins a un màxim de cinc.

Si la profunditat de l'arbre d'optimització és molt gran, el programa pot veure afectat el seu rendiment (ja que es necessitarà una gran capacitat de còmput). No obstant, en el projecte es considera la millor alternativa, ja que un quadre sempre es fabrica amb quatre trossos i l'arbre d'expansió no portarà mai cap problema perquè no tindrà en cap cas, més de quatre nivells de profunditat.

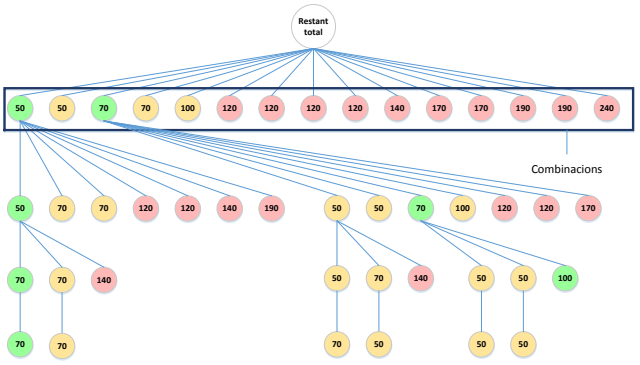



Fig. 6. Arbre d'expansió de l'exemple on s'indica en verd les combinacions seleccionades, en groc les vàlides però repetides i en vermell les que no són possibles degut a l'estoc existent de la motllura que el client ha seleccionat.

## 6.2. Aspectes de seguretat

### 6.2.1. Preparació per la creació de certificats digitals en "Windows"

El primer pas per la generació de certificats digitals per COESIG va ser la descarrega del paquet "Visual C++ 2008 Redistributables".

Després es va descarregar el paquet *OpenSSL Light (que és gratuït)*. [2] Depenent de l'arquitectura, la ubicació d'instal·lació per defecte serà "C:\OpenSSL-Win32" o "C:\OpenSSL-Win64". COESIG utilitza l'arquitectura de 32 bits. Al acabar la instal·lació, es va afegir "C:\OpenSSL-Win32\bin" a les variables d'entorn de Windows.

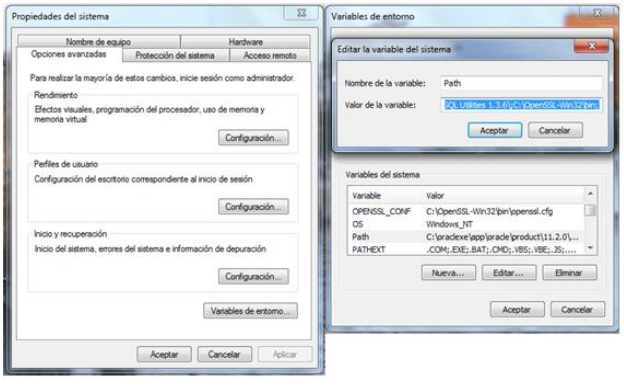



Fig. 7. Mostra la creació de la variable d'entorn "OpenSSL".

### 6.2.2. Creació de certificats digitals

Un cop instal·lat "OpenSSL" es van generar els certificats digitals del servidor i del client a partir de l'autoritat de certificació (CA).

Per fer-ho es van seguir els passos del manual *Setting Up SSL Certificates and Keys for MySQL*. [3] Al final es va executar l'opció de verificació dels dos certificats digitals.

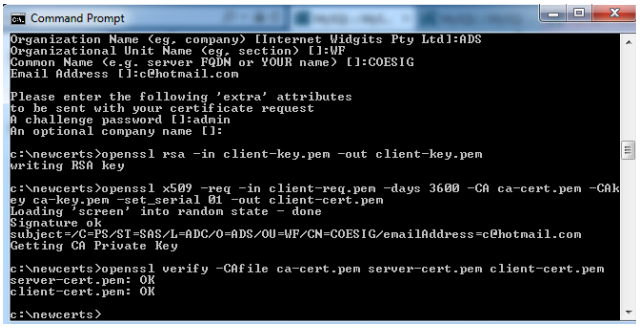



Fig. 8. Verificació del certificat del servidor i del client.

