SISTEMA DE GESTIÓ DE TALLERS

(Aplicació de tecnologies Web 2.0 a la planificació de la feina d' un taller mecànic)

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Rafael Flores Vergel
i dirigit per
Xavier Binefa Valls
Bellaterra, 16 de Juny de 2008

Universitat Autònoma de Barcelona

etse
Escola Tècnica Superior d'Enginyeria

El sotasignat, Xavier Binefa Valls

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Rafael Flores Vergel

I per tal que consti firma la present.

Signat: Xavier Binefa Valls

Bellaterra, 16 de Juny de 2008

Agraïments

Abans de començar amb el contingut del projecte, voldria aprofitar per agrair a aquelles persones que m'han recolzat tant durant aquest temps. En primer lloc, i com no podria ser d'altra manera, agrair a la meva família el recolzament que m'ha donat, i de manera molt especial a la meva dona, per la seva paciència, tant en els bons moments com en els més difícils.

En segon lloc, voldria agrair a Xavier Binefa, el meu director de projecte, per l'esforç que ha fet perquè aquest projecte sigui una realitat, fent possible la finalització dels meus estudis.

Per últim, agrair els comentaris i idees als meus companys de feina, els quals a més de considerar-los uns excel·lents professionals, els considero bons amics.

Resum del Projecte

El projecte consisteix en una aplicació que facilita la gestió d'un taller de reparació de vehicles, adquirint especial importància la planificació, control i seguiment dels treballs que porten a terme els operaris. L'aplicatiu es desenvoluparà en entorn web i pretendrà aproximar-se al nivell de prestacions que una aplicació tradicional d'escriptori, gràcies a l'ús de tecnologies Web 2.0 com AJAX, però amb tot el valor afegit que implica una aplicació web.

Resumen del Proyecto

El proyecto consiste en una aplicación que facilita la gestión de un taller de reparación de vehículos, adquiriendo especial importancia la planificación, control y seguimiento de los trabajos que llevan a cabo los operarios. El aplicativo se desarrollará en entorno web y pretende aproximarse al nivel de prestaciones que podría ofrecer un aplicativo tradicional de escritorio, gracias al uso de tecnologías Web 2.0 como AJAX, pero con todo el valor añadido que implica una aplicación web.

Project Summary

The project consists of an application that facilitates the management of a car workshop. It gives special importance to plan, control and monitoring of the work carried out by the operatives/workers. The application will be developed in web environment. It approximates to the level of benefits that could offer a traditional desktop application, thanks to the use of Web 2.0 technologies such as AJAX. But with all the added value that involves a web application.

1. Introducció

En aquest apartat es pretén fer una presentació inicial del projecte, marcant els objectius, motivacions personals i entorn de l'aplicació així com les limitacions d'aquesta. A més, es mostrarà una breu descripció del contingut de les diferents parts de la memòria.

1.1. Contingut del PFC

El projecte realitzat consisteix en una aplicació web que dóna resposta a algunes de les necessitats de gestió d'un taller de reparació de vehicles, especialment a la gestió de treballs, proveint-nos d'una eina visual per la planificació d'aquests, així com per la seva supervisió. Per altra banda, l'aplicatiu ens facilita l'anàlisi dels treballs, podent veure, per exemple, la desviació en el temps dedicat respecte al previst així com realitzar altres tasques molt habituals, com l'administració de clients i vehicles associats o la configuració de treballs habituals.

El sistema dissenyat es pot implantar a qualsevol taller de vehicles i, seguint l'actual tendència de construir grans aplicacions a partir de components més petits, es basa en una arquitectura que facilita la integració amb altres mòduls que el puguin completar. Això s'aconsegueix entre altres, amb l'ús de Web Services, que aporten gran independència entre l'aplicació que els utilitza i els propis Web Services.

En el projecte desenvolupat ens hem recolzat en tecnologies Web 2.0, que seran descrites en capítols posteriors, per obtenir una aplicació RIA (Rich Internet Application), on la principal característica és una interfície rica, que millora l'experiència de l'usuari. Per tant, en la implementació de l'aplicatiu he utilitzat de manera habitual Web Services i AJAX, elements vertebrants de Web 2.0, el qual farà que la implementació en entorn web no comporti els típics inconvenients d'aquest tipus d'entorn, però sí els seus avantatges, com són, per exemple, no requerir cap software addicional al client, tret d'un navegador d'Internet.

Des del punt de vista tecnològic, ens hem basat en el Servidor d'Aplicacions Internet Information Services (IIS) de Microsoft, emprant com a llenguatge de servidor ASP.NET 2.0 (VB.NET), i com a Repositori de dades SQL Server 2005. A més, s'han usat tecnologies com HTML, Javascript i CSS, entre altres.

Amb aquesta aplicació, gràcies a les tecnologies que hem utilitzat, totalment vigents i acceptades per la comunitat de desenvolupadors web, he pogut ampliar els meus coneixements d'aquest entorn, així com aplicar altres coneixements adquirits durant la carrera, com l'estudi d'algorismes, enginyeria del software, sistemes experts, bases de dades o xarxes.

1.2. Abast

A l'anterior apartat vaig introduir d'una manera més o menys general els objectius del projecte. A continuació comentarem aspectes concrets del seu abast, és a dir, característiques i limitacions de manera que quedaran clarament delimitades les fronteres fins a on s'entén el projecte, tant per la seva part positiva (allò que el projecte pretén fer) com per la negativa (allò que no es contempla).

El projecte es composa de dues parts ben diferenciades:

1. Una base de dades on s'emmagatzemarà tota la informació relacionada amb els clients i els seus vehicles, així com els treballs a realitzar en aquests.
2. Una aplicació capaç d'accedir a les dades emmagatzemades a la base de dades per portar a terme les tasques que els usuaris requereixen, sempre relacionades amb la gestió del taller mecànic.

L'aplicatiu facilitarà la gestió d'un taller mecànic donat que facilita algunes de les tasques més habituals, però en cap cas es tracta d'un ERP (Enterprise Resource Planning). Un ERP es caracteritza per tractar-se d'una aplicació que integra en un únic sistema tots els processos de negoci d'una empresa com la facturació, comptabilitat, inventari, recursos humans, etc... , i en el nostre cas no és així. Tot i això complim amb uns dels requeriments més importants d'un ERP: la existència d'una base de dades centralitzada. Així doncs, tot i que és molt habitual abusar del terme ERP, nosaltres no ho farem, i parlarem d'aplicació de gestió.

D'altra banda, és important comentar que la solució implementada permet el fàcil accés a les dades, mitjançant Web Services. D'aquesta manera altres mòduls podran accedir a la informació per tal d'oferir funcionalitats que completin la gestió. Així doncs, mitjançant la integració de diferents mòduls, cadascun dels quals

cobririen diferents necessitats de l'organització, podríem arribar a disposar d'un autèntic ERP, capaç de cobrir tots el processos de la organització.

Per tant, l'objectiu d'aquest projecte és:

- Dissenyar i implementar la base de dades en la que s'emmagatzemarà tota la informació de l'organització.
- Dissenyar i implementar l'aplicatiu que utilitza la base de dades per donar resposta a les necessitats dels usuaris.
- Plantejar i implementar l'aplicació de manera que aquesta pugui, en un futur, integrar-se amb més mòduls per tal de completar la solució.

En el que es refereix a limitacions, comentar que l'aplicació està pensada inicialment per petits tallers de vehicles, amb un únic equip de treball, però que volen tenir una eina que els permeti millorar la seva gestió, augmentant així la seva productivitat, gràcies a una bona organització de la feina i una disponibilitat total d'informació.

1.3. Estructura de la memòria

Tot i que pel títol de cada capítol, que trobem a l'índex de continguts de la memòria, és fàcil deduir quin serà el seu contingut, a continuació es descriuen breument:

- Estudi de viabilitat: En aquest capítol es pretén determinar de manera objectiva si el projecte pot continuar o si es pot optar per altres alternatives per resoldre la situació a tractar, pel que s'estudiarà en profunditat el sistema a realitzar així com els possibles riscos que podrien posar en perill la viabilitat del projecte.

- Tecnologia: Un cop hem establert la viabilitat del projecte, estudiarem quines han estat les tecnologies utilitzades per portar a terme el projecte així com detalls generals de l'entorn en que aquest s'ha desenvolupat, l'entorn web.

- Disseny Funcional: En aquest capítol detallarem la solució adoptada per donar resposta als requeriments, és a dir, el model de dades, les interfícies d'usuari i, en general, el funcionament global del sistema.

- Proves: Per considerar que l'aplicatiu funciona correctament, hem de realitzar proves el més exhaustives possible per tal de garantir l'absència d'errors, o com a mínim d'errors greus. Aquest capítol s'encarregarà de detallar com s'han dut a terme aquestes proves.

- Conclusions: En aquest capítol explicarem si s'han aconseguit els objectius definits inicialment així com les possibles ampliacions i millores.

2. Estudi de viabilitat

Aquest apartat pretén determinar de manera objectiva si el projecte pot continuar o si es pot optar per altres alternatives per resoldre el problema proposat. Analitzarem els objectius i requeriments del projecte, així com els factors que poden afectar a la viabilitat d'aquest.

2.1. Introducció a l'estudi de viabilitat

Tots els projectes són possibles, però si es tenen infinits recursos i temps. Això normalment no és així ja que quan es vol portar a terme un projecte ens trobem nombroses limitacions. Per tant és molt necessari fer un estudi de qualsevol projecte abans de començar el seu desenvolupament. En el nostres cas, vàrem realitzar aquest estudi en el moment que vam tenir clars els requeriments, i un cop establerta la viabilitat, vàrem començar amb la implementació.

Per establir la viabilitat d'un projecte hem d'avaluar una sèrie de factors, entre els que es troba la viabilitat econòmica (cost de desenvolupament del sistema, avaluat pels guanys que suposarà la seva implementació), viabilitat tècnica (funcionament, rendiment i d'altres restriccions que ens podrien afectar) i alternatives (una avaluació dels enfocaments alternatius al desenvolupament del sistema). Aquests factors seran tractats o no per l'anàlisi de viabilitat depenent de si la naturalesa del projecte ho requereix. En el nostre cas ens centrarem en la viabilitat tècnica.

2.2. Objecte

2.2.1. Descripció de la Situació a Tractar

Un taller de reparació de vehicles:

- És una petita organització, amb poca infrastructura informàtica i sense personal especialitzat.
- Està organitzat de manera que compta amb personal de recepció, responsable o cap de taller i operaris (o mecànics) encarregats de portar a terme els treballs necessaris als vehicles dels clients. Per tant, cada tipus d'usuari portarà a terme unes tasques diferents.
- El personal de recepció ha de poder gestionar els diferents clients així com els vehicles associats a aquests.

- El personal de recepció ha de poder enregistrar la recepció de vehicles i les intervencions que necessiten, per tal que els operaris, en cas de ser convenient, comencin els treballs corresponents.
- Els treballs que s'han de portar a terme als vehicles han de ser planificats (pel cap de taller), de manera que els operaris sàpiguen en tot moment quin treball han de realitzar.
- Els treballs s'han de realitzar en el temps previst, entregant-se el vehicle al client a la finalització d'aquests. Per tant, un treball té una duració prevista, així com una data d'inici i una altra de fi.
- Els treballs estaran formats per una o més de les feines predefinides, per les quals existeix un temps estàndard, al qual s'ha d'ajustar el que realment ha dedicat l'operari. Aquests temps estàndards podran ser comunicats als clients si ells ho sol·liciten.
- Els operaris han de poder imputar de manera senzilla totes les feines realitzades, així com els materials inclosos a cadascun dels treballs. A més hauran de poder indicar en tot moment quin treball estan duent a terme.
- El cap de taller, ha de poder veure de manera fàcil en quin estat es troben els treballs.
- El cap de taller, un cop s'han finalitzat els treballs necessaris a un vehicle, comprovarà si les previsions van ser correctes d'acord amb el temps dedicat.
- El detall de treballs realitzats a un vehicle, quedarà disponible des de l'aplicatiu, per futures consultes.

Aquest aplicatiu pretén cobrir totes aquestes necessitats bàsiques d'un taller mecànic a més d'altres que es detallaran posteriorment, i que faran més àgil la gestió del taller, ajudant a millorar la seva productivitat.

2.2.2. Objectius

Amb la realització d'aquest projecte pretenem proveir a un taller mecànic d'una aplicació que permeti millorar la seva gestió i, en conseqüència la seva productivitat. Per aconseguir això és necessari millorar-la a nivell individual, per cadascun del membres que formen part del personal, des de recepció fins als operaris, pel qual és molt important que disposin fàcilment de tota aquella informació que requereixen així com que puguin realitzar les seves tasques habituals d'una manera àgil. Així doncs, recepció haurà de poder accedir de

manera àgil a la informació dels clients i dels seus vehicles per enregistrar els treballs a realitzar, el cap de taller haurà, entre altres, de ser capaç de planificar els treballs a realitzar i de consultar la situació d'aquests, i els operaris no perdran temps preguntant quin treball han de realitzar després d'haver finalitzat un altre, sinó que disposaran immediatament d'aquesta informació. A més, en acabar els treballs, podran imputar fàcilment els materials emprats i temps dedicat, informació que a la seva vegada quedarà enregistrada per ser fàcilment consultada pels diferents usuaris de l'aplicatiu, és a dir, per recepció, cap de taller o fins i tot pels operaris.

És molt important que l'aplicació sigui fàcil d'utilitzar ja que d'altre manera, al tractar-se d'una eina d'ús diari, podríem aconseguir l'efecte contrari al nostre objectiu bàsic, que no és altre que millorar la productivitat. Per tant, pretenem una aplicació d'ús intuïtiu i fàcil aprenentatge, on les tasques més freqüents siguin àgils.

2.2.3. Perfil dels usuaris

L'aplicatiu que desenvolupa aquest projecte, tot i basar-se en un entorn web, està destinat a un ús intern dins d'una organització, en concret un taller mecànic, motiu pel qual no està pensat per a penjar-se a Internet. D'altra banda, qualsevol usuari no podrà accedir lliurement sinó que haurà de tenir un codi d'accés. A més, depenent del tipus d'usuari, aquests accediran a una o altre informació i podran portar a terme unes o altres operacions, les que li pertoquin segon els seu rol al taller.

En quant a coneixements informàtics, no caldrà ni molt menys ser un expert informàtic, sinó que amb unes nocions bàsiques d'utilització del navegador serà suficient per poder accedir a les funcionalitats que l'eina proporciona, evitant la necessitat d'una formació específica. A més, la interfície d'usuari es dissenyarà tenint en compte la usabilitat de la mateixa, de manera que a més de fàcil, la utilització resulti còmoda.

2.2.4. Fonts d'informació

Les fonts d'informació que s'utilitzaran per portar a terme el projecte i assolir els objectius definits, seran bàsicament aquelles referents al desenvolupament d'aplicacions web amb els llenguatges o tecnologies utilitzades, com són ASP.NET

(VB.NET), JavaScript, HTML i XML així com referents a altres disciplines que també seran necessàries com l'enginyeria del software o les bases de dades. Es consultaran llibres de cada matèria i apunts adquirits durant la carrera així com altres fonts disponibles a Internet, bàsicament fòrums i tutorials.
La documentació consultada serà indicada a l'apartat de bibliografia del present document.

2.3. Descripció del Sistema a Realitzar

2.3.1. Descripció dels Requeriments

Per realitzar una primera descripció dels requeriments funcionals, farem servir casos d'ús, una eina de fàcil comprensió del llenguatge UML[1]. Començarem per definir els actors del sistema, és a dir, els diferents rols: recepció, cap de taller i operaris. Per tant, per definir els diferents requeriments funcionals distingirem entre l'usuari que en farà ús d'aquests.

Un primer requeriment amb el que ens trobem és exigir la identificació amb un codi d'usuari i una paraula de pas. Això és necessari ja que cadascun d'aquests tipus d'usuari tindran un paper diferent a la organització, i per tant, les funcionalitats a les que podran accedir també seran diferents. Per tant, en funció del rol de l'usuari identificat, les pantalles de que disposaran seran diferents, i fins i tot, les dades que podran veure o modificar en aquestes seran diferents.

El primer tipus d'usuari, recepció, serà en la majoria de casos el primer contacte amb el client. Per tant, serà l'encarregat de portar a terme el registre d'aquest a l'aplicació, així com dels seus vehicles i dels treballs a realitzar en aquests. El perfil d'aquest usuari és el d'un administratiu amb nocions de mecànica, capaç, per tant, de donar una estimació de temps previst per portar a terme els treballs requerits (basant-se en els temps predefinits per cada operació a realitzar). A més, aquest usuari haurà de dur a terme altres tasques bàsiques com són:

- Administració d'usuaris de l'aplicació.
- Administració dels materials que poden ser empleats als treballs que es realitzen als vehicles.

[1] És el llenguatge de modelat de sistemes de software més conegut i utilitzat en l'actualitat. Es tracta d'un llenguatge gràfic per visualitzar, especificar, construir i documentar un sistema software.

- Administració de les tasques que poden constar en els treballs, així com dels seus temps predefinits.
- Administració de les marques i models de vehicles disponibles a l'aplicació.

En la següent figura veiem el detall dels diferents casos d'ús d'aquest actor, en notació UML.

Figura 1: Diagrama de casos d'ús de Recepció i Cap de taller

El segon tipus d'usuari és el que més funcionalitats pot portar a terme, el cap de taller. Aquest, a més de poder realitzar totes les tasques que pot realitzar recepció, pot planificar els treballs, de manera que els operaris puguin portar-los a terme. El perfil corresponent a aquest usuari és el d'un mecànic expert, amb suficient criteri com per saber amb una relativa exactitud, el temps requerit per portar a terme els treballs necessaris. A més, serà responsabilitat d'aquest, analitzar els temps dedicats a cadascuna de les tasques incloses als treballs, i dels treballs en el seu conjunt, i fer que aquests s'ajustin al màxim als temps predefinits per cadascuna de les tasques, que són en el qual basem les nostres previsions. De la mateixa manera, podrà fer ús de l'eina estadístiques, per calcular temps promitjos per determinats treballs o clients, per exemple. El detall dels casos d'ús corresponents a aquest usuari ho podem veure a l'anterior figura ja que a ella representem conjuntament els corresponents a recepció i al cap de taller.

L'últim tipus d'usuari amb el que ens trobem és l'operari o mecànic. Aquest serà l'encarregat de dur a terme els treballs definits per recepció o el cap de taller, seguint la planificació realitzada per aquest últim. A més, és el responsable d'indicar a quin treball està treballant així com les tasques realitzades (amb els temps dedicats) i els materials emprats. En aquest cas, i com és obvi, el perfil és el d'un mecànic capaç de realitzar qualsevol del treballs que el vehicle del client pugui requerir.
En la següent figura veiem el detall dels diferents casos d'ús disponibles per l'usuari mecànic, en notació UML.

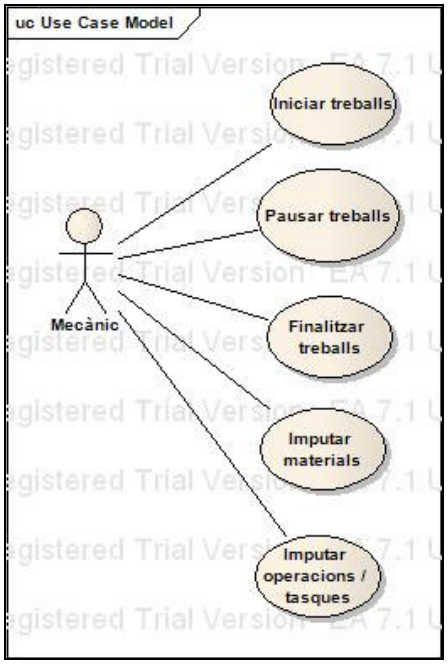


Figura 2: Diagrama de casos d'ús de Mecànic

Com veiem, la majoria de funcionalitats fan referència als treballs que es porten a terme als vehicles. En l'aplicatiu cada treball constituirà una Ordre de Reparació, abreujat OR, i contindran totes les operacions a realitzar al vehicle. Aquestes, en funció de la situació dels treballs, es trobaran en un o altre estat. Concretament els diferents estats en que pot estar una OR són:

1. RECEPCIONADA
   El vehicle ha estat recepcionat al taller, motiu que ha causat l'alta de la OR associada.
2. PLANIFICADA
   La OR ha estat planificada pel cap de taller, de manera que els operaris la puguin realitzar. El vehicle pot estar o no recepcionat.
3. EN CURSO
   La OR s'està realitzant. És a dir, totes les operacions que hi consten s'estan duent a terme pels operaris.
4. EN ESPERA
   La OR, tot i que en algun moment va ser iniciada, actualment no s'està duent a terme i està pendent de tornar-se a reprendre.
5. FINALIZADA
   Totes les tasques incloses a una OR han finalitzat.

A la figura 3 podem veure el diagrama d'estats d'una OR i hi podem observar:

- L'estat inicial d'una OR variarà en funció de si la crea recepció o el cap de taller.
- Una OR sols es pot iniciar si el vehicle està recepcionat.
- Qualsevol OR en estat RECEPCIONADA implica que el vehicle associat està recepcionat. Per tant, en el moment que es planifiqui es podrà iniciar, passant a l'estat EN CURSO.

A més dels requeriments funcionals comentats, comptem amb els següents requeriments no funcionals que també haurà de satisfer la solució: fàcil d'utilitzar, estable, segur i escalable. Respecte a aquest últim aspecte, escalabilitat, comentar que en el nostre cas es refereix tant al fet de suportar creixement en el volum de dades i usuaris com a les funcionalitats que s'ofereixen, havent-se de facilitar que altres mòduls utilitzin les dades que proporciona l'aplicatiu, per tal de completar la solució amb funcionalitats addicionals.