### 6.2.3. Configuració de la base de dades amb SSL

Per comprovar quin és l'estat actual del sistema ssl a la base de dades es pot fer servir la comanda "*mysql> show global variables like 'have_%ssl'*". Sempre sortirà per defecte disabled.

Per fer la modificació, s'ha d'anar al fitxer d'opcions de la base de dades i afegir la ruta del arxiu que conté les autoritats de confiança SSL i el certificat X509 i la clau X509 del servidor en format PEM. També s'han d'habilitar les connexions SSL.

Un cop s'ha acabat amb la part del servidor, només cal crear un usuari (client) que requereixi ssl: *GRANT ALL PRIVILEGES ON *.* TO 'quadreshortassl'@'localhost' IDENTIFIED BY 'admin' REQUIRE SSL.* S'ha d'especificar també a la pestanya ssl les autoritats de confiança SSL i el certificat X509 i la clau X509 del client en format PEM.

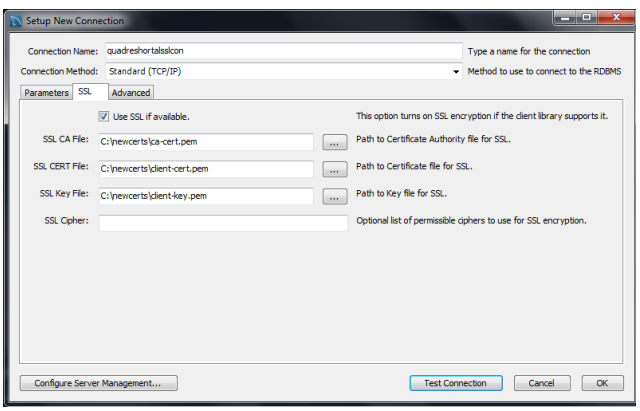



Fig. 9. Pantalla amb la configuració SSL del client.

Per últim, per realitzar la connexió entre COESIG i la base de dades, serà necessari que l'aplicació es connecti amb SSL sinó la base de dades la refusarà ('*SSL Mode=Required*').

## 7 CODIFICACIÓ I TEST DE L'APLICACIÓ

### 7.1. Plantejament del codi

El codi es divideix en quatre carpetes:

1. La carpeta **"Recursos"** conté totes les imatges (icones) que el programa necessita al funcionar.

2. La carpeta **"DAO"** ("Data Access Object") conté la interfície entre el programa i la base de dades "MySql". Els DAO són un patró de disseny (segons Core J2EE) i estan considerats una bona pràctica en el món del disseny del SW. *L'avantatge d'utilitzar DAO és que qualsevol "Objecte de Negoci" (aquell que conté detalls específics d'operació o aplicació), no requereix coneixement directe de l'origen o destí definitiu de la informació que manipula.* [4] En resum, la resta del programa queda aïllat de la base de dades. Això permet que si es produeix un canvi a BD, la resta del programa no es vegi afectat. A més, les classes DAO contenen tota la lògica de creació, lectura, modificació i esborrat (model CRUD).
El model CRUD ("Create, Read, Update, and Delete") és tan vàlid per HTTP com per SQL ja que és acrònim de les quatre operacions bàsiques que han de permetre les bases de dades (i en general, qualsevol mitjà que permeti emmagatzemar informació de manera persistent).

| Operació | SQL | HTTP |
|---|---|---|
| **Create** | INSERT | PUT / POST |
| **Read** | SELECT | GET |
| **Update** | UPDATE | PUT / PATCH |
| **Delete** | DELETE | DELETE |

Taula 2: Taula que mostra la traducció de les operacions CRUD a "SQL" i "HTTP".

```
namespace Coesig.DAO
{
    public class ProveidorDAO : BaseDAO
    {
        public bool crearProveidor(Proveidor proveidor)...
        public List<Proveidor> getProveidor()...
        public bool actualitzarProveidor(Proveidor proveidor)...
        public bool esborrarProveidor(Proveidor proveidor)...
    }
}
```

Fig. 10. Exemple de la classe "ProveidorDAO" seguint el model CRUD

3. La carpeta **"Dades"** conté les classes que fan accessible a la resta del programa la informació que tenim a base de dades i que s'obté per mitjà de les classes DAO. Com a exemple tenim la classe "Material", on podem veure com tots els paràmetres que li arriben de la classe "MaterialDAO" són fets accessibles a la resta del programa.

```
namespace Coesig.Dades
{
    public class Material : IStock
    {
        public int IdMaterial { get; set; }
        public int IdProveidor { get; set; }
        public string Nom { get; set; }
        public string Tipus { get; set; }
        public float Preu { get; set; }
        public float Gruix { get; set; }
    }
}
```

Fig. 11. Exemple de la classe "Material", on podem veure com tots els paràmetres que li arriben de la classe "MaterialDAO" són fets accessibles a la resta del programa.

4. La carpeta de **"Forms"** conté la lògica de cada finestra del programa. Aquesta carpeta és necessària en projectes C# i Windows Forms.

### 7.2. Interfícies

Una interfície conté les definicions d'un grup de funcions relacionades que una classe o struct pugui implementar. Mitjançant les interfícies, pot incloure, per exemple, comportament de diversos orígens en una classe. Aquesta funció és important en C# perquè el llenguatge no admet l'herència múltiple de classes.

*Característiques de les interfícies:*

*1. Una interfície és com una classe base abstracta. Qualsevol classe o struct que implementi la interfície ha d'implementar tots els seus membres.*

*2. Una interfície no pot ser instanciada directament.*

*3. Les interfícies poden contenir esdeveniments, mètodes, indexadors i propietats.*

*4. Les interfícies no contenen implementacions de mètodes.*

*5. Una classe o struct pot implementar diverses interfícies. Una classe pot heretar una classe base i també implementar una o més interfícies.* [5]

COESIG calcula els pressupostos fent ús de la interfície "IStock". En la linia de codi "*CostBase += stock.CalcularCost(midaQuadre)*" el programa calcula el cost d'una manera o d'una altra depenen si es tracta d'un material o d'una tira de motllura.

```
public class TiraMotllura : IStock
{
    public int IdTiraMotllura { get; set; }

    public int IdProveidor { get; set; }

    public float CalcularCost(Size midaMotllura)
    {
        return (midaMotllura.Height + midaMotllura.Width + 4 * Amplada) * Preu / 50; // * 2 / 100
    }
}

public class Material : IStock
{
    public int IdMaterial { get; set; }

    public int IdProveidor { get; set; }

    public float CalcularCost(Size midaMotllura)
    {
        return midaMotllura.Height * midaMotllura.Width * Preu / 10000;
    }
}
```

Fig. 12. Els següents fragments de les classes "material" i "tira motllura" mostren les dues maneres de calcular el cost.

### 7.3. Test

Durant el període de desenvolupament del software, s'ha realitzat una inversió de temps important a la realització de tests. COESIG valida absolutament tots els camps que l'usuari introdueix per teclat i té també en compte les claus primàries i foranies de la base de dades. D'aquesta manera s'evita l'aparició d'excepcions no controlades. A més, s'indica a l'usuari quin paramtre és l'incorrecte i que ha de fer per corregir-lo (es dóna "feedback"). Per acabar els tests, es va generar una versió beta del programa perquè l'usuari acabés de corroborar el bon funcionament, no tan sols de l'aplicació, sinó de tot el sistema.

## 8 DISSENY DE L'APLICACIÓ

### 8.1. Pantalla principal

Només obrir el programa es mostren totes les motllures de l'empresa i al fer doble clic sobre qualsevol d'elles, es pot gestionar l'estoc. El mateix passaria amb la resta de materials. Per veure els altre materials, només cal moures per les tres pestanyes: "motllures", "materials" i "proveïdors". Al costat de cada producte, hi ha dues icones: una per editar i l'altre per eliminar. A la part superior, es troba una creu verda que permet al client afegir un nova motllura, material o proveïdor sense la necessitat de fer servir sempre el menú desplegable de la banda superior. Així doncs, el client pot afegir el que desitgi d'una manera ràpida i intuïtiva.