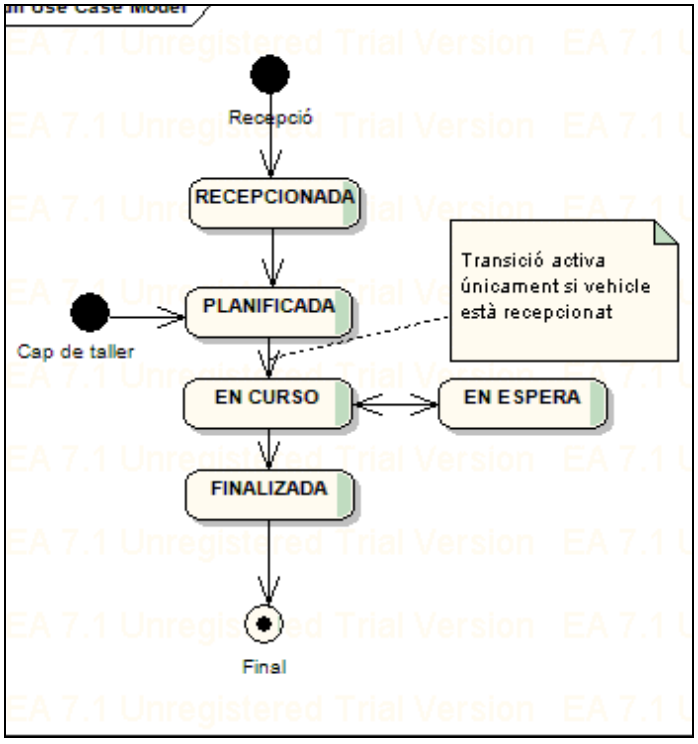


Figura 3: Diagrama d'estats d'una OR amb notació UML

2.3.2. Recursos

Els recursos necessaris per la realització del projecte podem dividir-los en dos tipus: Hardware i Software. A més, tindrem en compte tant la fase d'anàlisi i desenvolupament del projecte així com el moment en que el desenvolupament finalitzi i s'hagi d'explotar l'aplicació a un entorn de producció.

L'aplicació, tal i com ja vam comentar, es tracta d'una aplicació en entorn web, implementada amb tecnologia ASP.NET 2.0. Per tant, el servidor utilitzat serà de plataforma Windows, i comptarà amb el servidor d'aplicacions IIS (Internet Information Services), que constitueix l'entorn d'execució d'aplicacions ASP.NET, així com amb SQL Server 2005, que és el Repositori de dades que farem servir, escollit per la seva perfecta integració amb l'arquitectura ASP.NET.

Des del punt de vista de connectivitat, remarcar que al tractar-se d'una aplicació d'ús intern a la organització, no necessitem que el servidor tingui ni tan sols accés a Internet, pel qual pot tractar-se d'un PC únicament accessible per la nostra xarxa local.

És important comentar que tot i que aquestes necessitats seran comuns tant a la fase de desenvolupament com a la de producció, hi ha una petita diferència, concretament pel que fa a la distribució SQL Server 2005 emprada. Mentre que pel desenvolupament farem servir la versió Express (per tractar-se d'una distribució gratuïta) en producció convindria disposar d'altre tipus de llicència, evitant així la limitació a 4GB en l'espai ocupat per la Base de Dades.

En quant als recursos necessaris per tal que un usuari pugui fer ús de l'aplicació, només caldrà que disposi d'un PC amb un navegador d'Internet, concretament Internet Explorer, amb una versió 7.0 o posterior. Tot i semblar que això ens pot limitar molt els usuaris de la nostra aplicació, hem de recordar que es tracta de membres de la nostra companyia, pel que es tracta d'un entorn controlat, on fàcilment podem assegurar quin serà el navegador, cosa que no podríem assumir en una aplicació destinada al consum des d'Internet, on els terminals dels usuaris són molt més heterogenis.

D'altra banda, per poder portar a terme el desenvolupament, disposarem de l'entorn de desenvolupament per ASP.NET de Microsoft, Visual Studio .NET 2005,

així com de les eines que ens proveeix SQL Server 2005, bàsicament SQL Server Management Studio. A més, portarem a terme un control de versions del codi font que generem, utilitzant Microsoft Visual Source Safe. Per elaborar els prototipus de pantalles utilitzarem Fireworks 4.0 i per documentar tota la informació necessària MSWord. Per últim, utilitzarem Enterprise Architect trial version, de Sparx Systems, per realitzar els diagrames UML que consten a la documentació.

2.3.3. Avaluació de riscos

Hem de tenir en compte, quins problemes podrien afectar al rendiment i al correcte funcionament de l'aplicació. Bàsicament contemplarem els dos tipus de problemes més habituals amb els que es troba qualsevol aplicació web: els relacionats amb el temps de resposta i els relacionats amb les incompatibilitats entre navegadors.
Pel que fa al temps de resposta, normalment vindrà determinat per :

- Rendiment del servidor.
- Connexió del servidor amb Internet.
- La càrrega d'Internet.
- Connexió de l'usuari a Internet.
- Velocitat de l'ordinador i del navegador de l'usuari.

Com podem veure, la majoria d'aquests factors, els més determinants, estan relacionats amb Internet, pel que en el nostre cas no apliquen, al tractar-se d'una aplicació únicament accessible des de la pròpia empresa, sense accés des d'Internet. Altre factor que afecta negativament als temps de resposta d'aplicacions web, és el plantejament tradicional, és a dir, el fet d'enviar completament una pàgina des del client al servidor, i haver d'esperar una resposta per continuar. En aquest sentit, en el nostre cas, al treballar amb tecnologia AJAX on fem enviaments parcials de les pàgines (reduint per tant la quantitat de dades a enviar / rebre), i al fer crides asíncrones al servidor, tampoc haurà de ser un factor condicionant.

Per l'altre aspecte comentat, compatibilitat de navegadors, com ja hem avançat anteriorment, el fet de tractar-se d'una aplicació d'ús intern també evitarà que tingui incidència en la nostra aplicació doncs limitarem l'ús al navegador Internet Explorer 7.0 i posteriors, amb el que garantirem el correcte funcionament.

Tot i que sembla que amb el nostre plantejament d'aplicar tecnologies Web 2.0 tot són avantatges, especialment amb l'ús d'AJAX, aquest també implica algun risc. Concretament, per utilitzar AJAX a la nostra aplicació empraré el *framework*

ASP.NET AJAX, que és un dels més utilitzats doncs es tracta de la versió desenvolupada per Microsoft i per tant, a priori, la que millor s'integra amb aquesta tecnologia. Tot i això, es tracta d'un dels *frameworks* més nous, i per tant, menys testejat, pel que és possible que trobem algun tipus de *bug*[2]. En tal cas, hauríem d'anar cercant les possibles solucions a les fonts d'informació anteriorment esmentades.

2.3.4. Seguretat

La seguretat és un dels temes més importants del món web, així com també principal motiu pel qual molta gent no es decideix a endinsar-se en aquest món, degut a la desconfiança que encara els genera.

De totes maneres, tenint en compte que estem parlant d'una eina de gestió d'ús intern, els problemes de seguretat que puguin sorgir estan molt més acotats. Per accedir a l'aplicació, s'haurà d'estar en possessió d'un codi d'usuari i d'una paraula de pas per poder-se identificar. A més de no permetre l'accés sense codi, cadascun dels usuaris únicament podran accedir a la informació que li pertoqui així com realitzar les operacions autoritzades segons el seu rol a la organització.

Altre aspecte a tenir en compte és protegir la informació emmagatzemada a la base de dades, ja que cap persona no autoritzada hauria de poder accedir a aquestes dades utilitzant l'administrador del Gestor de Base de Dades. Això es podrà solucionar protegint l'accés a la informació de la base de dades amb un nom d'usuari i un password.

Per últim, en referència a la seguretat del codi font de l'aplicació, comentar que un cop finalitzat el desenvolupament, en la posta en producció, únicament requerim copiar els ensamblats (arxius .dll) al servidor on s'explotarà l'aplicació, i no el seu codi font. Per tant, serà decisió nostra si incloguem o no a l'entorn de producció el codi font, i permetem llavors la seva còpia o modificació, o pel contrari, preferim únicament lliurar els ensamblats, de manera que qualsevol modificació ens hagi de ser sol·licitada.

[2] Computer bug, defecte de software.

2.3.5. Organització del Projecte

El desenvolupament de software implica molts passos i activitats que s'han d'organitzar per tal d'establir com es portarà a terme la producció d'aquest, el qual es coneix com model de desenvolupament. Per entendre el procés de creació de software primer serà necessari entendre que és el que entenem per software. Normalment s'associa software amb programes. Això és incorrecte, el software no sols està format per programes sinó que a més està format per estructures de dades i per una documentació. Per tant, quan parlem de les activitats per desenvolupar software implicarà desenvolupar programes, estructures de dades i la documentació associada. Per explicar el model escollit farem un breu repàs dels models més importants:

Model seqüencial: És el model organitzatiu més simple i a la vegada un dels més utilitzats. Aquest model estableix una organització lineal en les diferents fases de desenvolupament, per tant no començarà una fase fins que no s'hagin acabat i certificat totes les anteriors. Aquest model comporta varis problemes: (1) els projectes reals no solen tenir un cicle de vida tan marcat com en el que es basa aquest model. Normalment en el desenvolupament hi ha tornades enrere. (2) dificultat de tenir clars des d'un principi tots els requeriments que aquest model imposa ja que molts d'aquests van sorgint durant el desenvolupament del mateix software. A més, aquest model impossibilita que el client pugui observar l'evolució del software produït ja que orienta tota la planificació cap a una única data de lliurament.

Prototipatge*:* És un model més realista ja que normalment un client no dóna un conjunt de requisits detallats, com indicava l'anterior model, sinó que dóna un conjunt d'objectius a complir. Mitjançant la construcció de prototipus que simulin el funcionament del software a desenvolupar el client coneixerà els requeriments que el software a desenvolupar haurà de satisfer. Aquest model també comporta varis problemes: (1) el cost, normalment es construeixen varis prototipus o el que seria el mateix, es desenvolupen parcialment varies vegades el software, i això, òbviament encareix molt el procés. (2) el client veu funcionar els prototipus i interpreta que és una versió definitiva, demanant que el software estigui acabat com abans millor.

Evolutiu*:* Concebeix la idea que el software pot ser desenvolupat per etapes, per increments, afegint en cada etapa una nova funcionalitat al sistema fins que aquest estigui acabat. Així, es parteix d'una llista de control que conté, en ordre, totes les tasques que ha d'incorporar el sistema. A cada pas traiem una nova tasca de la llista, dissenyant la seva implementació, codificant-la i provant-la. A continuació es realitza un anàlisi parcial del sistema obtingut i actualitzem la llista, podent afegir si cal nous requeriments. Seguidament afegirem al sistema la següent funcionalitat de la llista i així successivament. Aquest model bàsicament presenta dos problemes: (1) deixa d'existir la fase de manteniment per convertir-se tot el cicle de vida del software en el manteniment i (2) dificultat per marcar la fi d'inclusió de requeriments a la llista de control per part del client podent allargar molt el procés de desenvolupament del sistema.

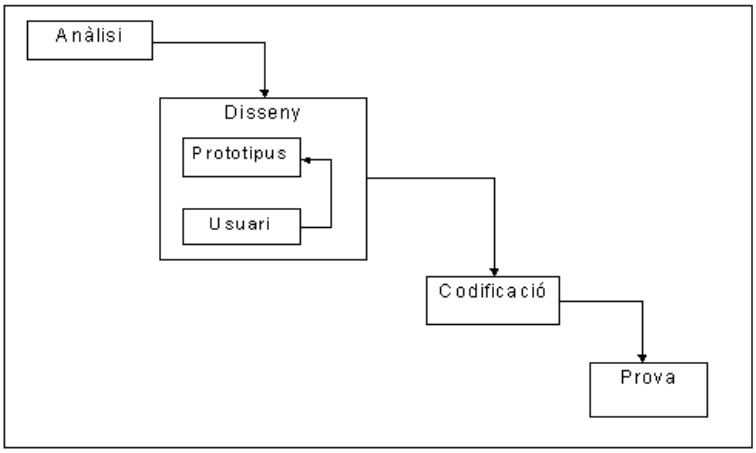
Espiral*:* Aquest és un metamodel que es pot acomodar a qualsevol dels anteriors models, aprofitant el millor de cadascun d'ells. És un model cíclic que es basa en l'elecció del millor model en funció del nivell de risc[3]. Aquest model pretén identificar i eliminar els problemes d'alt risc mitjançant un acurat procés de disseny, per evitar que es tractin de la mateixa manera els problemes més trivials i els més complexos.

Una vegada tenim clars aquests conceptes podem passar a comentar quin ha estat el model escollit per la realització del projecte i les raons. S'ha escollit el model seqüencial, bàsicament per tenir clars els requeriments des del començament, però amb una variació, dins de la fase de disseny, en el disseny de la interfície d'usuari, hem aplicat el *model de prototipatge*. Aplicarem aquest model exclusivament al disseny de la interfície d'usuari, per així modificar-la en funció de les opinions dels usuaris i dels resultats que aquests obtenen amb una o altra interfície. Per tant, mitjançant la construcció de prototipus, buscarem obtenir la interfície més ràpida i còmoda per l'usuari, la més usable.

El model que hem escollit ens marca l'ordre en què es realitzaran les diferents fases del projecte. Començarem per l'etapa d'anàlisi, posteriorment realitzarem el disseny (amb la peculiaritat abans descrita) i una vegada aquest ha estat completat

[3] Circumstàncies que poden perjudicar al procés de desenvolupament i a la qualitat del software, podent encarint el producte o afectar al correcte funcionament d'aquest.

passarem a l'etapa de codificació. Quan s'hagi acabat la codificació començarem la fase de prova per tal de verificar que l'aplicació compleix amb els objectius marcats inicialment. El gràfic següent il·lustra el procés.



2.4. Planificació del projecte

La planificació del projecte la faré fixant-me l'objectiu de presentar-lo en el proper mes de juny, i la dividiré en els quatre grans blocs de funcionalitats que he identificat, així com en la part de definició de funcionalitats, disseny de la base de dades, proves i, per últim, elaboració de la memòria i de la presentació.

La planificació detallarà el mes en que s'inicia i es finalitza cadascuna de les parts, ja que m'és molt difícil concretar a priori unes dates fixes – que seria el cas ideal – donat que he de compaginar la realització del projecte amb la meva feina, pel que la dedicació de tots els dies no serà la mateixa i no la puc preveure amb exactitud.

Per últim, comentar que a la planificació que s'inclou a continuació, es té en compte que tot i que algunes tasques es poden paral·lelitzar, això no serà sempre possible (podran coincidir en mes, però no en els dies). Concretament, fins que el funcional no estigui completament acabat, no es començarà el disseny de la base de dades, i fins que aquest disseny no es finalitzi, no es començarà el desenvolupament de cap mòdul, per portar a terme el projecte d'una manera ordenada i aconseguir així els millors resultats possibles.

Amb tot això, la planificació és la que consta al següent quadre:

Concepte	Mes inici	Mes fi
Definició funcional	Novembre	Desembre
Disseny base de dades	Desembre	Desembre
Bloc de gestió de clients	Gener	Gener
Bloc de gestió de vehicles	Gener	Gener
Bloc de gestió d'ordres de treball	Febrer	Febrer
Bloc de planificació	Març	Abril
Proves	Abril	Maig
Memòria	Maig	Juny
Presentació	Maig	Juny

Com podem veure, un dels blocs que hem planificat és el de proves, però cal dir, que els diferents blocs de desenvolupament inclouen també la realització de proves, d'un àmbit més específic. En un capítol posterior es detallaran àmpliament les proves realitzades.

Per últim, destacar que, com la resta del present estudi de viabilitat, aquesta planificació es va realitzar abans del començament del projecte, i donat que no estava previst el desenvolupament del bloc d'administració aquest no hi consta.
Val a dir, per tant, que la realització del projecte ha millorat les previsions en quant a planificació, i per tant, s'ha considerat oportú desenvolupar aquest mòdul addicional, en un intent per oferir una aplicació el més completa possible.

2.5. Conclusions

Donades les característiques de l'aplicació a desenvolupar podem deduir el seu grau de complexitat ja que implica un àmbit multidisciplinar que va des dels coneixements en programació (VB.NET, JavaScript,...), enginyeria de la usabilitat, disseny gràfic, disseny de bases de dades fins als coneixements en servidors. És per això que el desenvolupament d'aplicacions web sofisticades és més efectiu amb l'ajuda d'especialistes en disciplines concretes.

Moltes de les matèries o disciplines amb les quals treballarà el projecte han estat tractades durant el transcurs de la carrera, bàsicament bases de dades, sistemes

experts, algorismes i programació o xarxes, i per tant el projecte ens servirà per posar-ho en pràctica. A més, també s'ha tractat amb especialitats que no es cursen a la carrera pel que, dintre de les nostres possibilitats, ens haurem de formar en aquestes, emprant les fonts comentades.

Aquest desconeixement previ de les matèries que implica la realització del projecte és un factor que podria condicionar negativament la previsió de finalització del mateix, que és d'un curs acadèmic. Tot i això, penso que les matèries que implica el projecte són molt interessants i de total actualitat, motiu que fa que aquest projecte em resulti molt atractiu.

Cal comentar que l'àmbit de desenvolupament d'aplicacions web és molt dinàmic, pel que és freqüent l'aparició de noves tecnologies pel qual tota persona que es vulgui dedicar al desenvolupament d'aquestes no es pot limitar a uns coneixements adquirits sobre una tecnologia concreta, sinó que ha d'autoformar-se en les que van sorgint. Així, mitjançant aquest projecte podrem desenvolupar la maduresa adquirida durant la carrera per ser capaç d'assimilar nous coneixements, concretament per desenvolupar aplicacions web.

Així doncs, per tots els arguments anteriorment exposats, crec que el projecte compleix tots els requisits necessaris per satisfer correctament el contingut d'un projecte de final de carrera.

3. Tecnologia

En aquest capítol es presenten alguns dels conceptes bàsics de l'entorn web, en el qual es basa el projecte, així com les diverses tecnologies utilitzades.

3.1. Introducció a les tecnologies web

Abans d'entrar en el detall de les diferents tecnologies que s'han emprat en la realització d'aquest projecte, cal tenir present que aquest s'ha implementat en entorn web, pel que comentarem alguns aspectes bàsics d'aquest, per facilitar la comprensió del detall de cadascuna de les tecnologies que posteriorment seran explicades.

Primer de tot hem de saber que hi han nombroses diferències entre una aplicació tradicional i les aplicacions web. Les aplicacions que s'executen en un entorn visual sovint interactuen amb finestres, com per exemple aplicacions creades amb Visual Basic i executades en un entorn Windows. En canvi, quan parlem d'un entorn web, hem de parlar de comunicació entre ordinadors i aquesta ha d'estar basada en protocols, pel que no podem posar un executable al servidor i que els clients vegin les finestres de l'aplicació des del seu ordinador connectat a Internet.

HTTP és el protocol de la web i aquest implica que la sortida del nostre "executable" no han de ser pantalles de Windows, com al cas anterior, sinó codi HTML (HyperText Markup Language). La raó és bastant evident, hem de minimitzar el trànsit de dades per Internet ja que la transmissió per la xarxa és molt més lenta que a un mateix ordinador.

Les aplicacions per la web, emprant el World Wide Web son aplicacions basades en una estructura client-servidor que és en la que es basa la programació per xarxes TCP/IP. Així, tenim dos parts, el client (el navegador), el servidor (servidor web) i un protocol pel que es comuniquen el HTTP.

La part del client de les aplicacions web està formada per codi HTML que forma la pàgina web, i en alguns casos amb codi executable mitjançant els llenguatges script dels navegadors (bàsicament JavaScript) o mitjançant petits programes (applets) de Java.

La part del servidor està formada per un programa o script que és executat pel servidor web generant codi HTML que serà enviat al navegador (client). Tradicionalment a aquest programa o script que és executat al servidor web se l'ha anomenat CGI (Common Gateway Interface). Posteriorment veurem el funcionament concret pel cas d'ASP.NET.

3.2. Llenguatges de Programació

3.2.1. HTML (HyperText Mark-up Language)

Com hem comentat anteriorment, una pàgina web es visualitza des del nostre navegador, i està constituïda per diversos recursos com imatges, vídeos, etc... i el que farà que tot això es visualitzi correctament, és el seu el codi font, és a dir, HTML.

L'HTML es va crear en un principi amb objectius divulgatius, per l'intercanvi de documents científics i tècnics, pensat pel seu ús per especialistes, tot i que aquest objectiu es va veure ràpidament sobrepassat, havent-se de crear inclús nous elements que completessin les necessitats que van anar sorgint, i que de vegades han suposat l'aparició de problemes de compatibilitat entre diferents plataformes.