El programa disposa també d'un filtre perquè el client pugui buscar d'una manera fàcil el que desitja. Els filtres es mostren al fer clic a l'icona de la lupa.
Per últim, per generar pressupostos o per optimitzar l'estoc, s'ha d'accedir a la pestanya d'inici on apareixerien aquestes opcions.

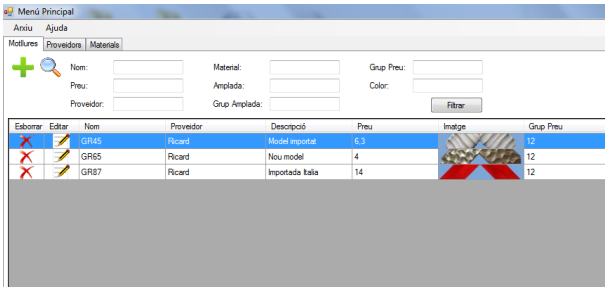



Fig. 13. Detall de la pantalla d'inici del programa.

### 8.2. Pantalla per generar pressupostos

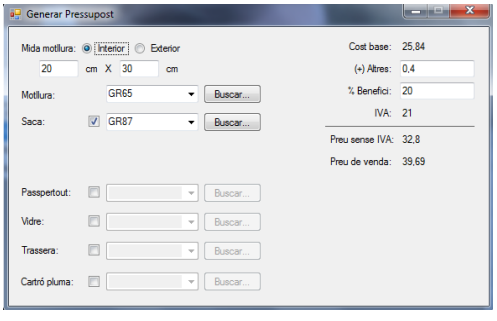



Fig. 14. Pantalla per generar pressupostos

Primer, el client selecciona si la mida que introdueix és externa o interna. Això és molt important alhora de calcular el cost de la motllura, ja que el programa agafa el cost lineal de la motllura que selecciona l'usuari i el multiplica pel perímetre del quadre.

Si la mida és interior, caldrà suma l'amplada de la motllura per saber el que s'ha gastat. Si la mida és exterior, s'agafa directament el perímetre i es multiplica pel preu del metre lineal de la motllura escollida.

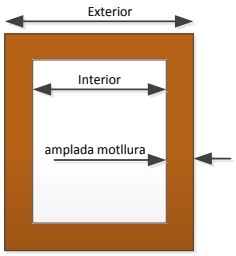



Fig. 15. Esquema de les mides d'una motllura.

S'ha de tenir en compte, que quan es talla una tira es perden trossos; el que fa que l'empresa hagi de cobrar la mida exterior perquè és el que realment es perd.

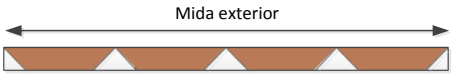



Fig. 16. Detall del tross de motllura perdut en els talls de la tira.

La *saca* és opcional. Una *saca* és una segona motllura que es situa a l'interior de la principal. Això significa que la mida interior de la motllura és l'exterior de la saca. Aleshores és tan senzill com multiplicar el perímetre de la mida interior de la motllura pel preu del metre lineal de la saca.

Per continuar fent els càlculs, es necessita saber si el quadre té o no saca. En cas de que tingui saca; el vidre, la trassera, el passpertout i el cartró ploma aniran sempre per dins. Això significa que haurem de calcular la mida interior de la saca restant la seva amplada. Sinó té saca, agafarem les mides interior de la motllura per fer els càlculs de cost dels materials.

Tots els costos dels materials es calculen en preu per metre quadrat. Així doncs, haurem de calcular l'àrea que hi ha dins de la motllura o dins de la saca (si en porta) i multiplicar pel preu del metre quadrat. Els quadres de text de la saca, de la motllura i dels materials s'autocompleten. No obstant, sinó recordem el nom i volem fer una cerca entre totes les motllures, les saques o els materials, tenim el botó buscar.

S'ha tingut en compte a nivell de codi que tot i que els materials estan junts perquè tenen els mateixos atributs, l'usuari quan busca un tipus en concret només vol veure els que són d'aquell tipus. Per exemple, si l'usuari vol veure vidres, el programa només li retorna els vidres.

Per últim tenim els valors de la dreta. El cost base s'actualitza a mesura que anem introduint dades i representa el que perd l'empresa al realitzar el quadre en qüestió. Aquest preu només serveix a nivell intern. S'ha afegit també un quadre per algun detall molt especial que el client pugui demanar i que s'hagi de sumar al final de tot de manera excepcional. Sota, podem introduir el percentatge de benefici que volem incloure i el programa ens calcularà a quin preu l'hem de vendre al client.

L'IVA que es fa servir en els càlculs és parametritzable des del menú desplegable de l'aplicació. De moment té un valor d'un 21%.

### 8.3. Pantalla de control d'estoc

Si es fa doble clic sobre qualsevol motllura es pot gestionar l'estoc per mitjà de la pantalla mostrada a la "fig. 17".

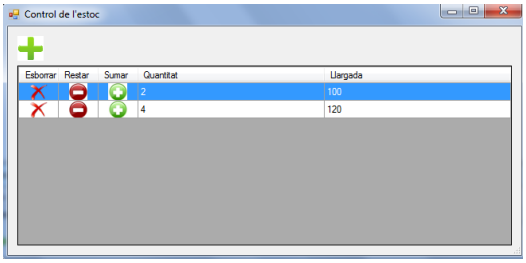



Fig. 17. Pantalla per controlar l'estoc

• Creu verda: Obre una pantalla per afegir estoc d'una llargada diferent a les existents. Si l'usuari afegeix estoc d'una llargada existent, el programa suma automàticament les unitats a les que ja hi havia.

• Creu vermella: Esborra tot l'estoc d'una determinada llargada.

• Icona de restar: Disminueix l'estoc d'una determinada llargada en una unitat. Si l'estoc arriba a "0" al disminuir, l'entrada s'esborra automàticament.

• Icona de sumar: Augmenta l'estoc d'una determinada llargada en una unitat.

### 8.4. Pantalla de detall de les motllures

Quan s'afegeix una nova motllura o es modifica una d'existent, es mostra una pantalla amb els seus detalls.
S'han ocultat els identificadors autoincrementals (tant el de la tira de motllura com el de proveïdor) que es fan servir com a clau primària a la BD. *Això respon a la necessitat de que el comportament intern de l'aplicació sigui transparent a l'usuari.* [6]

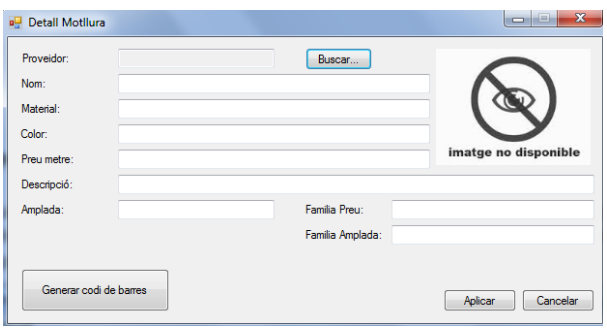



Fig. 18. Pantalla que mostra els detalls d'una motllura

També s'ha creat un botó de buscar proveïdor. Quan es fa clic, s'obre una pantalla que mostra una llista dels proveïdors que hi ha a la BD. Això evita que l'usuari pugui afegir un proveïdor inexistent i garanteix la consistència de la BD i del programa. *A més, es validen tots els camps i es mostren missatges de "feedback" a l'usuari* (recomanat a [6]).

### 8.5. Pantalla per generar codis de barres

Quan es fa clic al botó generar codi de barres s'obre aquesta finestra. Per defecte s'agafa el nom de la motllura per generar el codi de barres, però l'usuari pot modificar el que cregui convenient i tornar a generar el codi que desitgi. Fent un simple clic amb el ratolí el codi de barres es copia al porta papers del sistema operatiu i l'usuari el pot enganxar a qualsevol processador de texts i generar tantes etiquetes com vulgui.