HTML indicarà l'estructura i contingut d'una pàgina web i, fins a cert punt el format amb que aquest es visualitza, la seva aparença al navegador. Es composa d'etiquetes, que tenen la forma <B> o <U>, on cada una significa alguna cosa. Per exemple, <B> significa que s'escriu en negreta, <U> significa que s'escriu subratllat, <A> és un enllaç, … Gairebé totes les etiquetes tenen la seva corresponent etiqueta de tancament, que indica que a partir d'aquell punt no deu d'afectar la etiqueta. L'etiqueta de tancament té el mateix format que la d'obertura, i només canvia en què conté el caràcter "/". Així, </B> s'utilitza per indicar que s'ha deixat d'escriure en negreta, </A> per indicar que s'ha acabat el text o la imatge que fa d'enllaç, …

D'aquesta manera, veiem que l'HTML no és més que una sèrie d'etiquetes que s'utilitzen per definir la forma o l'estil que volem aplicar al nostre document. A la següent taula es poden veure uns quants exemples d'etiquetes de codi HTML amb la seva corresponent visualització al navegador.

codi HTML	visualització al navegador
<B>Codi HTML</B>	**Codi HTML**
<I>Codi HTML</I>	*Codi HTML*
<U>Codi HTML</U>	Codi HTML

Aquestes etiquetes formaran el document HTML, que estarà delimitat per les etiquetes < HTML > y </ HTML >. Dins d'aquest document podem distingir també dos parts principals ben diferenciades:

1. la capçalera, delimitada per <HEAD> i </HEAD> on es col·loquen etiquetes de tipus informatiu com per exemple el títol de la pàgina, paraules clau, etc...
2. el cos de la pàgina, flanquejat per les etiquetes <BODY> i </BODY>, que serà on col·locarem el nostre text i imatges delimitats a la vegada per altres etiquetes com les que ja s'han descrit.

L'estructura anteriorment comentada queda reflectida a l'exemple que mostrem a continuació, així com la seva visualització en un navegador:

```
<html>

 <head>
  <title>document exemple</title>
 </head>

 <body>
   <B>Codi HTML</B> <BR>
   <I>Codi HTML</I> <BR>
   <U>Codi HTML</U> <BR>
 </body>

</html>
```

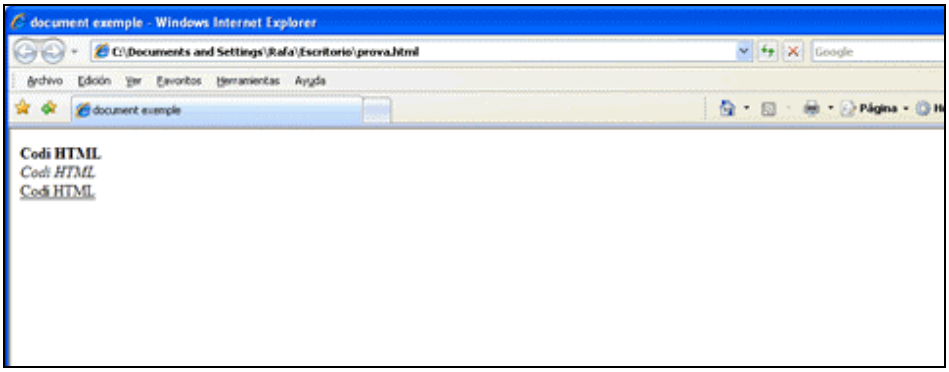


Figura 4: Visualització d'HTML al navegador

En la realització del projecte he utilitzat una evolució d'HTML, el que s'anomena XHTML (eXtensible HyperText Mark-up Language) o com comunment se li coneix, HTML 4, el qual es diferencia de versions anteriors en que compleix les normes sintàctiques d'XML. Quan parlem d'XML detallarem a què ens referim amb això.

3.2.2. XML (eXtensible Mark-up Language)

XML consisteix en un llenguatge de marques, més concretament un metallenguatge, és a dir, partint d'XML podem crear nous llenguatge, com HTML o cXML. El seu principal objectiu és permetre compartir les dades amb les que es treballa a tots els nivells, i per totes les aplicacions i suports al no estar lligat a cap arquitectura ni sistema operatiu concret. D'aquesta manera juga un paper importantíssim en el món actual, que tendeix a la globalització i la compatibilitat entre els sistemes, ja que és la tecnologia que permetrà compartir la informació d'una manera segura, fàcil i fiable.

En el cas particular del nostre projecte, la importància d'XML és encara més marcada ja que la tecnologia ASP.NET en la qual s'implementa forma part de la plataforma .NET Framework, i aquesta es basa completament en XML per portar a terme l'enviament i accés a dades així com per treballar amb Serveis Web, concepte que serà descrit posteriorment.

Algunes de les característiques més interessants d'XML són:

- No treballa amb un conjunt de marques fixes sinó, que a diferència d'HTML, proporciona completa llibertat per definir-les.
- És completament portable i independent de plataforma.

- Contempla que els documents continguin instruccions destinades als programes informàtics que els processin.
- Permet estructurar i descriure dades per tal que puguin ser interpretades per qualsevol aplicació en qualsevol sistema .
- És un llenguatge formal, des del punt de vista de les dades i la forma de guardar-les.
- És extensible, de forma que es pot utilitzar a tots els camps del coneixement.
- És fàcil de llegir, editar, implantar, programar i aplicar als diferents sistemes.
- És de lliure ús, no és propietat de cap empresa, companyia o organització.

Existeix una llista de casos immensa on aplicar XML, i alguns d'aquests casos pràctics són:

- Comunicacions de dades
- Migracions de dades
- Aplicacions web

Un document XML es pot definir a partir dels següents conceptes:

- ***Elements:*** Serveixen per marcar les unitats estructurals bàsiques d'informació, i poden contenir altres elements.
- ***Atributs:*** Parells *nomAtribut-valor* que s'utilitzen per marcar informació descriptiva de l'ocurrència d'un element.
- ***Entitats:*** Serveixen per a proporcionar una etiqueta amb la que anomenar a una part del document.
- ***Notacions:*** Per aportar informació a les aplicacions (no parsers XML) sobre les dades.
- ***Comentaris***
- ***Instruccions de Processament:*** Proporcionen informació addicional a les aplicacions que processen el document .
- ***Declaració XML***

Molt relacionat amb XML trobem el DTD (Document Type Definition), el qual conté el conjunt de definicions dels elements que podem trobar en un document XML, així com l'estructura d'aquest. La utilització d'un DTD en XML és opcional, però segons ho incloguem o no distingirem entre:

- Documents XML ben formats: compleix amb les especificacions del llenguatge XML respecte a les regles sintàctiques. Per exemple, ha d'existir un únic node arrel, totes les seves marques han de tenir el corresponent tancament, etc...
- Documents XML validats: documents que a més d'estar ben formats, contenen una estructura i semàntica definida a un DTD.

A l'apartat anterior vàrem parlar d'XHTML, i vaig comentar que a diferència d'HTML, aquest complia les regles sintàctiques d'XML. Doncs bé, ens referíem a que han d'estar ben formats, i per tant, hauran de comptar amb:

- Etiquetes en minúscules
- Etiquetes ben tancades
- Correcta anidació d'elements
- Valors d'atributs entre cometes
- Ús de l'atribut id

I a més, estaran validats mitjançant el corresponent DTD.

Per acabar, mostraré un senzill exemple de document XML, utilitzat per transmetre dades referents a models de vehicles:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<marcas>
        <marca id="1" descripcion="Seat">
                <modelos>
                        <modelo id="1" descripcion="Ibiza"/>
                        <modelo id="2" descripcion="Cordoba"/>
                        <modelo id="3" descripcion="Toledo"/>
                        <modelo id="4" descripcion="Alhambra"/>
                        <modelo id="5" descripcion="Leon"/>
                        <modelo id="6" descripcion="Altea"/>
                </modelos>
        </marca>
</marcas>
```

3.2.3. JavaScript

JavaScript és un llenguatge de programació utilitzat per crear petits programes encarregats de realitzar certes accions dintre de l'àmbit d'una pàgina web.

Entenent com a client la màquina des d'on es visualitza una pàgina web, es tracta d'un llenguatge de programació del costat del client, ja que és el navegador el que

suporta la càrrega de processament. Gràcies a la seva compatibilitat amb la majoria de navegadors moderns, és el llenguatge de programació del costat del client més utilitzat.

Amb JavaScript podem crear efectes especials a les pàgines, així com definir interactivitats amb l'usuari. El navegador del client és l'encarregat d'interpretar les instruccions JavaScript i executar-les per realitzar aquests efectes i interactivitats, de manera que el major recurs, i potser l'únic, amb el que compta aquest llenguatge és el propi navegador.

JavaScript és el següent pas, després de l'HTML, que pot fer un programador que decideixi millorar les seves pàgines i la potència dels seus projectes. És un llenguatge de programació bastant senzill i pensat per fer les coses amb gran rapidesa. Fins i tot persones que no tinguin experiència prèvia en programació poden aprendre aquest llenguatge amb facilitat i utilitzar-lo amb tota la seva potència amb només una mica de pràctica.

Entre les accions típiques que es poden realitzar amb JavaScript tenim dos vessants. Per una banda els **efectes especials** sobre pàgines web, per crear continguts dinàmics i elements que tinguin moviment, que canviïn de color, o qualsevol altre dinamisme. Per altra banda, JavaScript permet executar instruccions com resposta a les accions de l'usuari, amb el que es poden crear **pàgines interactives** amb programes com calculadores, agendes, o taules de càlcul. A més, JavaScript permet realitzar certes accions en funció de l'event que tingui lloc en una pàgina, entenent per event qualsevol acció que es faci o que faci la pàgina. Diferents events podrien ser redimensionar la pàgina, carregar-la, fer un click amb el ratolí, modificar un camp, …

El codi JavaScript s'inclou en un fitxer HTML entre les etiquetes <SCRIPT language="JavaScript"> i </SCRIPT>.

JavaScript és doncs un llenguatge amb moltes possibilitats, permet la programació de petits scripts, però també de programes més grans, orientats a objectes, amb funcions, estructures de dades complexes,… A més, aquest llenguatge posa a disposició del programador tots els elements que formen la pàgina web, per a que aquest pugui modificar-los dinàmicament.

Resumint amb poques paraules, amb JavaScript el programador es converteix en el veritable controlador de cada event que té lloc a la pàgina quan l'està visualitzant el client al navegador.

En aquest projecte l'aplicació del JavaScript s'ha centrat en la validació del contingut dels camps dels formularis, en mostrar diàlegs amb missatges d'error o d'avís, en realitzar les crides al servidor per a obtenir les dades que es volen mostrar als usuaris, i finalment en transformar les dades obtingudes per a mostrar una interfície comprensible pels usuaris.

3.2.4. DHTML

DHTML significa "HTML Dinàmic", però en realitat és senzillament l'ús conjunt de JavaScript i fulls d'estil. És fàcil suposar quelcom més complicat pel seu nom, ja que sovint, dinàmic s'utilitza per referir-se a pàgines generades sobre la marxa, però en aquest cas, el terme dinàmic de DHTML es refereix a que es mou.
Tot i això, DHTML és molt poderós ja que ens permet crear efectes que fins ara eren impossibles amb HTML i JavaScript.
A més, els fulls d'estil (CSS – Cascading Style Sheets) ens permetran posicionar elements en la pantalla exactament on volem, sense barallar-nos amb taules HTML.

DHTML ens permetrà doncs crear pàgines i aplicacions web que funcionin més com aplicacions de software que com pàgines web normals i corrents, però hem de tenir en compte certs aspectes que no podrà realitzar DHTML, com és accedir o controlar la màquina de l'usuari. Per raons de seguretat, escriure a la màquina de l'usuari està molt limitat. Podem emmagatzemar dades a la màquina de l'usuari però sols mitjançant una cookie, i fins i tot en aquest cas estarem molt limitats a arxius de text molt petits. D'aquesta manera es protegeix als usuaris d'aquells scripts que poden causar danys a la seva màquina o que permetin a programadors amb pocs escrúpols accedir a informació personal.

En aquest projecte l'aplicació de DHTML s'ha centrat en posicionar de manera precisa alguns elements de disseny a la pantalla així com en l'aplicació de fulls d'estil, de manera que resulti més senzill aconseguir un estil homogeni. D'aquesta manera, a més, resultarà molt més fàcil qualsevol modificació d'estil, ja que tenim localitzada la seva definició als fitxers .css, pel que modificant aquests, es podria donar una aparença completament diferent a l'aplicació.

3.2.5. ASP.NET

Com ja vaig comentar en capítols precedents, el projecte s'ha desenvolupat en ASP.NET, concretament en ASP.NET 2.0, el qual és una millora d'ASP.NET (ASP.NET 1.0). El motiu d'escollir aquesta tecnologia és, entre altres, el coneixement previ que tinc d'ASP. Tot i això, entre aquestes dues tecnologies existeixen diverses diferències, i algunes d'aquestes força importants, com he anat descobriment a mesura que més coneixia aquesta nova arquitectura.

ASP i ASP.NET fan referència a tecnologies que es fan servir al costat del servidor per portar a terme aplicacions web dinàmiques i que per tant, mostren informació que variarà amb el temps. Així doncs, parlem d'aplicacions que s'adapten a la informació que introdueix l'usuari, a la informació recuperada de la base de dades i a les condicions del moment en el qual s'executa l'aplicació.

La diferència més important entre aquestes dues tecnologies és que ASP consisteix en un llenguatge interpretat en el servidor, mentre que ASP.NET és un llenguatge que es compila en el servidor abans de retornar informació al client. Tot i que podria semblar que el fet de compilar l'aplicació i retornar la informació al client farà que s'incrementi el temps d'espera, això no és així, ja que el procés de compilació de .NET és realment fiable i sorprenentment ràpid, gràcies a una compilació que es realitza sota demanda.
Per tant, mentre que en ASP o altres llenguatges com PHP, cada línia de codi és analitzada i retornada en format HTML al client (per cada petició), en ASP.NET s'ha de compilar la pàgina ASP.NET al servidor, per després, retornar el resultat en HTML al client. És important remarcar que aquesta compilació no es realitzarà per cada petició, sinó només quan resulti convenient.

Per acabar d'entendre les diferències entre un i altre plantejament, mostrarem els avantatges i inconvenients de cadascun:

Llenguatge interpretat

Avantatges	Inconvenients
El codi és còmode per depurar ja que no és necessari tornar a compilar després d'un canvi.	L'execució es relentitza al ser necessària la interpretació línia a línia cada cop.
No és necessari disposar d'un compilador ja que l'intèrpret (que forma part del navegador) executa l'script.	El codi és visible i pot ser objecte de plagi per part d'altres persones.
El manteniment és fàcil i ràpid, per part de l'autor o	L'usuari té accés al codi i pot modificar-lo, fent-lo

altre programador.	malbé.

Llenguatge compilat

Avantatges	Inconvenients
El codi compilat s'executa molt més ràpid, al no requerir-se cada cop d'una traducció.	És necessari disposar d'un compilador-linkador per el procés de compilació.
El codi compilat no pot ser "obert" per altres persones. No és necessari transmetre el codi font.	El codi compilat normalment ocupa bastant més espai a disc, ja que incorpora en el propi codi algunes llibreries del sistema.
El codi compilat pot estar, íntegrament, inclòs a un sol fitxer.	Depurar un programa implica tornar-lo a compilar després dels canvis.

Tot i que les diferències són importants, podria semblar que ASP.NET és simplement una nova versió d'ASP, però això no és així. ASP s'ha hagut de reconstruir completament, formant ara part d'un entorn de programació orientat a objectes, i dintre d'aquest, formant part d'un entorn compilat. De fet no sols es va reconstruir ASP sinó que en un intent per unificar el món del desenvolupament, Microsoft va dissenyar un nou model, del qual ASP.NET forma part, és el que anomenem .NET Framework i permetrà no sols el desenvolupament d'aplicacions ASP.NET que ara ens ocupa, sinó també d'aplicacions per a Windows. Gràficament podem veure aquest plantejament en la següent figura:

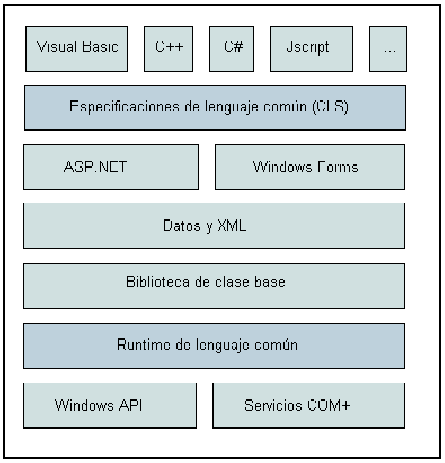


Figura 5: Capes de l'aquitectura .NET

Alguns dels aspectes més importants que s'hi observa en aquesta jerarquia de capes:

– El nucli principal de .NET Framework es situa a "Runtime de lenguaje común" o el que es coneix com CLR (Common Language Runtime). És el responsable d'executar el codi i s'encarrega de dir a Windows el que ha de fer amb les

instruccions proporcionades així com d'altres tasques com la gestió de memòria.

- El nivell "biblioteca de clase base" posarà a la nostra disposició una gran quantitat d'objectes que podrem utilitzar a les nostres aplicacions.
- Al nivell "datos y XML" es fa referència a la capa d'accés a dades, tant referint-nos a classes que ens permetran treballar amb bases de dades com amb XML.
- Al primer dels nivells trobem els diferents llenguatges que posa a la nostra disposició aquest model de desenvolupament, de manera que podrem escollir aquells amb el que més còmodes ens trobem. Entre altres trobem Visual Basic.NET (és l'escollit en el nostre cas), així com C#, etc...
- La capa CLS s'encarrega de traduir el llenguatge escollit en un conjunt d'instruccions iguals per a .NET.

Altres aspectes característics d'ASP.NET són:

- Existència de controls de servidor que podem fer servir a les nostres aplicacions.
- Permet crear i consumir Serveis Web.
- Treballa de manera directa amb XML.
- Millora de la seguretat i escalabilitat.
- Permet fàcil separació entre presentació i contingut mitjançant el model *code-behind*. Les pàgines d'ASP.NET, conegudes com formularis web (web forms), són el principal medi per la construcció de les aplicacions, però amb el model code-behind, es subdivideixen en dos parts: arxiu .aspx (contindrà els aspectes de presentació) i arxiu .aspx.vb (contindrà els aspectes de contingut).
- Facilita la creació de components reutilitzables mitjançant el que es coneix com Controls d'Usuari (User Controls).

Per acabar, mostrarem un breu exemple d'una pàgina d'ASP.NET.

```
<script runat="server" language="vb">
        Sub Page_Load(Source :Object, e :EventArgs)
                Identificador.href=http://www.enlaceweb.org
                Identificador.target = "_blank"
        End Sub
</script>

<form runat="server">
        <font face="verdana" size="2">
                <a runat="server" id="identificador">Enlace</a>
        </font>
</form>
```

i visualitzat al navegador quedaria com es mostra a la següent imatge

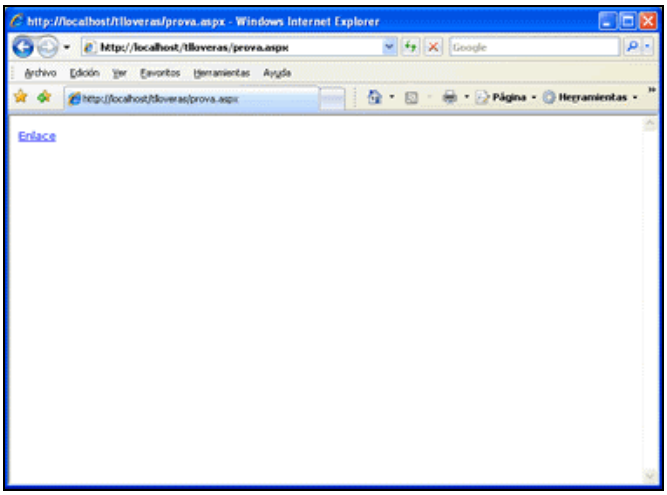


Figura 6: Visualització de pàgina ASP.NET en navegador

3.3. AJAX (Aynchronous JavaScript And XML)

AJAX és una tècnica de desenvolupament web, que consisteix en l'ús conjunt d'un seguit de tecnologies ja existents:

- XHTML o (HTML) i CSS.
- Document Object Model(DOM) al qual accedirem mitjançant JavaScript, per tal de mostrar i interactuar dinàmicament amb la informació.
- L'objecte XMLHttpRequest per intercanviar dades de manera asíncrona amb el servidor.
- XML, com a format més comú per la transferència de tornada.

Mitjançant AJAX podem crear aplicacions interactives RIA (Rich Internet Applications) que milloren l'experiència de l'usuari. Aquestes s'executen en el client, és a dir, al navegador web dels usuaris i manté la comunicació asíncrona amb el servidor en un segon pla. D'aquesta manera és possible realitzar canvis sobre la mateixa pàgina sense necessitat de recarregar-la, el que significa major interactivitat i velocitat.

Per entendre millor el funcionament d'AJAX recordarem com funcionaven les aplicacions web 'tradicionals':

– Un client fa una petició HTTP al servidor web (normalment mitjançant un navegador web).

- El servidor web processa la petició dinàmicament, mitjançant un llenguatge com ara ASP.NET.
- Un cop processada la petició HTTP pel servidor, aquest retorna una resposta HTML al client (pàgina web).
- Mitjançant aquesta pàgina generalment el cicle torna a començar ja que el client podrà tornar a fer peticions, etc...

En canvi, amb AJAX quan un client fa una petició HTTP al servidor, la fa mitjançant JavaScript. El servidor processa la petició i en comptes de retornar al client una pàgina HTML, li retorna un resultat en XML, que és processat per JavaScript per actualitzar únicament les seccions de la pàgina necessàries (sense haver de carregar una nova pàgina).
Gràficament podem veure aquest nou plantejament a la següent figura:

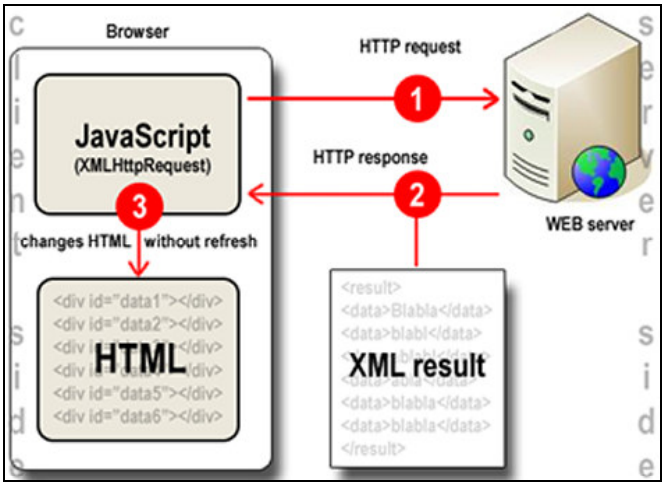


Figura 7: Funcionament d'AJAX

Com tot, AJAX no és perfecte, pel que comptarà amb avantatges i inconvenients. Concretament:

Avantatges

- Les pàgines no es recarreguen constantment.
- El temps d'espera és menor.
- Es poden aconseguir coses que sense AJAX no serien possibles, com el conegut Gmail, per exemple.

Inconvenients

- Falta d'integració amb el botó 'enrere' dels navegadors. Això és degut a que sempre ens trobem a la mateixa pàgina, i això pot arribar a confondre a l'usuari.
- És necessari que el navegador suporti i tingui habilitat Javascript. No és un gran inconvenient ja que pràcticament tots els navegadors moderns suporten Javascript.
- Al haver-se d'executar més codi al costat del client, pot relentitzar-se el rendiment de la màquina del client. Per això s'ha d'utilitzar AJAX amb moderació.
- Al no recarregar les pàgines i sempre estar a la mateixa, no es té una URL a la que poder-se referir, en cas de, per exemple, voler recomanar una pàgina o tornar-hi.

Actualment hi ha multitud de llibreries que faciliten l'ús d'AJAX i la seva integració a les aplicacions. En el nostre cas, hem decidit utilitzar *ASP.NET AJAX*, tal i com ja vam introduir en el capítol anterior, degut a la seva perfecta integració amb ASP.NET, i l'utilitzarem de manera habitual per fer més útil i agradable a l'usuari l'ús de l'aplicatiu.

Per acabar, mostraré un exemple on es veurà la facilitat amb la que podem utilitzar AJAX al nostre aplicatiu, per cridar funcionalitats del costat del servidor des de codi JavaScript. Concretament, a l'exemple s'observa com es crida a un Web Service (svcGestion), al seu mètode recuperarInfoMaterial, facilitant els paràmetres que calguin, a més dels noms de les funcions que s'han de cridar en cas que l'execució del Web Service tingui èxit o no, respectivament. Aquesta simplicitat és possible gràcies a que la llibreria *ASP.NET AJAX* s'encarrega dels detalls interns, extenent JavaScript amb noves classes i objectes que nosaltres podrem utilitzar per cridar els elements del servidor, els Web Services. I tot això, simplement afegint algunes línies de codi addicionals al costat del servidor, per registrar el Web Service i indicar que s'han de generar les classes i objectes que calguin.

```
function muestraDetalleMaterial(idMaterial)
{
   var modalPopupBehavior = $find('programmaticModalPopupBehaviorMat');
   document.getElementById(varIdMaterialMO).value = idMaterial;
   modalPopupBehavior.show();
    // crida al WS
   svcGestion.recuperarInfoMaterial(idMaterial, SucceededCallbackMaterial,     FailedCallback);
}
```

3.4. Serveis Web (Web Services)

Un Servei Web és un component software que es comunica amb altres components, mitjançant missatges codificats en XML, utilitzant protocols de transport estàndard d'Internet, com SMTP o HTTP, capaços de travessar perfectament els firewalls. De manera intuïtiva, un Servei Web és una 'caixa

negra', sense interfaç d'usuari, que pot ser reutilitzat per donar servei a programes en comptes de a persones. Cal a dir, que constitueix la solució amb més futur per la integració d'aplicacions a Internet, així com una bona alternativa per a integrar petites aplicacions en una intranet.

La idea bàsica de funcionament dels Web Services consisteix en enviar una petició via HTTP a un servei ubicat a una URL determinada. Aquest servei rep la petició, la processa i retorna una resposta a l'origen també mitjançant el protocol HTTP.

Des del punt de vista tècnic, els Serveis Web necessiten de diverses especificacions i tecnologies: formes estàndards de representar dades, formats de missatges extensibles i comuns, llenguatges extensibles de descripció de serveis i mecanismes de descobriment de Serveis Web o de proveïdors d'aquests. D'aquí la necessitat de recolzar-se en els 4 principals estàndards: XML, SOAP, UDDI i WSDL.

- SOAP **(**Simple Object Access Protocol): No és altre cosa que un format que ens permet estructurar les dades XML. Dit d'altre manera, SOAP és una manera de presentar les dades XML, el que vindria a ser un Schema de XML. Podríem accedir a Serveis Web sense l'ús de SOAP, mitjançant HTTP POST o HTTP GET, però no podríem passar-li tipus d'objectes complexes (classes, estructures o DataSets).

- UDDI **(**Universal Discovery , Description and Integration): Ens proveeix d'una mena de buscador per localitzar Serveis Web que podran ser utilitzats / consumits des de les nostres aplicacions.

- WSDL (Web Service Description Language): Consisteix en un format d'intercanvi d'informació, entre el servei i consumidor, els quals han de tractar les dades rebudes i enviades per un i altre. Per entendre's, han de parlar el mateix 'idioma', és a dir, el mateix WSDL, ja que es tracta d'un format d'intercanvi d'informació. WSDL és, com no, un document XML que descriu els protocols o serveis de transport, la semàntica de les crides o peticions i les crides en si.

  En resum, SOAP és un protocol d'intercanvi de dades i s'encarrega d'esquematitzar la informació d'intercanvi, WSDL s'encarrega de descriure com ha de fer-se l'intercanvi d'informació i UDDI s'encarrega de descobrir els serveis web

que hi ha a la xarxa, ja sigui aquesta global (Internet) o bé una xarxa interempresarial (Extranet) o bé una xarxa privada (Intranet).

Per finalitzar aquesta introducció, cal dir que els Serveis Web poden tenir moltes i molt variades funcionalitats ja que el que permeten és accedir a través d'un protocol HTTP a una determinada funcionalitat d'un sistema. D'aquesta manera les possibilitats d'utilització són molt amplies. En el nostre cas els hem utilitzat, per exemple, per recuperar informació de la base de dades, per incorporar-ne, per modificar-la, etc...i tot això, aprofitant les bondats d'AJAX, de manera asíncrona, el qual ens ha permès millorar molt l'agilitat de l'aplicació.

3.5. Servidor d'Aplicacions

Com hem comentat en un capítol anterior, el servidor d'aplicacions utilitzat per poder executar aplicatius ASP.NET és IIS (Internet Information Services). És el mateix servidor que ja s'utilitzava per ASP, però millorat, i el podem trobar a qualsevol dels sistemes operatius Windows, tot i que amb versions diferents.
Es tracta d'un servidor ideal per una petita companyia com al nostre cas (tallers mecànics) degut a la seva facilitat de configuració, així com per executar-se en la plataforma que habitualment trobarem en aquest tipus d'organització, Microsoft Windows.
Algunes de les característiques més importants d'aquest servidor són:

- Ofereix servei de FTP, Ghopher, World Wide Web, enviament e-mail i notícies.
- Entorn fiable i d'alt rendiment.
- Entorn segur; suporta alguns dels protocols de seguretat més importants com Kerveros v5, Digest Authentication, etc...
- Capacitat d'executar les aplicacions en coles de processos diferents a les emprades pels Serveis Web.
- Fàcil administració mitjançant consola, MMC (Microsoft Management Console), a més de poder-ho fer mitjançant línia de comandes.
- Pot ser administrat i mantingut remotament .
- Permet creació d'errors personalitzats.
- Pel que es refereix a al servei World Wide Web, permet allotjar diverses aplicacions en el mateix servidor.

3.6. Model de 3 capes i Sistema Gestor de Bases de Dades

El Sistema Gestor de Bases de Dades (SGBD) constitueix la capa de Dades de la nostra aplicació, basada en una arquitectura de 3 capes. Per tant, abans d'entrar en detall del SGBD explicarem que suposa una arquitectura de 3 capes com la que nosaltres hem adoptat.

3.6.1. Model de tres capes

L'arquitectura de 3 capes és un model d'aplicació que estructura aquesta en tres nivells o capes: capa de presentació, capa de negoci i capa de dades.

L'adopció d'aquest model implica bàsicament millores en quant al manteniment i escalabilitat i entén l'aplicació formada per capes independents entre si, que poden ser desenvolupades de manera independent i que a més, poden ajudar a fer l'aplicatiu independent de la plataforma o almenys d'algun dels seus elements. Per exemple, podríem arribar a fer que l'aplicació fos independent d'un SGBD concret, de manera que, per exemple, tant pogués funcionar de manera correcta amb SQL Server o Oracle, simplement modificant una de les capes, però sense afectar a la resta de l'aplicatiu.

La següent figura mostra una arquitectura típica de 3 capes

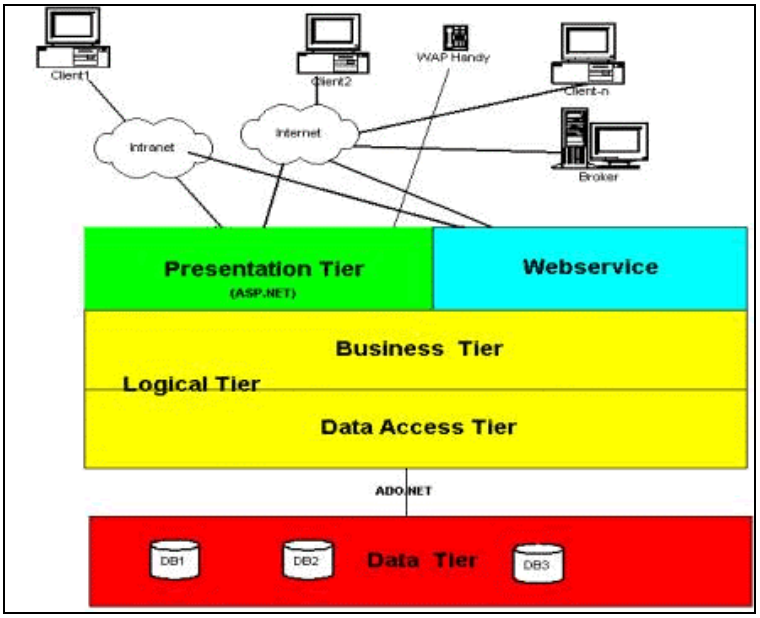


Figura 8: Arquitectura en 3 capes

Com es pot veure a la figura, la capa de dades (Data Tier) està constituïda pel SGDB i com explicarem posteriorment, s'encarrega de l'emmagatzemament de les dades així com de rebre sol·licituds d'emmagatzematge o recuperació de dades per part de la capa de negoci. La lògica de procés de les dades emmagatzemades a la capa de dades constituirà la capa de negoci (Business Tier), i la capa de presentació (Presentation Tier) serà la responsable de la comunicació amb l'usuari. Com aspecte important a comentar, podem veure que a la figura 8, la capa de negoci forma part del que s'anomena capa lògica (Logical Tier) així com la capa d'accés a dades (Data Access Tier). En el nostre cas, però, sols hem implementat la primera, però en cas de desitjar un aplicatiu independent d'un SGBD concret, hauríem de desenvolupar també la capa d'accés a dades, una diferent per cada SGBD, sense afectar a la resta de l'aplicatiu.

Per acabar amb aquesta breu introducció a l'arquitectura 3 capes, comentar que a més dels avantatges que hem exposat, aquest plantejament també implica algun inconvenient, el més destacable és que podria suposar un augment en la càrrega que suposen les comunicacions per la xarxa, degut a un increment en el tràfic de dades.

3.6.2. Sistema Gestor de Bases de Dades

SQL Server 2005 és el servidor de dades emprat, per ser més correctes, és el Sistema Gestor de Bases de Dades (SGBD) utilitzat. Per tant, SQL Server s'encarregarà de tot el treball d'administració de dades, permetent-me a mi centrar-me en els detalls de la nostra aplicació.

En realitat SQL Server és un Sistema Gestor de Bases de Dades relacional, RDMS (Relational DataBase Management System) i l'element principal d'aquest és la base de dades, entenent com a tal l'objecte lògic que permet l'emmagatzemament i recuperació de dades. Dit d'altre manera, una base de dades és una col·lecció de dades estructurada. Al tractar-se d'una base de dades relacional, les dades que aquesta emmagatzema s'estructuraren en taules separades, relacionades entre sí.

Els RDMS comporten avantatges davant altres sistemes d'emmagatzematge:

- Reducció de la redundància
- Eliminació de la inconsistència
- Informació compartida

- Estandardització de les dades
- Controls de seguretat
- Manteniment de la integritat
- Aconseguir un rendiment més alt
- Independència de les dades

Tot i això els sistemes de bases de dades també tenen inconvenients, bàsicament pel que fa a problemes d'integritat i seguretat. Al treballar amb la nostra base de dades, aquesta sempre ha de ser fiable i les dades no poden tenir incongruències o inconsistències. Per aconseguir-ho, a l'hora de fer el disseny de la base de dades tindrem en compte la *integritat* i la *normalització*.

Per integritat entenem la configuració dels valors de la base de dades que ens porten a un estat segur i representable en el món real. Per tant, si la informació compleix amb les Regles d'Integritat direm que la informació de la base de dades és correcta. Bàsicament apliquem dues regles d'integritat:

- Regla d'Integritat de les Entitats: Cap component de la clau primària[4] pot acceptar valors NULL.
- Regla d'Integritat Referencial: La base de dades no ha de tenir valors de clau externa[5] sense concordança.

En quant a la normalització pretén mitjançant les formes normals (primera, segona, tercera, quarta i una variació de la tercera forma normal anomenada forma normal de Boyce-Codd) mantenir l'estructura lògica de les dades en una forma normal "neta", "purificada", per tal de mitigar els problemes de les anomalies d'inserció, esborrat i actualització que ocasionen treball innecessari ja que s'han d'aplicar els mateixos canvis a varis casos de dades, així com evitar el problema de la pèrdua accidental de dades o la dificultat de representació de determinats fets.
Per tant, la normalització és un procés de transformacions progressives que s'aplica per aconseguir la forma normal desitjada, i ha de ser reversible, en cas contrari voldrà dir que s'han perdut dades.

[4] Entenem per clau primària aquell atribut(o atributs) que identifiquen de forma única cada tupla d'una relació.

[5] Entenem per clau externa l'atribut(o atributs) d'una relació R2, els valors del qual han de coincidir amb valors d'una clau primària d'una relació R1 (R1, R2 no tenen per què ser diferents).

En el nostre cas particular, SQL Server posarà a la nostra disposició mecanismes per assegurar la integritat de les dades.

Normalment el RDMS respondrà a peticions escrites en un llenguatge d'alt nivell com SQL (Structured Query Language), encarregant-se el propi RDMS dels detalls d'emmagatzematge, obtenció, format i transmissió de les dades.

A la nostra aplicació és freqüent l'accés a les dades contingudes al RDMS. Per accedir a les dades ho podríem fer executant consultes SQL que formarien part del propi codi de la nostra aplicació però no seria la millor forma de fer-ho ja que complicaria el manteniment. Una millor manera de fer això és utilitzant una característica avançada que inclou SQL Server (així com altres RDMS): els Stored Procedures. Es tracta de blocs de codi SQL que es guarden al mateix RDMS, i que poden ser cridats des de qualsevol altre aplicatiu. Mitjançant l'ús d'aquests aconseguim, entre altres, millorar el rendiment de la nostra aplicació, al tractar-se de codi precompilat, i que per tant s'executa més ràpid. A més, amb el seu ús millorarem la seguretat doncs no viatja per la xarxa tota la informació referent a la sentència a executar, facilitem el manteniment i afavorim la reutilització de codi.

4. Disseny Funcional

En aquest punt del document es descriurà la solució adoptada per donar resposta als requeriments ja exposats, així com el procés seguit des de que a partir dels requeriments inicials del sistema es va realitzar el disseny gràfic de les interfícies d'usuari fins a desenvolupar l'aplicació, passant per la definició del model de dades.

4.1. Disseny d'Interfícies d'Usuari

Un dels primers passos importants que vaig fer en la realització d'aquest projecte va ser el disseny de les interfícies d'usuari. L'elaboració d'uns prototipus inicials, l'evolució dels quals es presenten en aquest document, em va ajudar a aclarir la informació que es visualitzaria en cada cas, i com es duria a terme.

Per altra banda, podem considerar que el procés de prototipatge no ha finalitzat, sinó que es poden realitzar proves amb usuaris per determinar quin disseny és més simple i fàcil d'utilitzar, quin disseny els satisfà més, per configurar-lo com el disseny final.

Abans de mostrar alguna de les interfícies més significatives és important conèixer alguns dels criteris que s'han seguit. Principalment, hem de tenir en compte que l'abús d'alguns efectes fan que les pàgines resultin molt atractives visualment, però aquesta bona sensació desapareix ràpidament quan l'usuari ha de suportar un temps de descàrrega massa alt. Aquest tipus de problema relacionat amb el temps de descàrrega és molt determinant en el cas d'aplicacions disponibles a Internet, però no tant en el nostre cas, al tractar-se d'una aplicació d'ús intern.
Per tant, en el disseny de les nostres pàgines s'ha intentat mantenir l'equilibri, de manera que la nostra aplicació no resulti simplement agradable, sinó que també sigui funcional, usable. Així doncs, es tracta d'un disseny totalment centrat en l'usuari, on aspectes com el fàcil accés a les tasques més comuns o la fàcil llegibilitat són d'especial importància. Per tant, el disseny s'ha realitzat tenint en compte els següents criteris de qualitat:

- Facilitat d'aprenentatge
- Velocitat d'ús
- Freqüència d'errors
- Facilitat de retenció

Es tracta de criteris contraposats; per exemple, per reduir la freqüència d'errors cal un temps d'aprenentatge llarg o velocitat d'ús baixa (moltes confirmacions, missatges d'ajuda, etc...). Així, per valorar-los haurem de tenir en compte el tipus d'usuari que ha de treballar amb l'aplicació, bàsicament tenint en compte els seus coneixements i experiència ja que les seves necessitats seran diferents; per exemple, un usuari amb poca experiència necessitarà feedback informatiu després de cada pas mentre que un usuari amb més experiència valorarà més un temps de resposta baix.
En el nostre cas, com vàrem mencionar, ens adrecem a uns usuaris que tenen pocs coneixements informàtics i poca experiència en la navegació web però per altre banda, al tractar-se d'una eina d'ús diari, requeriran una aplicació que els permeti una alta velocitat d'ús.

Seguint les pautes i criteris anteriors s'ha dissenyat la interfície d'usuari que mostrem a continuació, amb alguns exemples significatius, on es poden veure algunes de les decisions de disseny adoptades.

A la primera de les imatges (figura 9) podem veure la pantalla que ens permet gestionar el detall d'un client així com els seus vehicles, i en segon terme, deshabilitat, el buscador que ens permet localitzar clients i vehicles per diferents criteris de cerca. Hi han varis aspectes que considero especialment important remarcar:

- Homogeneïtat: En aquesta pantalla podem veure algunes convencions d'estil que s'adopten, com per exemple, mostrar els camps obligatoris amb un color de fons diferent (groc). Aquests criteris s'aniran repetint al llarg de totes les interfícies, de manera que l'aparença sigui homogènia i per tant, faciliti l'aprenentatge.
- Deshabilitar parts de la pantalla que no són d'aplicació en un determinat moment; evitem confusió per l'usuari i per tant, evitem que aquest cometi errors.
- Ús de pocs colors i de combinacions d'aquests que faciliten la llegibilitat (per ex., fons blanc i color de lletra negra).
- Mantenim el contexte; segons diversos criteris, l'usuari realitza cerques de clients al buscador i per visualitzar el detall d'algun d'aquest, simplement ha de clickar-hi a sobre. D'aquesta manera s'obre en una nova finestra el detall del client, de manera que en segona instància manté el llistat i hi pot

tornar en qualsevol moment, sense moure's de pàgina, ni haver de tornar a fer la cerca, fent per tant més còmode l'ús de l'aplicació.

En aquest sentit, destacar que al llarg de tot l'aplicatiu s'ha intentat que l'usuari en tot moment sàpiga on es troba, mantenint el contexte, minimitzant el número de salts entre pàgines. Per portar això a terme s'han utilitzat tècniques com l'ús de pestanyes, de manera que es pugui anar fàcilment d'una pàgina a altre, sense perdre el contingut visualitzat a cadascuna d'aquestes.

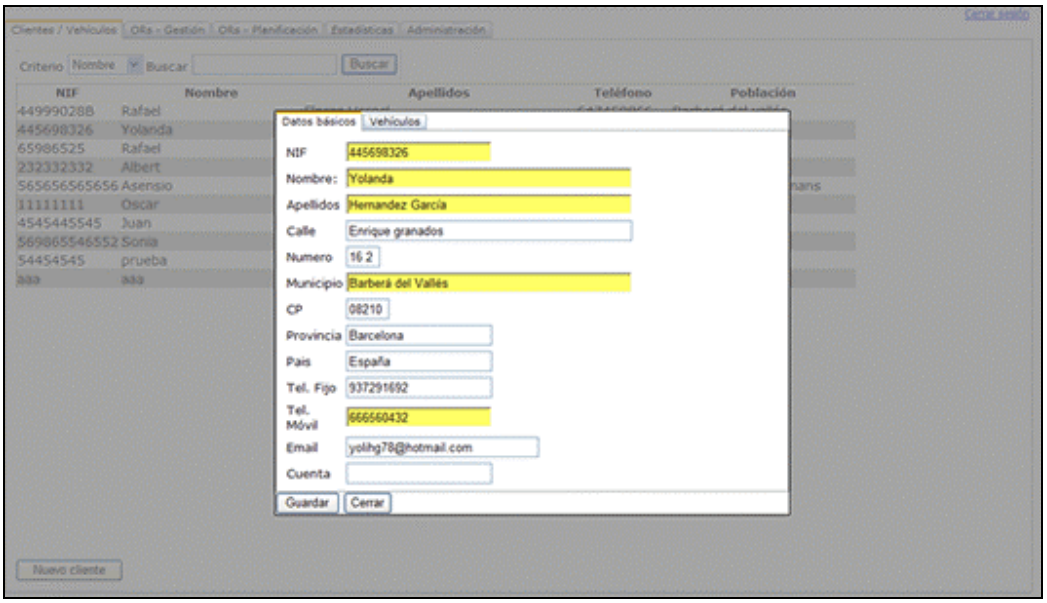


Figura 9: Pantalla de detall de client amb buscador en segon terme en mode deshabilitat

A la figura 10 i 11 podem veure alguns dels recursos emprats per tal de facilitar el treball dels usuaris. En el primer cas (figura 10) fem servir un control calendari, de manera que l'usuari pugui escollir còmodament la data. L'altre cas (figura 11) es tracta d'un control avançat d'AJAX, que permet que conforme l'usuari va escrivint, el contingut es vagi actualitzant en base a la informació de la base de dades. Concretament, l'usuari introdueix la matrícula, i en base al que escriu, l'aplicació ho completa per tal que l'usuari seleccioni alguna de les disponibles. Com veiem, en els dos casos es tracta de recursos que fan que la interfície sigui més còmoda per l'usuari permetent-li guanyar velocitat d'ús i per tant, incrementar la seva productivitat.

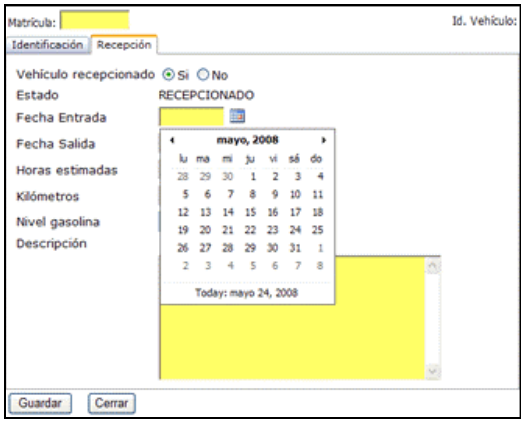


Figura 10:Detall d'OR

Figura 11: Detall d'OR

A la següent figura (figura 12) queda il·lustrat un altre comportament implementat a gran part de l'aplicació amb l'objectiu de fer el seu ús més intuïtiu. Ens referim a que l'usuari tingui feedback del que està portant a terme l'aplicació. Concretament, s'indica a l'usuari que s'està processant la seva petició, enviant-se per tant, la informació necessària al servidor. En aquest cas, tornem a fer ús d'AJAX, per únicament enviar la informació necessària, no tota la pàgina, i refrescant únicament la part de la pàgina que resulti convenient. Aquest feedback es pot veure en altres aspectes de l'aplicació, com per exemple, al posicionar-nos a una fila de qualsevol de les taules i canviar aquesta de color.

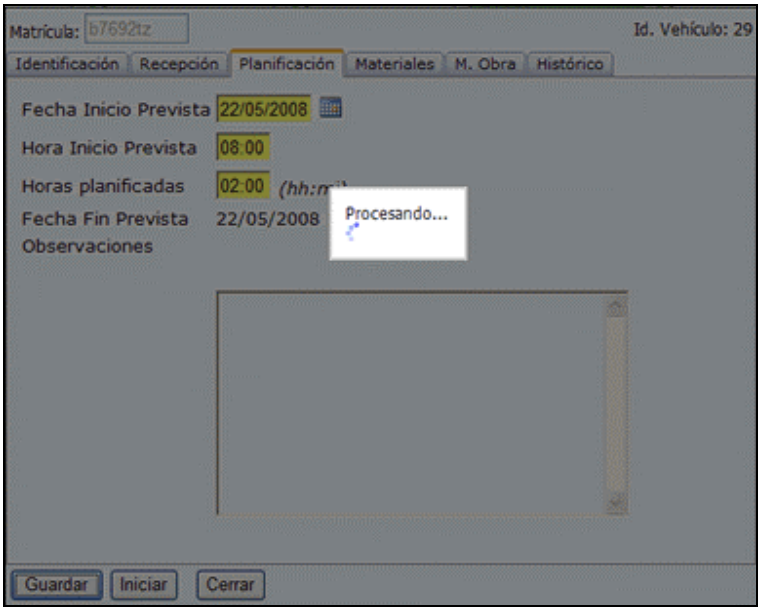


Figura 12: Detall OR

Per acabar, a la següent figura podem veure una de les pantalles que millor reflexa el dinamisme i facilitat d'ús que perseguim amb aquesta aplicació. Es tracta d'una eina visual que permet de manera ràpida i amb pocs clicks de ratolí crear nous treballs, i planificar-los convenientment. A més, permet accedir al detall dels treballs així com aspectes més generals, com el grau d'ocupació de les diferents franges horàries. Tot això sense recarregues de pàgines, de manera totalment dinàmica, gràcies a l'ús d'AJAX.

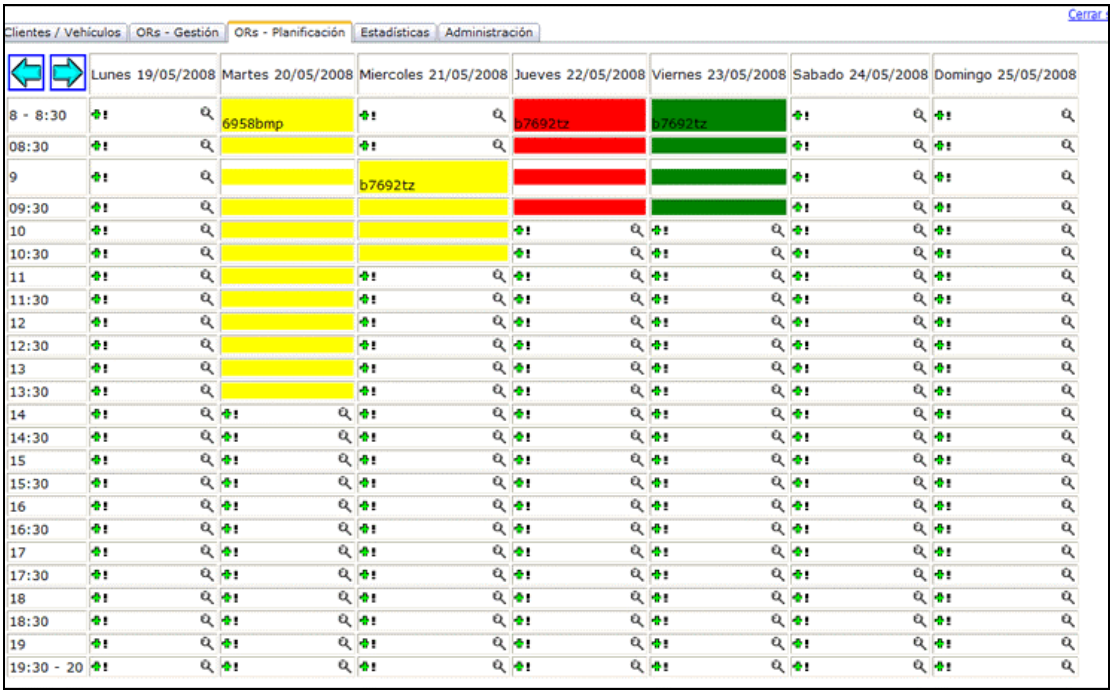


Figura 13: Calendari de planificació de treballs

En aquest apartat hem mostrat algunes de les interfícies més significatives, per tal de mostrar aspectes del seu disseny que es repetiran al llarg de les diferents pantalles, i que intentaran ajudar al nostre objectiu global d'incrementar la productivitat dels usuaris.

En un proper capítol, on detallem el disseny del sistema, podrem veure la totalitat de les pàgines.

4.2. Model de Dades

La base de dades és una part vital de l'aplicació, sense ella no hauríem pogut implementar-la ja que precisament es basa en la informació que aquesta emmagatzema. Per tant, el disseny de la base de dades condiciona considerablement la resta de l'aplicació i la manera com aquesta accedeix a la informació en ella emmagatzemada. Al seu disseny s'han tingut en compte els aspectes teòrics explicats en el capítol anterior, extrets bàsicament de l'assignatura base de dades.

Al tractar-se d'una base de dades relacional , haurem d'estudiar les taules que ella implica i les relacions entre elles. Per tant, mostrarem les taules emprades, amb una breu descripció de les dades que emmagatzemen així com la descripció i tipus dels seus camps. Un cop haguem vist el detall de totes les taules, inclouré un diagrama entitat-relació on es podrà veure el detall de les relacions entre aquestes. Per acabar explicaré quins han estat els Stored Procedures desenvolupats i quina és la seva funcionalitat, així com els seus paràmetres d'entrada i valors de retorn.

4.2.1. Taules

tbClientes

Emmagatzema tota la informació referent als clients.

Camp	Tipus de dades	Descripció
idCliente	int	Identificador únic de client
sNombre	Varchar(50)	Nom del client
sApellidos	Varchar(100)	Cognoms del client
sCalle	Varchar(100)	Carrer del domicili
sNumero	Varchar(4)	Número de domicili
sProvincia	Varchar(20)	Província de domicili
sPais	Varchar(20)	País de residència
sNif	Varchar(20)	NIF / CIF o NIE del client
sTelFijo	Varchar(20)	Telèfon fixe del client
sTelMovil	Varchar(20)	Telèfon mòvil de, client
sEmail	Varchar(30)	Email del client
sMunicipio	Varchar(50)	Municipi del domicili
sCP	Varchar(5)	Codi postal del domicili
sCuenta	Varchar(50)	Núm. compte corrent

tbVehiculos

Conté tots els vehicles que pertanyen als clients. Com és lògic, aquest taula està relacionada amb l'anterior, tbClientes, pel camp idCliente.

Camp	Tipus de dades	Descripció
idVehiculo	int	Identificador únic de vehicle
idCliente	int	Identificador del client propietari
idMarca	int	Identificador de la marca del vehicle
idModelo	int	Identificador del model del vehicle
sVersion	Varchar(50)	Versió del vehicle
sMatricula	Varchar(50)	Matrícula del vehicle
sBastidor	Varchar(50)	Batidor del vehicle
codColor	Varchar(50)	Codi del color del vehicle
codMotor	Varchar(50)	Codi del motor del vehicle

tbORS

Com ja vam comentar amb anterioritat una de les funcionalitats bàsiques d'aquesta aplicació és la possibilitat de planificar els treballs a realitzar als vehicles del clients així com poder supervisar l'evolució d'aquests. Doncs bé, cadascun d'aquests treballs és el que s'anomena Ordre de Reparació (OR), i és precisament el que es guarda en aquesta taula.

Camp	Tipus de dades	Descripció
idOR	int	Identificador únic de la OR
idVehiculo	int	Identificador únic del vehicle afectat
bRecepcionado	bit	Flag que indica si ha estat o no recepcionat el vehicle
codEstado	Varchar(50)	Estat de la OR
dtFechaEntrada	datetime	Data d'entrada prevista
dtFechaSalida	datetime	Data de sortida prevista
nHorasEstimado	int	Hores previstes per la OR
nKms	int	Quilòmetres que té el vehicle en el moment de recepcionar-se
codNivGasolina	Varchar(50)	Nivell de benzina del vehicle en el moment de recepcionar-se
sDescripcion	Varchar(100)	Descripció dels treballs a realitzar
dtFechaInicioPrevista	datetime	Data prevista d'inici dels treballs
dtFechaFinPrevista	datetime	Data prevista d'acabament dels treballs
nHorasPlanificadas	int	Número d'hores planificades pels treballs
sObservacionesPlanificacion	Varchar(100)	Observacions
codProcedencia	Varchar(50)	Codi de la procedència (si aquesta s'ha donat d'alta des de Recepció o per Planificació)

.

tbHistoricoOR

Mitjançant aquesta taula s'enregistren les accions que es porten a terme a les ORs, concretament s'enregistren els següents esdeveniments o accions :

- Creació de la OR
- Planificació de la OR
- Inici de la OR
- Pausa de la OR
- Finalització de la OR

A més, per cadascun d'aquests esdeveniments, es registra la persona que ho causa així com la data i hora en que es porta a terme.

Camp	Tipus de dades	Descripció
idHistoricoOR	int	Identificador únic del registre
idOR	int	Identificador únic de la OR relacionada
idUsuario	int	Identificador únic de l'usuari que porta a terme l'acció
codAccion	Varchar(50)	Codi d'acció
dtFecha	datetime	Data en que es porta a terme l'acció

tbUsuarios

Relació d'usuaris autoritzats per treballar amb l'aplicatiu. Conté totes les dades de l'usuari , incloses les d'accés a l'aplicatiu. Una dada molt important que també s'emmagatzema en aquesta taula és el Rol, comentat en capítols anteriors, i el qual condicionarà les funcionalitats a les que podrà accedir l'usuari.

Camp	Tipus de dades	Descripció
idUsuario	int	Identificador únic d'usuari
sNombre	Varchar(50)	Nom d'usuari
sApellidos	Varchar(50)	Cognoms de l'usuari
codRol	Varchar(50)	Rol o funció que té l'usuari a la organització. Els possibles valors són REC, MEC, JEF que corresponen a Recepció, Mecànic i Cap de taller, respectivament.
codUsuario	Varchar(50)	Codi d'usuari
sPassword	Varchar(50)	Password d'accés
BHabilitado	Bit	Valor lògic que indica si té o no habilitat l'accés a l'aplicatiu

tbMaterialesxOR

Mitjançant aquesta taula es manté la relació de materials emprats a una OR determinada.

Camp	Tipus de dades	Descripció
idMaterialOR	int	Identificador únic del registre
idOR	int	Identificador únic de la OR relacionada
idMaterial	int	Identificador únic del material
nCantidad	int	Número d'unitats emprades d'aquest material

tbOperacionesxOR

A més de materials, a una OR s'inclouen tota una relació de tasques o operacions que s'han de dur a terme per completar la OR. Aquesta informació és el que emmagatzema aquesta taula.

Camp	Tipus de dades	Descripció
idOperacionOR	int	Identificador únic del registre
idOR	int	Identificador únic de la OR relacionada
idOperacion	int	Identificador únic de l'operació
nMinutosTrabajo	Varcharint	Número de minuts que s'han dedicat a realitzar la operació

tbMateriales

Relació de tots aquells materials que es poden fer servir a les OR, i que per tant constaran a la taula tbMaterialesxOR, per les OR que correspongui.

Camp	Tipus de dades	Descripció
idMaterial	int	Identificador únic del registre
sReferencia	Varchar(50)	Identificador únic de la OR relacionada
sDescripcion	Varchar(50)	Identificador únic del material

tbOperaciones

Relació de totes aquelles operacions que es poden dur a terme a una OR. És important remarcar la importància del camp *nMinutosEstandar* ja que és el temps que es preveu per realitzar la operació, i s'ha d'ajustar al temps real que els operaris han de dedicar, i que fan constar així a la taula tbOperacionesxOR.

Camp	Tipus de dades	Descripció
idOperacion	int	Identificador únic del registre
codOperaciojn	Varchar(50)	Identificador únic de la OR relacionada
sDescOperacion	Varchar(50)	Identificador únic de l'operació

nMinutosEstandar	int	Número de minuts previst per portar a terme l'operació

tbMarcasVehiculo

Conté la relació de marques de vehicles disponibles a l'aplicatiu.

Camp	Tipus de dades	Descripció
idMarca	int	Identificador únic de marca de vehicle
sDescripcion	Varchar(50)	Descripció de la marca

tbModelosxMarcaVehiculo

Conté la relació dels models disponibles per cada marca de vehicles.

Camp	Tipus de dades	Descripció
idMarca	int	Identificador únic de marca de vehicle
idModelo	int	Identificador únic del model de vehicle
sDescripcion	Varchar(50)	Descripció del model

4.2.2. Diagrama Entitat – Relació

A la següent imatge (figura 14) es mostra el diagrama entitat-relació del model de dades, on es poden veure perfectament les relacions entre les diferents taules.

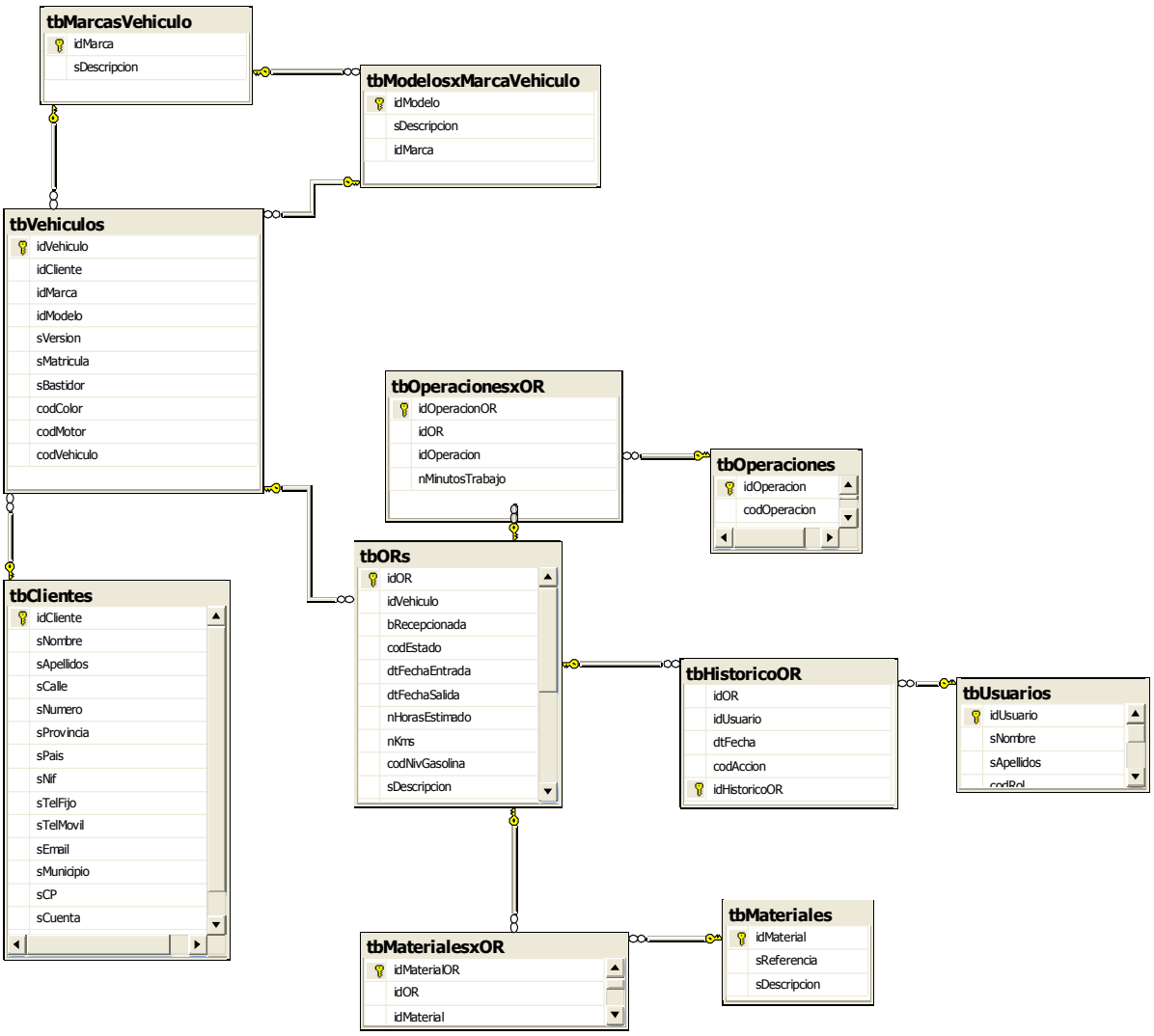


Figura 14: Diagrama entitat-relació model de dades