Fig. 19. Pantalla per generar codis de barres

Per la generació de codis de barres, s'ha fet servir la "*Barcode Image Generation Library*". [7]

| | | |
|---|---|---|
| Code 128 | Code11 | Code 39 |
| Code 93 | EAN-8 | EAN-13 |
| UPC-A | UPC-E | JAN-13 |
| MSI | ISBN | Standard 2 of 5 |
| Interleaved 2 of 5 | PostNet | UPC Supplemental 2 |
| UPC Supplemental 5 | Codabar | ITF-14 |
| Telepen | Pharmacode | FIM |

Taula 3: Llista de formats soportats per la Barcode Lib

COESIG fa servir el "code 128" perquè inclou tots els caràcters que el client necessita codificar. A més, el codi que permet generar el codi de barres és molt intel·ligible i fàcil de mantenir gràcies a la llibreria escollida.

### 8.6. Pantalla "Sobre COESIG"

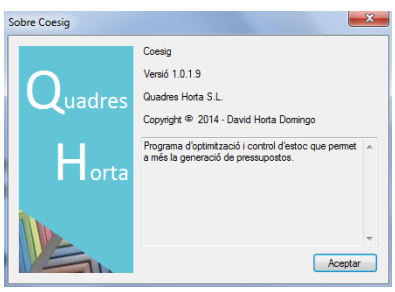



Fig. 20. Pantalla que conté la informació essencial del programa

Al menú desplegable dins de l'apartat "ajuda" hi ha la finestra "sobre COESIG". Aquesta finestra conté la informació obtinguda de les metadades del projecte. Aquí es pot veure la versió del programa, el client propietari i els drets d'autor a més d'una descripció de la funcionalitat bàsica del programa.

### 8.7. Pantalla de generació de quadres

La pantalla de generació de quadres engloba, juntament amb la pantalla que permet generar els pressupostos, la part més interessant del programa. Aquesta pantalla és accessible des del menú desplegable i permetrà optimitzar l'estoc de l'empresa. Primer, s'ha de seleccionar si la mida del quadre que es vol fabricar és interna o externa. Això és molt important alhora d'optimitzar, ja que el programa necessita saber la mida exterior de la motllura per poder aplicar l'algoritme d'optimització.
S'ha de tenir en compte, que quan es talla una tira, la mida exterior és el que realment es perd. Així doncs, si la mida és interior, caldrà sumar l'amplada de la motllura.

El quadres de text de la motllura s'autocompleta. No obstant, sinó es recorda el nom i es vol fer una cerca entre totes les motllures, es pot fer clic al botó "buscar".

Un cop tots els paràmetres s'han introduït, només cal fer clic al botó "calcular". El programa retornarà quines tires de l'estoc són les més òptimes (és a dir, les que produeixen menys tros restant) ordenades de més a menys i fins un màxim de cinc. En l'apartat següent s'explica tots els detalls del codi i el funcionament d'aquesta pantalla per mitjà d'un exemple.

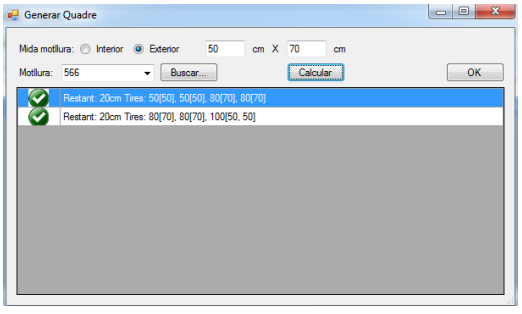



Fig. 21. Pantalla per optimitzar l'estoc

## 9 RESULTATS

El principal mètode d'avaluació utilitzat per verificar les bases del treball ha estat la implementació.
El contacte amb els stakeholders ha estat constant el que ha facilitat un resultat adequat a les necessitats de l'empresa.