Com es pot veure, el RDMS utilitzat, SQL Server, representa gràficament les claus primàries (PK – Primary Key) de cadascuna de les taules mitjançant la icona  i constituiran el que es coneix com claus foranees (FK – Foreign Keys) de les taules amb les que es relaciona. Per exemple, veiem que a la taula tbClientes la PK és el camp idCliente, doncs bé, podem dir que aquest mateix camp és FK de la taula tbVehiculos.

4.2.3. Stored Procedures

En capítols anteriors ja vam comentar el que són els Stored Procedures (SP) així com els avantatges que suposa el seu ús. A continuació mostrarem el detall dels Stored Procedures desenvolupats.

ActualizarCliente

Encapsula la lògica necessària per actualitzar les dades d'un client i retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idCliente	int	Identificador del client
nombre	Varchar(50)	Nom del client
apellidos	Varchar(100)	Cognoms del client
calle	Varchar(100)	Carrer del domicili
numero	Varchar(4)	Número del domicili
provincia	Varchar(20)	Província del domicili
pais	Varchar(20)	País de residència
nif	Varchar(20)	NIF
telFijo	Varchar(20)	Telèfon fixe
telMovil	Varchar(20)	Telèfon mòbil
municipio	Varchar(50)	Municipi on resideix
CP	Varchar(5)	CP del municipi on resideix
cuenta	Varchar(50)	Núm. de compte corrent
email	Varchar(30)	Email del client

ActualizarMarca

S'encarrega d'actualitzar una marca concreta de vehicles, i retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idMarca	int	Identificador de marca de vehicle
descripcion	Varchar(50)	Descripció de la marca

ActualizarMaterial

Encapsula la lògica necessària per actualitzar els materials que poden ser utilitzats als treballs realitzats als vehicles. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idMaterial	int	Identificador de material
referencia	Varchar(50)	Referència del material
descripcion	Varchar(50)	Descripció del material

ActualizarModelo

Stored Procedure encarregat d'insertar o actualitzar, segons convingui, un model concret de vehicle, conforme als paràmetres rebuts. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idMarca	int	Identificador del vehicle
idModelo	int	Identificador de model
descripcion	Varchar(50)	Descripció del model

ActualizarOperacion

Utilitzat per actualitzar les dades d'alguna de les operacions predefinides que es porten a terme a les ORs. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idoperacion	int	Identificador de la operació
codigo	Varchar(50)	Codi d'operació
descripcion	Varchar(50)	Descripció de la operació
minutos	int	Número de minuts necessari per portar-la a terme

ActualizarOR

Porta a terme l'actualització del detall d'una OR, en base als paràmetres rebuts. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	int	Identificador del vehicle
idOR	int	Identificador de la OR
bRecepcionada	bit	Indicador de si el vehicle s'ha recepcionat
estado	Varchar(50)	Estat actual de la OR
fechaEntrada	datetime	Data d'entrada prevista del vehicle
fechaSalida	datetime	Data de sortida del vehicle
horasEstimado	int	Temps estimat per portar a terme la OR
kms	int	Kms que té el vehicle en el moment d'iniciar la OR
codNivGasolina	Varchar(50)	Nivell de benzina que té el vehicle en el moment d'iniciar la OR
descripcion	Varchar(100)	Descripció de treballs a realitzar

ActualizarORPlanificada

Porta a terme l'actualització del detall d'una OR, en base als paràmetres rebuts, però sols d'aquelles ORs que es troben planificades. A més, depenent del tipus de modificació, aquesta s'enregistra a l'històric. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	int	Identificador del vehicle
idOR	int	Identificador de la OR
bRecepcionada	bit	Indicador de si el vehicle s'ha recepcionat
estado	Varchar(50)	Estat actual de la OR
FechaEntrada	datetime	Data d'entrada prevista del vehicle
FechaSalida	datetime	Data de sortida del vehicle
HorasEstimado	int	Temps estimat per portar a terme la OR
Kms	int	Kms que té el vehicle en el moment d'iniciar la OR
CodNivGasolina	Varchar(50)	Nivell de benzina que té el vehicle en el moment d'iniciar la OR
descripcion	Varchar(100)	Descripció de treballs a realitzar
fechaInicioPrevisto	datetime	Data d'inici previst
fechaFinPrevista	datetime	Data de final previst
HorasPlanificadas	int	Núm. d'hores planificades
idUsuario	int	Identificador de l'usuari que actualitza la OR
observaciones	Varchar(100)	Observacions

ActualizarUsuario

S'utilitza per portar a terme l'actualització d'un usuari de l'aplicació. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idUsuario	int	Identificador de l'usuari
nombre	Varchar(50)	Nom d'usuari
apellidos	Varchar(50)	Cognoms de l'usuari
rol	Varchar(50)	Rol de l'usuari
codUsuario	Varchar(50)	Codi d'usuari
password	Varchar(50)	Contrasenya d'accés

ActualizarVehiculo

S'utilitza per portar a terme l'actualització d'un vehicle. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	int	Identificador de vehicle
idMarca	int	Identificador de marca
idModelo	int	Identificador de model
version	Varchar(50)	Versió del vehicle
matricula	Varchar(50)	Matrícula del vehicle
bastidor	Varchar(50)	Núm. bastidor
codColor	Varchar(50)	Codi del color
codMotor	Varchar(50)	Codi del motor

AnyadirMaterialesOR

Aquest SP és utilitzat per tal d'afegir els materials emprats a una OR. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR
idMaterial	int	Identificador del material
cantidad	int	Núm. d'unitats d'aquest material emprades

AnyadirOperacionesOR

Aquest SP és utilitzat per tal d'afegir les operacions realitzades a una OR. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR
idOperacion	int	Identificador de la operació
minutos	int	Núm. de minuts dedicats a portar a terme la operació

EliminarMaterialesOR

Utilitzat per tal d'eliminar algun dels materials que consten com utilitzats a una OR. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idMaterialOR	int	Identificador del registre a eliminar

EliminarOperacionesOR

Utilitzat per tal d'eliminar alguna de les operacions que consten com realitzades a una OR. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOperacionOR	int	Identificador del registre a eliminar

finalizarOR

Aquest SP és utilitzat per tal de finalitzar una OR, modificant el seu estat i fent-lo constar al seu històric, quedant enregistrat l'usuari i data. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR
idUsuario	int	Identificador de l'usuari que finalitza la OR

iniciarOR

Aquest SP és utilitzat per tal d'iniciar una OR, modificant el seu estat i fent-lo constar al seu històric, quedant enregistrat l'usuari i data. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR
idUsuario	int	Identificador de l'usuari que inicia la OR

insertarCliente

Encapsula la lògica necessària per insertar un client i retorna l'identificador assignat a aquest.

Nom paràmetre	Tipus paràmetre	Descripció
nombre	Varchar(50)	Nom del client
apellidos	Varchar(100)	Cognoms del client
calle	Varchar(100)	Carrer del domicili
numero	Varchar(4)	Número del domicili
provincia	Varchar(20)	Província del domicili
pais	Varchar(20)	País de residència
nif	Varchar(20)	NIF
telFijo	Varchar(20)	Telèfon fixe
telMovil	Varchar(20)	Telèfon mòbil
municipio	Varchar(50)	Municipi on resideix
CP	Varchar(5)	CP del municipi on resideix
cuenta	Varchar(50)	Núm. de compte corrent
email	Varchar(30)	Email del client

insertarMaterial

Encapsula la lògica necessària per insertar els materials que poden ser utilitzats als treballs realitzats als vehicles. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
referencia	Varchar(50)	Referència del material
descripcion	Varchar(50)	Descripció del material

insertarOperacion

Utilitzat per insertar operacions predefinides que es porten a terme als treballs que es realitzen als vehicles. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
codigo	Varchar(50)	Codi d'operació
descripcion	Varchar(50)	Descripció de la operació
minutos	int	Número de minuts necessari per portar-la a terme

insertarOR

Porta a terme la creació d'una OR, en base als paràmetres rebuts. Retorna l'identificador assignat a la OR.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	int	Identificador del vehicle
bRecepcionada	bit	Indicador de si el vehicle s'ha recepcionat
estado	Varchar(50)	Estat actual de la OR
fechaEntrada	datetime	Data d'entrada prevista del vehicle
fechaSalida	datetime	Data de sortida del vehicle
horasEstimado	int	Temps estimat per portar a terme la OR
kms	int	Kms que té el vehicle en el moment d'iniciar la OR
codNivGasolina	Varchar(50)	Nivell de benzina que té el vehicle en el moment d'iniciar la OR
descripcion	Varchar(100)	Descripció de treballs a realitzar
idUsuario	Int	Usuari que crea la OR

insertarORPlanificada

Porta a terme la creació d'una d'una OR i la posa com planificada, en base als paràmetres rebuts. A més, al tractar-se d'una OR planificada, s'anota la planificació a l'històric. Retorna l'identificador assignat a la OR.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	int	Identificador del vehicle
bRecepcionada	bit	Indicador de si el vehicle s'ha recepcionat
estado	Varchar(50)	Estat actual de la OR
fechaEntrada	datetime	Data d'entrada prevista del vehicle
fechaSalida	datetime	Data de sortida del vehicle
horasEstimado	int	Temps estimat per portar a terme la OR
kms	int	Kms que té el vehicle en el moment d'iniciar la OR
codNivGasolina	Varchar(50)	Nivell de benzina que té el vehicle en el moment d'iniciar la OR
descripcion	Varchar(100)	Descripció de treballs a realitzar
fechaInicioPrevisto	datetime	Data d'inici previst
fechaFinPrevista	datetime	Data de final previst
HorasPlanificadas	int	Núm. d'hores planificades
idUsuario	int	Identificador de l'usuari que actualitza la OR
observaciones	Varchar(100)	Observacions

insertarUsuario

S'utilitza per portar a terme la creació d'un usuari. Retorna l'identificador assignat a l'usuari.

Nom paràmetre	Tipus paràmetre	Descripció
nombre	Varchar(50)	Nom d'usuari
apellidos	Varchar(50)	Cognoms de l'usuari
rol	Varchar(50)	Rol de l'usuari
codUsuario	Varchar(50)	Codi d'usuari
password	Varchar(50)	Contrasenya d'accés
habilitado	Bit	Indica si l'usuari té o no accés a l'aplicatiu

insertarVehiculo

S'utilitza per portar a terme la creació d'un vehicle. Retorna l'identificador assignat al vehicle.

Nom paràmetre	Tipus paràmetre	Descripció
idCliente	Int	Identificador del client propietari del vehicle
idMarca	int	Identificador de marca

idModelo	int	Identificador de model
version	Varchar(50)	Versió del vehicle
matricula	Varchar(50)	Matrícula del vehicle
bastidor	Varchar(50)	Núm. bastidor
codColor	Varchar(50)	Codi del color
codMotor	Varchar(50)	Codi del motor

obtenerCliente

S'utilitza per recuperar tota la informació disponible d'un client concret. Retorna un registre amb totes les dades disponibles.

Nom paràmetre	Tipus paràmetre	Descripció
IdCliente	Int	Identificador del client

obtenerClientes

S'utilitza per recuperar tota la informació disponible dels clients que coincideixen amb els criteris de cerca facilitats per paràmetre. Retorna un registre per cada client localitzat, amb totes les dades disponibles.

Nom paràmetre	Tipus paràmetre	Descripció
criterio	Char(1)	Codi del criteri a utilitzar 'n': nom del client 'a': cognom del client 'd': nif del client 'm': matrícula d'algun dels vehicles del client
filtro	Varchar(100)	Valor del criteri a utilitzar

obtenerDesviacionesORxOperacion

S'utilitza per recuperar informació referent a la desviació entre el temps previst per cada operació, temps estàndard de l'operació que consta a tbOperaciones, i el que ha calgut a una OR concreta.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR

obtenerEstadisticasGlobalesOR

S'utilitza per recuperar informació relativa als temps dedicats a les ORs, però sols per aquelles coincidents amb els criteris de cerca facilitats.

Nom paràmetre	Tipus paràmetre	Descripció
fechaInicio	Datetime	Data d'entrada del vehicle de la OR
fechaFin	Datetime	Data de sortida del vehicle de la OR
matricula	Varchar(50)	Matrícula del vehicle de la OR
Dni	Varchar(50)	Dni del client

ObtenerHistoricoOR

Permet recuperar el detall de l'històric d'una OR. Retorna els registres que contenen l'històric de la OR que consta als paràmetres.

Nom paràmetre	Tipus paràmetre	Descripció
IdOR	Int	Identificador únic d'OR

ObtenerMaterial

Permet recuperar el detall d'algun dels materials que consten a la base de dades, retornant el registre corresponent.

Nom paràmetre	Tipus paràmetre	Descripció
IdMaterial	Int	Identificador únic de material

ObtenerMateriales

Permet obtenir una relació de tots els materials disponibles, filtrant aquests segons el valor facilitat per paràmetre. Retorna els registres coincidents.

Nom paràmetre	Tipus paràmetre	Descripció
sFiltro	Varchar(100)	Text que es cercarà al material, tant a la referència com a la descripció

ObtenerMaterialesOR

Permet recuperar el detall de tots aquells materials que consten com empleats a la OR facilitada per paràmetre. Retorna els registres coincidents.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	Varchar(100)	Identificador únic de la OR

ObtenerOperacion

Permet recuperar el detall d'alguna de les operacions predefinides que consten a la base de dades.

Nom paràmetre	Tipus paràmetre	Descripció
idOperacion	int	Identificador únic de la operació

ObtenerOperacionesMO

Permet recuperar el detall de totes aquelles operacions que coincideixen amb els criteris de cerca.

Nom paràmetre	Tipus paràmetre	Descripció
sFiltro	Varchar(100)	Text pel que es filtraran les operacions, tant al codi com a la descripció

ObtenerOperacionesOR

Recupera totes les operacions realitzades a la OR. Retornarà els registres corresponents a les operacions incloses.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	Int	Identificador únic de la OR

ObtenerOR

Recupera el detall de la OR. El retorn consisteix en un registre amb les dades de la taula tbORs.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	Int	Identificador únic de la OR

ObtenerORs

Recupera el detall d'aquelles ORs coincidents amb els criteris de cerca. El retorn seran els registres coincidents.

Nom paràmetre	Tipus paràmetre	Descripció
criterio	Char(1)	Codi del criteri a utilitzar 'n': nom del client 'a': cognom del client 'd': nif del client 'm': matrícula d'algun dels vehicles del client
filtro	Varchar(100)	Valor del criteri a utilitzar

ObtenerORsNoPlanif

Recupera el detall d'aquelles ORs coincidents amb els criteris de cerca i que encara no han estat planificades. El retorn seran els registres coincidents.

Nom paràmetre	Tipus paràmetre	Descripció
criterio	Char(1)	Codi del criteri a utilitzar 'n': nom del client 'a': cognom del client 'd': nif del client 'm': matrícula d'algun dels vehicles del client
filtro	Varchar(100)	Valor del criteri a utilitzar

ObtenerPlanificacionORs

Permet recuperar el detall d'aquelles ORs planificades, i amb data prevista inclosa en l'interval que consta als paràmetres.

Nom paràmetre	Tipus paràmetre	Descripció
fechaInicioIntervalo	datetime	Data d'inici d'interval
fechaFinIntervalo	datetime	Data final d'interval

ObtenerTodaInfoVehiculo

Permet recuperar els detalls del vehicle amb identificador coincident amb el facilitat per paràmetre. Retorna un registre on s'inclouen tots els detalls del vehicle.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	Int	Identificador únic de vehicle

ObtenerUsuario

Cridant a aquest SP obtindrem els detalls de l'usuari amb el que coincideixen les dades d'accés facilitades per paràmetre. Es retorna el registre corresponent a l'usuari.

Nom paràmetre	Tipus paràmetre	Descripció
codUsuario	Varchar(50)	Codi d'usuari
password	Varchar(50)	Contrasenya d'usuari

ObtenerUsuarios

Obté una relació dels detalls de tots els usuaris amb accés a l'aplicatiu, a excepció del password d'accés. Es retornen tants registres com usuaris hi constin.
No existeixen paràmetres d'entrada.

ObtenerUsuarioxId

Retorna els detalls de l'usuari facilitat per paràmetre.

Nom paràmetre	Tipus paràmetre	Descripció
idUsuario	Int	Identificador únic d'usuari

ObtenerVehiculo

Retorna els detalls del vehicle facilitat per paràmetre.

Nom paràmetre	Tipus paràmetre	Descripció
idVehiculo	Int	Identificador únic del vehicle

ObtenerVehiculos

Retorna el detall d'aquells vehicles coincidents amb els criteris de cerca.

Nom paràmetre	Tipus paràmetre	Descripció
criterio	Char(1)	Codi del criteri a utilitzar 'n': nom del client propietari 'a': cognom del client propietari 'd': nif del client 'm': matrícula del vehicle
filtro	Varchar(100)	Valor del criteri a utilitzar

ObtenerVehiculosCliente

Recupera el detall dels vehicles que pertanyen a un client concret. El retorn seran els registres coincidents.

Nom paràmetre	Tipus paràmetre	Descripció
idCliente	int	Identificador únic de client

ObtenerVehiculosMarcas

Recupera totes les marques disponibles de vehicles. El retorn seran tots els registres de marques que constin a la base de dades.
No existeixen paràmetres d'entrada.

ObtenerVehiculosMatricula

Obté el detall d'aquells vehicles on la matrícula coincideix amb la facilitada per paràmetre. Retornarà tants registres com vehicles coincidents trobi.

Nom paràmetre	Tipus paràmetre	Descripció
matricula	Varchar(50)	Matrícula del vehicle

ObtenerVehiculosModelos

Obté el detall d'aquells models de vehicles que pertanyen a la marca facilitada per paràmetre. Retornarà tants registres com models de vehicles coincidents trobi.

Nom paràmetre	Tipus paràmetre	Descripció
idMarca	Int	Identificador únic de la marca

pararOR

Aquest SP és utilitzat per tal de parar l'execució d'una OR, modificant per tant el seu estat i fent-lo constar a l'històric. Retorna un valor de control que ens indica si l'actualització s'ha realitzat correctament.

Nom paràmetre	Tipus paràmetre	Descripció
idOR	int	Identificador de la OR
idUsuario	int	Identificador de l'usuari que para la OR

4.3. Disseny de classes

Com ja vam introduir en un capítol anterior, VB.NET és un llenguatge orientat a objectes i, per tant, s'aplicaran conceptes com les classes, els objectes, la herència, la encapsulació o el polimorfisme.

Com ja sabem, la potència dels llenguatges orientats a objectes es basa en un bon disseny, que permeti la reutilització. Això es tradueix en el disseny de les classes - integrades en la nostra capa de negoci - pel que detallarem les principals que s'han dissenyat, així com els seu mètodes.

Cliente

La classe Cliente modela als clients del nostre taller, que poden ser tant persones físiques com jurídiques. A continuació mostrem el desglòs dels mètodes amb que compta:

InsertarCliente

Dóna d'alta un nou client a la base de dades, utilitzant els valors facilitats per paràmetre. Retorna l'identificador únic assignat al client.

ActualizarCliente

Actualitza les dades d'un client existent, segons les dades rebudes per paràmetre.

RecuperarDatosCliente

Recupera tota la informació relativa a un client concret, pel qual admet el seu identificador únic com a paràmetre.

RecuperarVehiculosCliente

Recupera la relació de vehicles que pertanyen a un client concret, facilitant el seu identificador com a paràmetre.

Vehiculo

La classe Vehiculo modela cadascun dels vehicles dels clients nostre taller. A continuació mostrem el desglòs dels mètodes amb que compta:

RecuperarDatosVehiculo

Recupera el detall del vehicle que correspon a l'identificador únic facilitat per paràmetre.

RecuperarTodaInfoVehiculo

Recupera informació ampliada del vehicle que correspon a l'identificador únic facilitat per paràmetre.

InsertarVehiculo

Dóna d'alta un nou vehicle a la base de dades, utilitzant els valors facilitats per paràmetre, entre altres, l'identificador únic del client propietari. Es retorna l'identificador únic assignat al vehicle.

ActualizarVehiculo

Mitjançant la crida a aquest mètode podem modificar el detall d'un vehicle, amb les dades que constin als paràmetres.

OR

Com hem comentat en nombroses ocasions, una Ordre de Reparació (OR) constitueix cadascun dels treballs que s'han de portar a terme al vehicle del client. Aquests, inclouran les diverses operacions a realitzar així com els materials empleats per poder completar la OR. A continuació mostrem el desglòs dels mètodes amb que compta:

InsertarORPlanificada

Permet la creació d'una nova OR, quedant aquesta planificada. Es retorna l'identificador únic assignat a la OR.

InsertarOR

Permet la creació d'una nova OR, segons els valors dels paràmetres. Es retorna l'identificador únic assignat a la OR.

ActualizarOR

Amb la crida a aquest mètode podem modificar el detall de la OR, amb els valors facilitats per paràmetre.

ActualizarORPlanificada

Amb la crida a aquest mètode podem modificar el detall de la OR, amb els valors facilitats per paràmetre. Em aquest cas, la OR ja està planificada, pel que consta d'informació addicional.

RecuperarDatosOR

Permet recuperar informació de detall de la OR, d'aquella amb identificador coincident amb el facilitat per paràmetre.

IniciarOR

La crida a aquest mètode permet iniciar l'execució d'una OR. D'aquesta manera, es canvia l'estat, s'enregistra l'inici a l'històric i a més, es comença a comptabilitzar el temps dedicat.

FinalizarOR

Cridant a aquest mètode indiquem que els treballs que inclou la OR han finalitzat. A més, s'enregistra a l'històric i es deixa de comptabilitzar el temps que s'hi dedica.

PararOR

La crida a aquest mètode permet pausar l'execució de la OR. A més, s'enregistra aquest fet a l'històric i es deixa de comptabilitzar el temps, tot i que sigui de manera temporal.

Operacion

La classe Operacion modela cadascuna de les tasques que s'inclouen a una OR. Aquestes tasques impliquen dedicar un temps que sumat correspondrà al dedicat a la OR en el seu conjunt. A continuació mostrem el desglòs dels mètodes amb que compta:

InsertarOperacion

Permet la creació d'operacions o tasques predefinides que podran ser utilitzades a les diferents ORs.

ActualizarOperacion

Permet l'actualització d'operacions o tasques predefinides que podran ser utilitzades a les diferents ORs.

Material

La classe Material modela cadascun dels materials inclosos a una OR. A continuació mostrem el desglòs dels mètodes amb que compta:

InsertarMaterial

Permet la creació de materials que podran ser inclosos a les nostres ORs.

ActualizarMaterial

La crida a aquest mètode permet actualitzar els materials que consten a la nostra base de dades i que, per tant, poden ser inclosos a les ORs que es portin a terme.

Usuario

La classe Usuario correspon a cada un dels usuaris que consten a la nostra aplicació i que, per tant, tindran accés. A continuació mostrem el desglòs dels mètodes amb que compta:

InsertarUsuario

Amb la crida a aquest mètode crearem els diferents usuaris que tindran accés a l'aplicació.

ActualizarUsuario

Amb la crida a aquest mètode actualitzarem les dades d'aquell usuari amb identificador coincident amb el facilitat per paràmetre.

ModelosVehiculo

La classe ModelosVehiculo correspon a cada un dels models de vehicles que consten a l'aplicació, per cadascuna de les marques. A continuació mostrem el desglòs dels mètodes amb que compta:

obtenerMarcas

Aquest mètode ens permet recuperar la relació de marques de vehicles que consten a la base de dades.

ObtenerModelosMarca

Aquest mètode ens permet recuperar la relació de models de vehicles enregistrats a la nostra base de dades, per una determinada marca.

CargarModelosMarca

Aquest mètode proveeix un mecanisme per actualitzar de manera massiva les marques i models de vehicles que consten a la base de dades.

A més de les classes mostrades en aquest apartat, a l'aplicatiu s'han definit d'altres, d'un ús més específic, dissenyades per donar resposta a les necessitats dels diferents mètodes que formen part del Servei Web implementat. Així, aquestes classes facilitaran, per exemple, l'intercanvi de dades que es porta a terme entre el client (funcions JavaScript) i els Servei Web, mitjançant tècniques AJAX. Per tant, no les expliquem a aquest apartat enfocat a classes d'ús global (formen part de la capa de negoci), sinó que seran detallades a l'apartat dedicat al Servei Web desenvolupat.

4.4. Disseny del Sistema

En apartats anteriors vam exposar les funcionalitats bàsiques del sistema. Ara mostrarem el detall de totes les funcionalitats, així com aspectes referents a la implementació d'aquestes.

Agruparem les funcionalitats que l'aplicatiu ofereix en sis blocs:

- Control d'accés: Facilita un mecanisme per validar que la persona que intenta accedir a l'aplicatiu està autoritzada, així com que únicament accedeix a aquelles funcionalitats per les que té autorització.

- Gestió de clients i vehicles: Aquest mòdul permet a l'usuari gestionar els clients (nous o existents) així com els seus vehicles, d'una manera ràpida i senzilla.

- Gestió d'ORs: Bloc que proveeix a l'usuari d'un buscador d'ORs, segons diferents criteris de cerca, per poder-hi accedir al detall, a més de permetre donar d'alta noves.

- Planificació d'ORs: Aquest bloc permet a l'usuari, cap de taller, planificar ORs, tant existents com noves. A més, dóna una visió global del grau d'ocupació de les diferents franges horàries així com de l'estat de les ORs, permetent també accedir al detall d'aquestes.

- Estadístiques: Consisteix en una eina bàsica per extreure informació global de les ORs, podent limitar aquestes, segons diversos criteris.

- Administració de paràmetres globals: Eina mitjançant la qual es poden gestionar els usuaris que tenen accés a l'aplicatiu, els materials que es poden utilitzar i les operacions predefinides que es poden incloure a les ORs, així com les marques i models de vehicles disponibles. Per facilitar aquestes tasques, es proveeix a l'usuari de diversos buscadors.

  Com ja vaig comentar, l'accés o no a determinades funcionalitats que aquí es detallen depèn del tipus d'usuari, del rol que jugui al taller, els quals ja han estat àmpliament explicats. Per tant, al detallar cadascun d'aquests blocs, farem referència als rols que podran fer ús d'aquests.
  Abans d'entrar en el detall de cadascun dels blocs citats anteriorment, comentar que el pes específic de cadascun d'aquests en l'aplicatiu és diferent, tant pel grau de complexitat que ha implicat el seu desenvolupament com pels beneficis que aporta als usuaris i taller en el seu conjunt.

4.4.1. Control d'accés

Com ja hem comentat en nombroses ocasions, aquesta aplicació constitueix una eina de gestió exclusivament d'ús intern, únicament accessible per usuaris autoritzats.

La pàgina d'identificació (figura 15) és òbviament la pàgina a través de la qual s'accedeix al sistema, i per tant és necessari disposar d'un codi d'usuari i una contrasenya per poder-se autentificar. Així doncs, si les dades facilitades no són correctes, mostrarem un missatge d'error i tornarem a demanar a l'usuari que s'autentiqui. En cas contrari, si les credencials són correctes, redirigirem a l'usuari a la pàgina principal de l'aplicació i, a més, recuperarem de la base de dades informació de l'usuari com nom, cognoms o tipus d'usuari. Aquesta informació serà emmagatzemada, de manera transparent a l'usuari, en una variable de sessió, en forma d'un objecte de la classe *Usuario,* quedant disponible per ser consultada en qualsevol moment. Així, amb aquesta informació podrem decidir al llarg de tota l'aplicació, quins serveis i informació hem de posar a l'abast de l'usuari, i quins no.

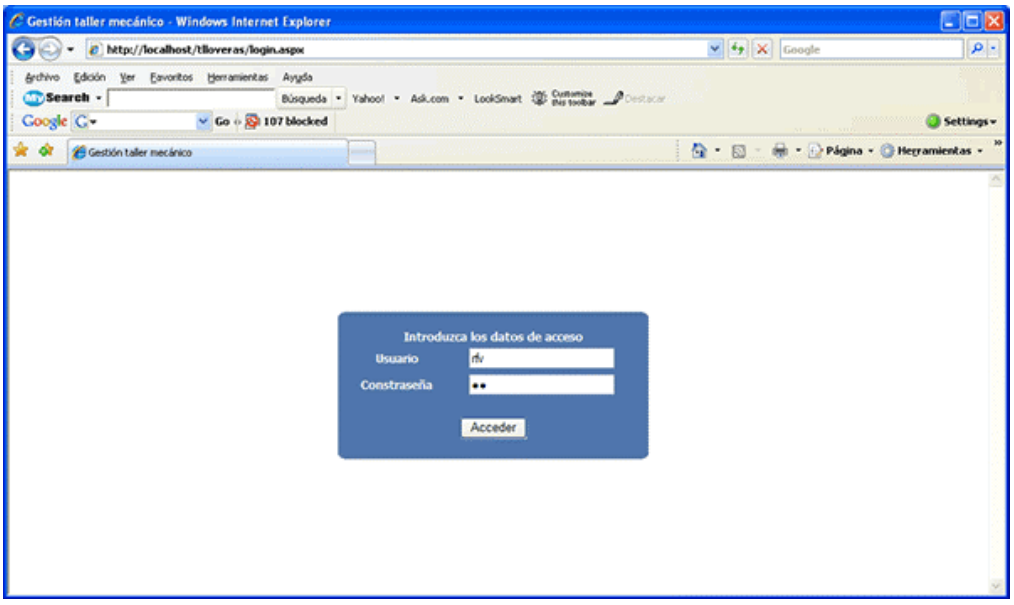


Figura 15: Pàgina d'identificació

4.4.2. Gestió de clients i vehicles