Durant la implementació s'ha pogut comprovar si realment es podia crear un sistema que permetés incrementar els beneficis de l'empresa tant a nivell econòmic com a nivell d'inversió de temps.

Després de portar a terme la implementació, es va poder verificar que tots els objectius inicials s'havien satisfet. La creació d'un sistema no només de control d'estoc sinó d'optimització i que a més permet realitzar pressupostos en un món tan concret i peculiar com el de l'enmarcació ha estat possible. A més, el sistema ha demostrat ser segur davant amenaces i refusar qualsevol connexió que no fos SSL i certificada correctament.

Cal dir que aquests resultats eren els esperats, donat que la implementació venia recolzada en una bona planificació i estudi de la matèria que es tractava.

## 10 CONCLUSIONS

Els objectius proposats a l'inici de l'article s'han aconseguit resoldre amb solidesa (des de la cerca d'informació fins la finalització de tot el sistema COESIG).

COESIG està en funcionament des de juny. Això ha servit per rebre "feedback" del client i acabar de refinar els últims detalls. No obstant, l'aplicació havia estat testejada i s'havien validat tots els camps d'entrada. Cal tenir en compte, que com és una aplicació que reb molts paràmetres d'entrada s'han de realitzar moltes validacions per evitar errors durant l'execució del programa.
També s'han controlat totes les claus primàries de la base de dades i les seves possibles relacions amb claus foràànies.

En resum, l'aplicació ha estat validada no només abans de ser entregada al client sinó durant la seva integració a l'empresa.

A nivell personal, aquest projecte ens ha aportat un elevat grau de satisfacció. Després d'una gran inversió de temps, hem pogut veure que hem satisfet les necessitats d'una empresa real que a partir d'ara podrà incrementar els beneficis gràcies al nostre sistema.

### 10.1. Ampliacions

Una de les possibles ampliacions en les quals es podria continuar treballant és l'aspecte de l'optimització.
Un exemple podria ser si un client demana 50 quadres de la mateixa mida amb la mateixa motllura. En el nostre projecte l'arbre d'optimització és per un sol quadre i els resultats podríen ser diferents si tinguéssim en compte que es volen fabricar 50 quadres iguals (tindríem 200 trossos de motllures que es podríen combinar de moltes maneres per fer els quadres).

## 11 AGRAÏMENTS

Volem donar les gràcies al tutor del treball Jaume Pujol, a l'empresa "Quadres Horta S.L." per la seva constant col·laboració i a totes aquelles persones que amb les seves idees han fet possible un projecte millor.

## 12 BIBLIOGRAFIA

[1] "Guía del Responsable de Ficheros". AGENCIA ESPANOLA DE PROTECCIÓN DE DATOS, 1999. Web. 29 Març 2014.
*http://www.agpd.es/portalwebAGPD/canaldocumentacion/publicaciones/common/pdfs/guia_responsable_ficheros.pdf*

[2] "Openssl". OPENSSL, 2013. Web. 20 Febrer 2014.
*http://www.openssl.org/docs/apps/openssl.html*

[3] "Setting Up SSL Certificates and Keys for MySQL". MYSQL, 2004. Web. 20 Febrer 2014.
*https://dev.mysql.com/doc/refman/5.0/en/creating-ssl-certs.html*

[4] "DAO Design Pattern: A Core J2EE Design Pattern". CodeFutures, 2014. Web. 5 Abril 2014.
*http://www.codefutures.com/j2ee-design-pattern/*

[5] "Guía de programación de C#". MICROSOFT DEVELOPER NETWORK, 2013. Web. 5 Abril 2014.
*http://msdn.microsoft.com/es-s/library/ms173156.aspx*

[6] Debbie Stone, Caroline Jarrett, Mark Woodroffe, Shailey Minocha. User Interface Design and Evaluation. Elsevier, 2005. Print.

[7] "Barcode Image Generation Library". Brad Barnhill. Web. 27 Febrer 2014.
*http://www.codeproject.com/Articles/20823/Barcode-Image-Generation-Library*