Com podem imaginar, gestionar els treballs a realitzar als vehicles dels clients implicarà poder gestionar també els propis clients i vehicles associats, necessitats a les que dóna resposta aquest bloc funcional.

La gestió de clients i vehicles constitueix la primera pestanya de l'aplicació, i permetrà a l'usuari (recepció o cap de taller) localitzar clients existents per tal de consultar / modificar les seves dades i vehicles. A més, permetrà a l'usuari crear nous clients i associar a aquests els vehicles que els pertoquin.

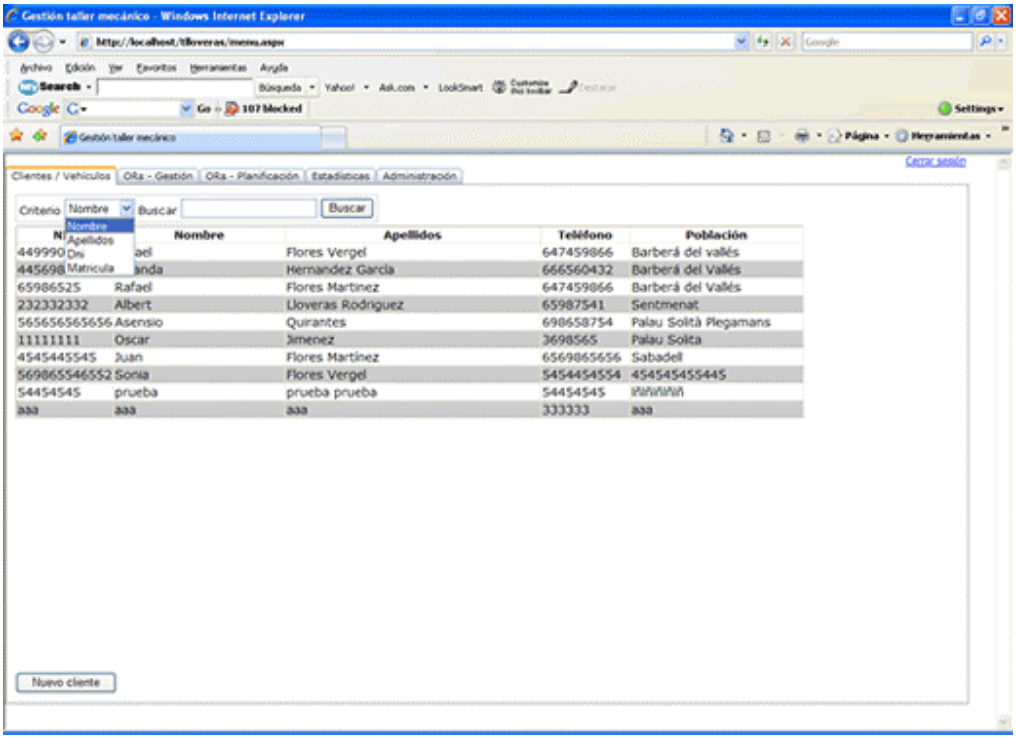


Figura 16: Buscador clients

En la figura 16 veiem la pàgina que permetrà localitzar clients segons diversos criteris (*Nombre, Apellidos, DNI, Matrícula)*. D'aquesta manera tant podrem accedir a les dades pròpies dels clients com a les dels seus vehicles. Per tant, aquest buscador constituirà el punt de partida per veure el detall del client i modificar les dades que convingui, tal i com es veure a la figura 17. En aquest punt és important tenir en compte que l'aplicatiu ens força a complimentar els camps obligatoris, remarcats convenientment amb un estil diferent, tal i com vam comentar en un apartat anterior dedicat a les interfícies d'usuari.

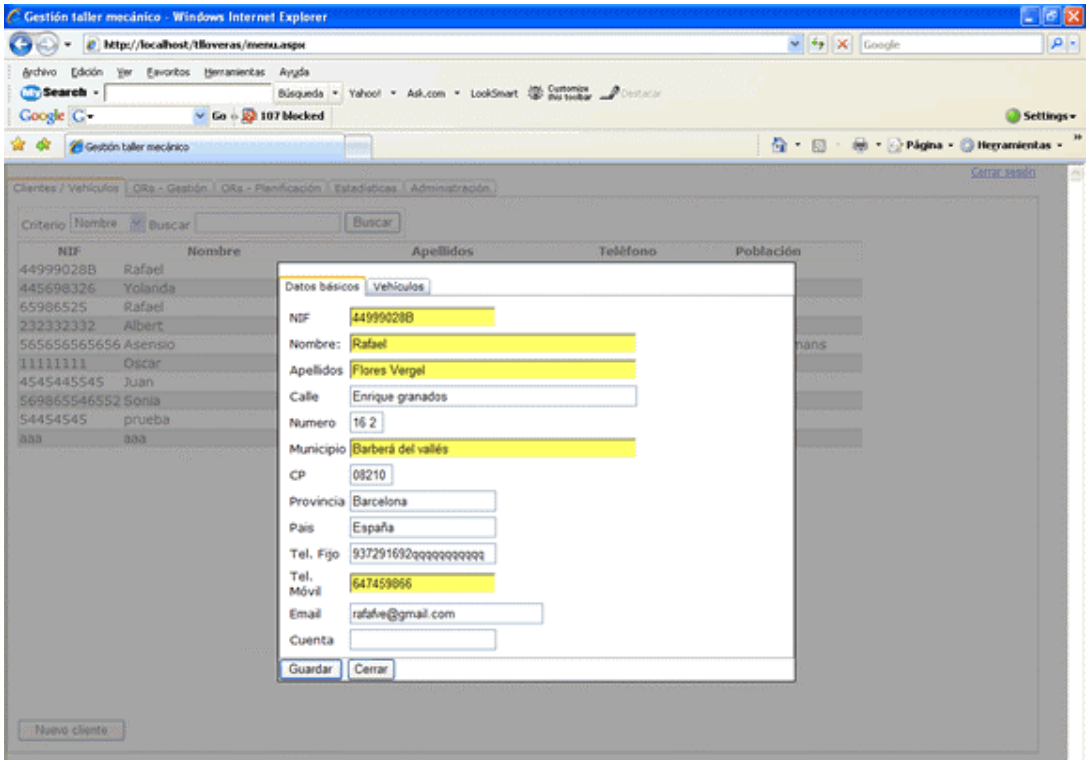


Figura 17: Detall de client

A la mateixa pantalla, encara que a una pestanya diferent (figura 18), podrem consultar els vehicles del client, i modificar o afegir els que creiem convenient. Per portar-ho a terme, es facilita la llista de vehicles que pertanyen al client, permetent-nos seleccionar aquell del que volem veure el detall (figura 19). D'altre banda, en el cas d'haver clickat el botó *Nuevo vehículo*, es mostraran tots els camps del detall buits, sense informar, per tal que l'usuari ho complimenti amb les dades del nou vehicle. Tant en un cas com en l'altre, en tancar el diàleg, ja sigui fent click a *Guardar* o a *Cerrar*, es tornarà a la llista de vehicles, actualitzant-se si escau.

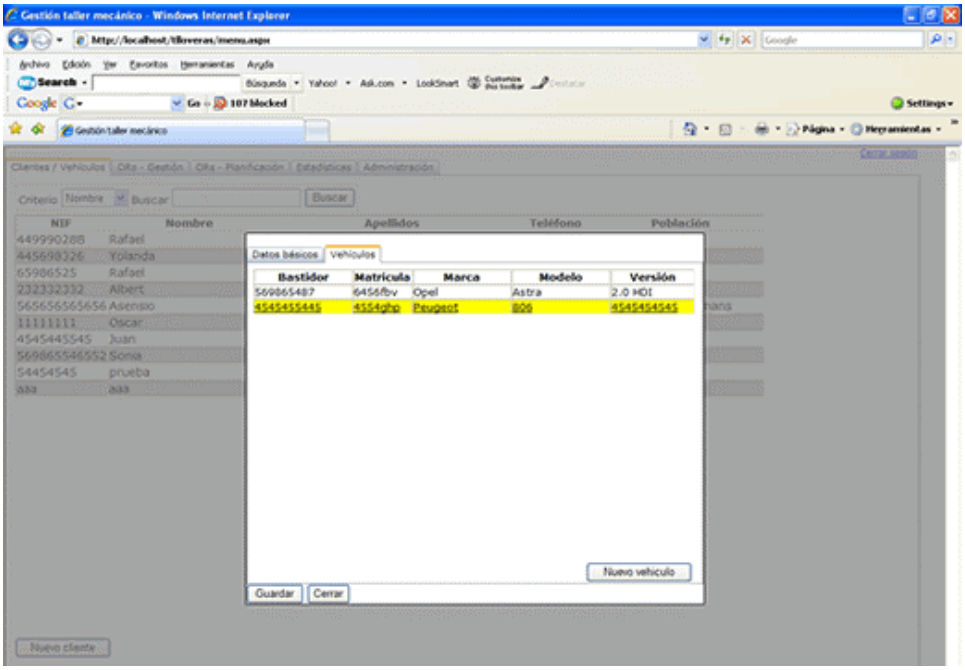


Figura 18: Relació de vehicles associats al client

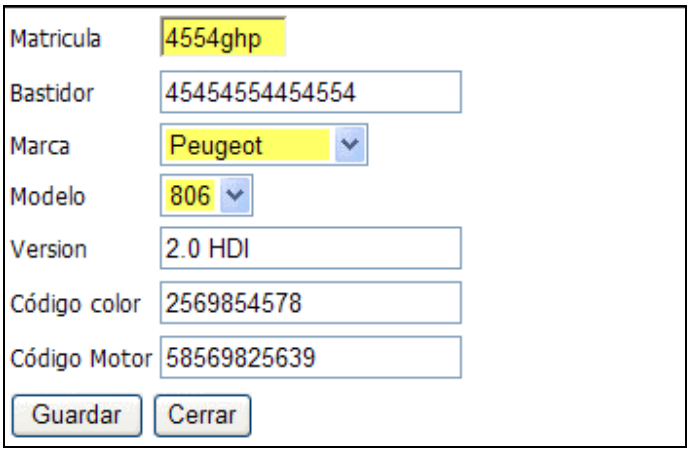


Figura 19: Detall de vehicle

Un aspecte important d'aquesta pantalla de detall de vehicle que es mostra a la figura 19 és la càrrega dinàmica dels models, en funció de la marca escollida, mitjançant la crida al Servei Web que ens proveeix la informació, utilitzant tècniques AJAX, de manera transparent a l'usuari.

4.4.3. Gestió d'Ors

Aquest bloc permet a l'usuari (recepció / cap de taller / mecànic) localitzar ORs per tal de visualitzar o modificar el seu contingut, així com crear noves (excepte mecànic), i és accessible des de la pàgina principal de l'aplicació, a la segona pestanya.

A la figura 20 podem veure la part principal d'aquest bloc, el buscador d'ORs, mitjançant el qual podem localitzar ORs segons diversos criteris: *Nombre, Apellidos, Matrícula i DNI.* Així, un cop localitzada la OR desitjada, al clickar sobre ella, podrem visualitzar el seu detall mitjançant la pantalla que es mostra a la figura 21, que també serà la utilitzada per crear noves ORs, al clickar *Nueva OR*.

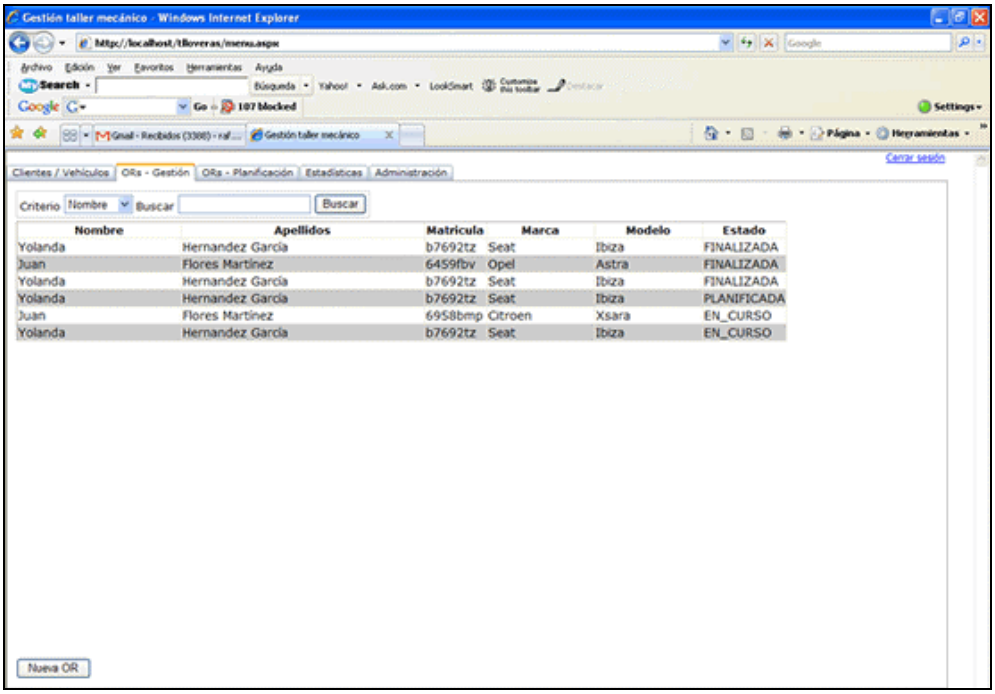


Figura 20: Gestió ORs

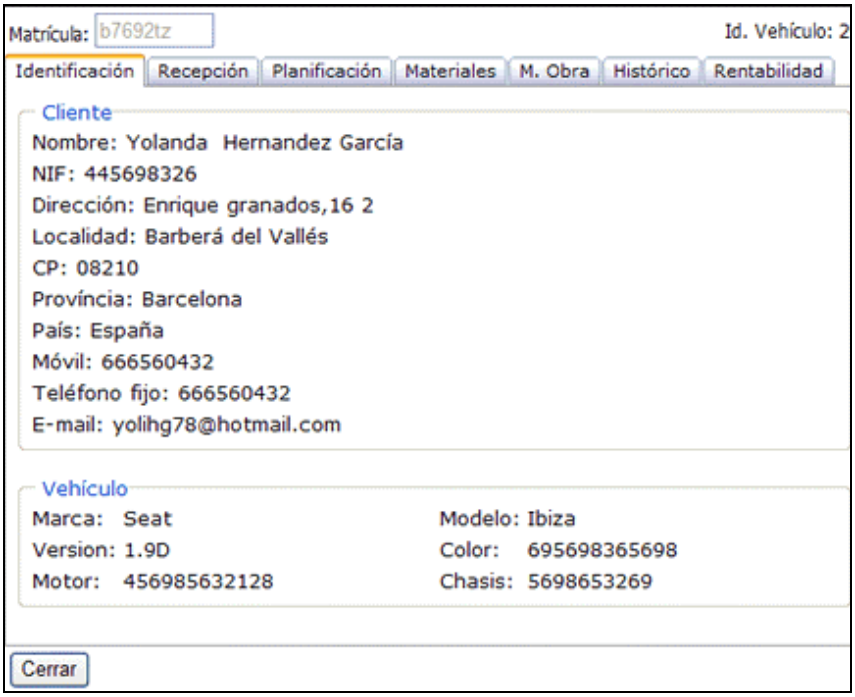


Figura 21: Detall OR

Com podem veure a la figura 21, la primera dada que formarà part del detall de la OR és la matrícula del vehicle. En cas d'estar-se visualitzant una OR ja existent, no es permet modificar, en canvi, pel cas d'una nova OR, a més de tractar-se d'un camp editable, s'ofereix ajuda per la seva cumplimentació. És a dir, conforme l'usuari va escrivint, se li mostra una llista amb les possibles matricules que comencen pels caràcters introduïts, de manera que l'usuari simplement ha d'escollir una de les diverses possibilitats. Per portar a terme aquesta funcionalitat, es torna a fer ús d'un control avançat d'AJAX, el conegut com *Autocomplete*, que realitzarà crides al Servei Web corresponent.

Altre aspecte que podem veure a la figura 21 és que la informació de detall d'OR s'ha dividit en diversos blocs d'informació, que es corresponen a diferents pestanyes de la pantalla:

– *Identificación:* Dades bàsiques d'identificació del client i vehicle al que afecta la OR. En cas de tractar-se d'una alta o creació d'OR, els camps es mostraran en blanc, i s'informaran de manera automàtica en el moment en que l'usuari informi la matrícula.

- *Recepción:* Dades referents a la recepció del vehicle. Dos camps a destacar en aquest bloc: *Vehículo Recepcionado* (en funció del seu valor una OR podrà o no ser iniciada) i *Descripción* (inclou un text descriptiu dels treballs a realitzar). A més, pels camps que contenen dates habilitarem controls de tipus calendari, per tal que l'usuari pugui escollir còmodament la data. Concretament, en aquesta pestanya s'utilitzaran per la data d'entrada i sortida del vehicle. El detall es pot veure a la figura 22.

- *Planificación*: Informació relativa a la planificació de la OR. Aquestes dades marquen la data i hora en que els operaris han de realitzar la OR, així com una descripció més acurada dels treballs a realitzar, juntament amb el temps previst, en base al qual es calcula l'hora prevista de finalització. Podem veure el detall d'aquesta pestanya a la figura 27.

- *Materiales:* Conté el desglòs de materials que s'han fet servir a la OR. Per simplificar el procés d'incorporar els materials, es permet seleccionar entre tots els disponibles, com es pot observar a la figura 23.

- *M. Obra:* Conté el desglòs de les operacions o tasques incloses en el treball així com el temps dedicat a aquestes. De la mateixa manera que amb els materials, existeix una llista d'operacions, d'entre les quals l'usuari ha d'escollir les portades a terme, com es pot observar a la figura 24.

- *Histórico:* Qualsevol acció realitzada sobre la OR i que impliqui un canvi d'estat, és enregistrada al històric amb la data en que s'ha produït, així com la persona que la dut a terme.

- *Rentabilidad:* Es tracta de dades únicament disponibles per aquelles ORs que estan finalitzades, i consisteix en un resum en quant a temps, per veure possibles desviacions respecte a les previsions inicials. Així, com es veu a la figura 25, es mostra el temps dedicat real (calculat pel propi aplicatiu en base a l'estat de la OR), així com el temps indicat per l'operari per cadascuna de les operacions, amb la diferència respecte a l'estàndard. A més, es mostra la diferència entre el temps dedicat i el planificat.

Com podem imaginar, el contingut i mode (lectura o edició) dels camps de cadascuna d'aquests pestanyes així com les accions disponibles (iniciar, parar o

finalitzar) dependran de l'estat de la OR així com de l'usuari connectat. En aquest sentit és important destacar alguns dels controls que porta a terme l'aplicatiu:

1. Una OR no es pot iniciar si el vehicle no s'ha recepcionat.
2. D'una OR que està en estat EN CURSO, no es poden modificar les seves dades tret de materials, operacions i comentaris.
3. Una OR no pot ser finalitzada si no s'han imputat els materials utilitzats.

A més, en cas de tractar-se d'una nova OR, s'hauran de complimentar els camps de cada pestanya marcats com obligatoris.

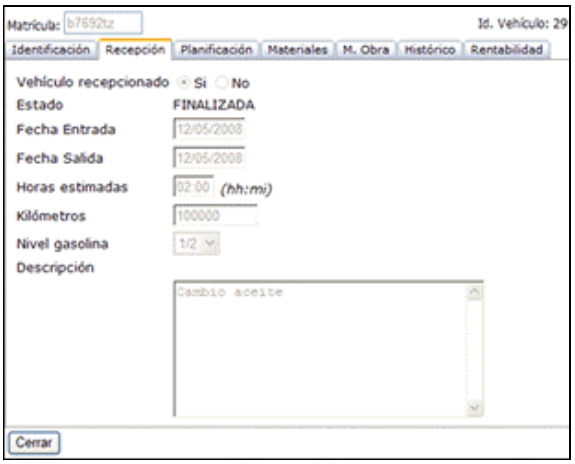


Figura 22: Detall de OR – Recepció Vehicle

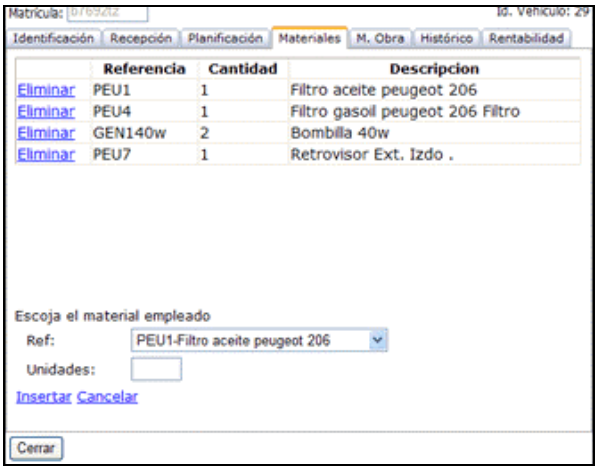


Figura 23: Detall de OR - Materials

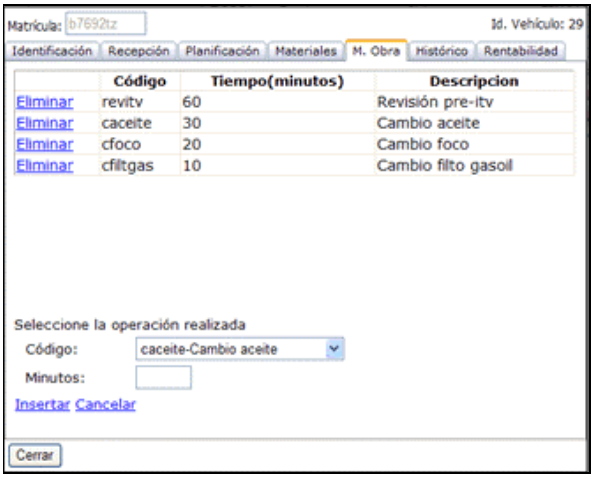


Figura 24: Detall OR – Ma d'obra

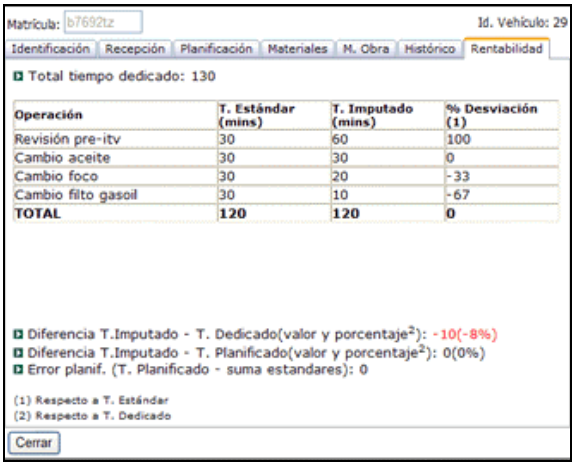


Figura 25: Detall OR - Rentabilitat

4.4.4. Planificació d'ORs

Aquest bloc és un dels que, des del punt de vista funcional, aporta més a l'usuari. Es troba disponible a la tercera pestanya i té l'aparença que podem veure a la figura 26.

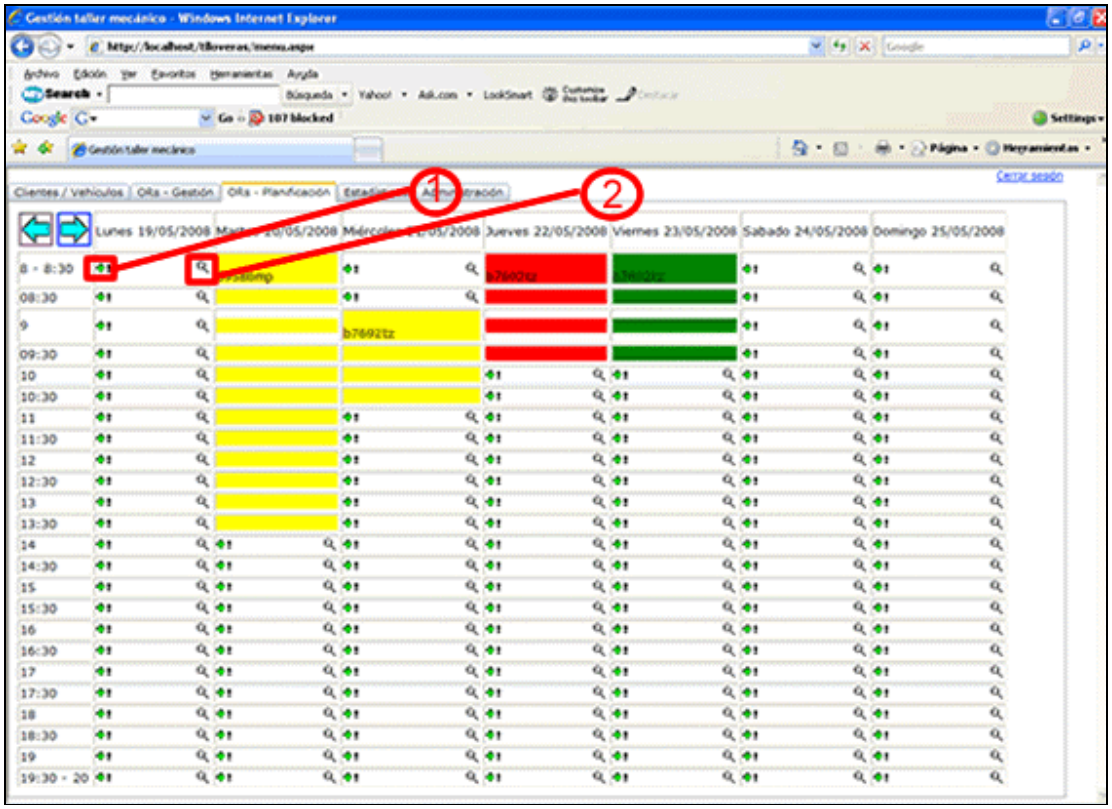


Figura 26: Eina de planificació d'ORs

Les accions que es poden portar a terme des d'aquesta pantalla són:

- Crear OR com a planificada: Quan un usuari, de tipus cap de taller, vol crear una nova OR i que aquesta quedi planificada per una data i hora concreta, clickarà a la casella corresponent al dia i hora d'inici, sobre la icona marcada amb (1), de manera que se li obrirà la pantalla de detall d'OR, amb els valors de data i hora d'inici informats, com es pot veure a la figura 27. A partir d'aquí, l'usuari ha de complimentar la resta de camps obligatoris que consten a la pestanya *Planificación*, a més de la matrícula del vehicle. En el moment que l'usuari guardi la OR i tanqui el detall, s'actualitzarà el calendari per incloure la nova OR, sempre i quan estiguin disponibles les hores sol·licitades, i no

sobrepassin de la hora de fi de jornada laboral. En cas contrari, es mostrarà un avís d'error.

- Planificar una OR ja existent: Una altra possibilitat de planificació d'ORs és que aquestes ja hagin estat prèviament creades, des de la pestanya *Gestión de ORs.* En aquest cas, per tal de planificar-les en una data/hora concreta, s'haurà de clickar la icona (2) indicada en la figura 26, de la casella horària que pertoqui. D'aquesta manera, es mostrarà la pantalla que es pot veure a la figura 28, que ens permetrà escollir la OR que desitgem planificar. A partir d'aquí, només faltarà complimentar la informació de la pestanya *Planificación,* i en cas de ser tot correcte quedarà convenientment planificada, actualitzant-se el calendari.

- Visualitzar planificacions existents i accés al detall d'aquestes: El calendari que es visualitza ens permet fer-nos fàcilment a la idea de l'ocupació del taller, per les diferents setmanes, així com de l'estat de les ORs, ja que depenent d'aquest es mostren al calendari en un color diferent. A més, ens permet accedir al detall d'una OR, clickant a sobre d'aquesta, podent consultar les seves dades, modificar-les o realitzar accions tals com iniciar-la, parar-la o finalitzar-la, segons el cas.

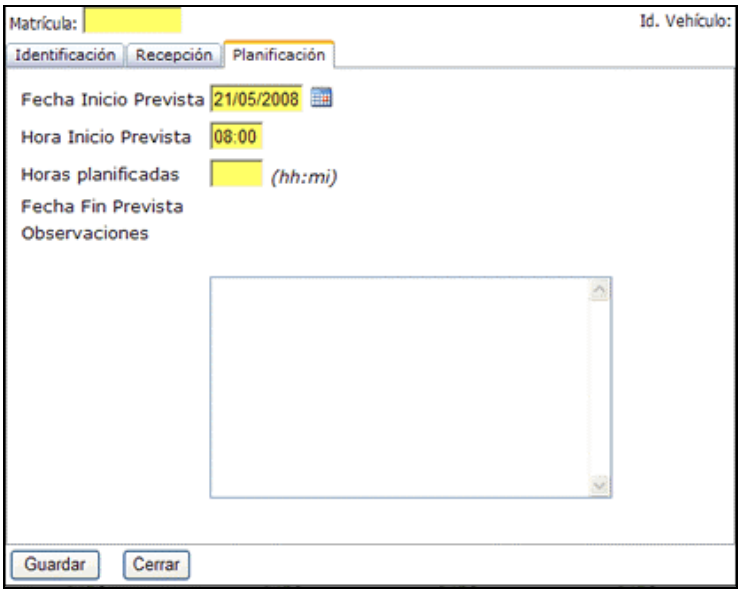


Figura 27: Detall OR - Planificació

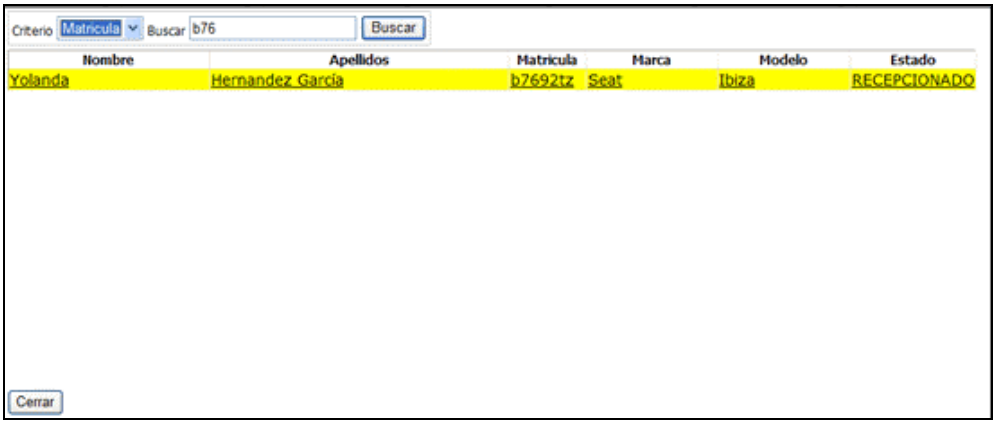


Figura 28: Buscador d'ORs pendents planificació

Tot i això, la disponibilitat de les funcionalitats que aquesta pestanya ofereix, dependrà de l'usuari identificat. Així, per exemple, si es tracta d'un mecànic, aquest no podrà crear noves ORs o planificar-ne, però sí visualitzar les planificacions existents.

4.4.5. Estadístiques

Aquest bloc constitueix la quarta pestanya de la pàgina principal i proveeix a l'usuari (recepció o cap de taller) informació global de les ORs ja finalitzades. Es refereix a informació bàsica, que pot ser limitada a un conjunt d'ORs basant-nos en criteris com la data d'inici, data de final, matrícula o client, tal i com podem veure a la figura 29. D'aquesta manera, podem saber, per exemple, per un client determinat (especificant el DNI), quantes ORs han entrat directament com planificades, quantes han estat introduïdes des de recepció, quin ha estat el promig de diferència entre el temps planificat i el dedicat, el promig de diferència entre el temps imputat i el dedicat real així com el temps promig per portar-les a terme. Es tracta de dades molt bàsiques però a la vegada útils que ajuden a avaluar el funcionament del taller.

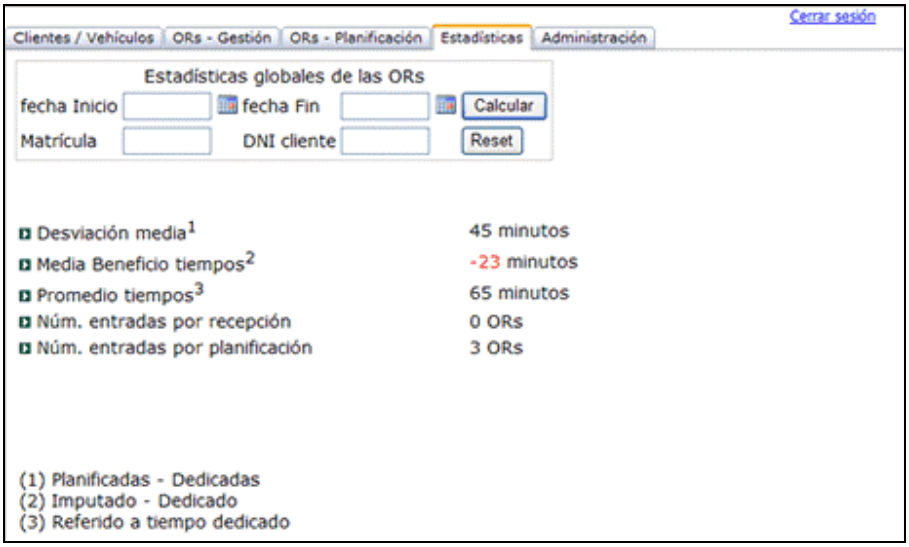


Figura 29: Estadístiques ORs

4.4.6. Administració

Des d'aquest bloc que es troba disponible a la cinquena i última pestanya, el personal de recepció i el cap de taller poden administrar qualsevol dels aspectes que són configurables a l'aplicatiu. Es tracta de poder administrar els usuaris que tenen accés a l'aplicació, les operacions o tasques que es poden portar a terme com a part d'una OR i els materials que les ORs poden incloure, així com les marques i models de vehicles disponibles a l'aplicatiu. El detall d'aquesta pestanya el podem veure a la figura 30 i hi podem observar que es divideix en tres parts ben diferenciades, corresponents als tres elements principals que es permet administrar (usuaris, operacions i materials), facilitant-se els corresponents buscadors per tal de localitzar l'element a editar, mitjançant la corresponent pantalla de detall. Com element comú entre les tres seccions ja comentades, s'inclou un botó per donar d'alta usuaris, operacions i materials, respectivament.

A la figura 31 podem veure quina informació es mostra com detall d'un usuari, i per tant, pot ser editada. Com es veu, a part del nom i cognoms, s'emmagatzema informació referent a l'accés, com paraula de pas i contrasenya així com el rol d'aquest al taller. A més, es disposa de la possibilitat de deshabilitar a l'usuari, de manera que no tingui accés a l'aplicatiu.

A la figura 32 veiem la pantalla que farem servir per editar el detall de les operacions que poden ser incloses a una OR. Com es veu, a més del codi que identifica la operació, comptem amb una descripció de la mateixa així com del

temps estàndard per portar-la a terme, i que per tant s'hauria d'aproximar al temps real que dediquen els operaris.

Els materials que es poden incloure a una OR són altre element que podrem administrar, mitjançant la pantalla de detall que mostrem a la figura 33.

L'últim element que es permet administrar són les marques i models de vehicles. Aquesta administració es realitza de manera diferent als casos anteriorment comentats. No es realitza una modificació element a element sinó que es realitza mitjançant la càrrega d'un fitxer XML, que contindrà la totalitat de marques de vehicles disponibles així com de models. Aquesta càrrega es portarà a terme clickant *Importar modelos / marcas de vehículos* de la pestanya administració, provocant que es mostri el diàleg de la figura 34, on simplement s'haurà d'escollir la ubicació de l'arxiu XML a carregar.

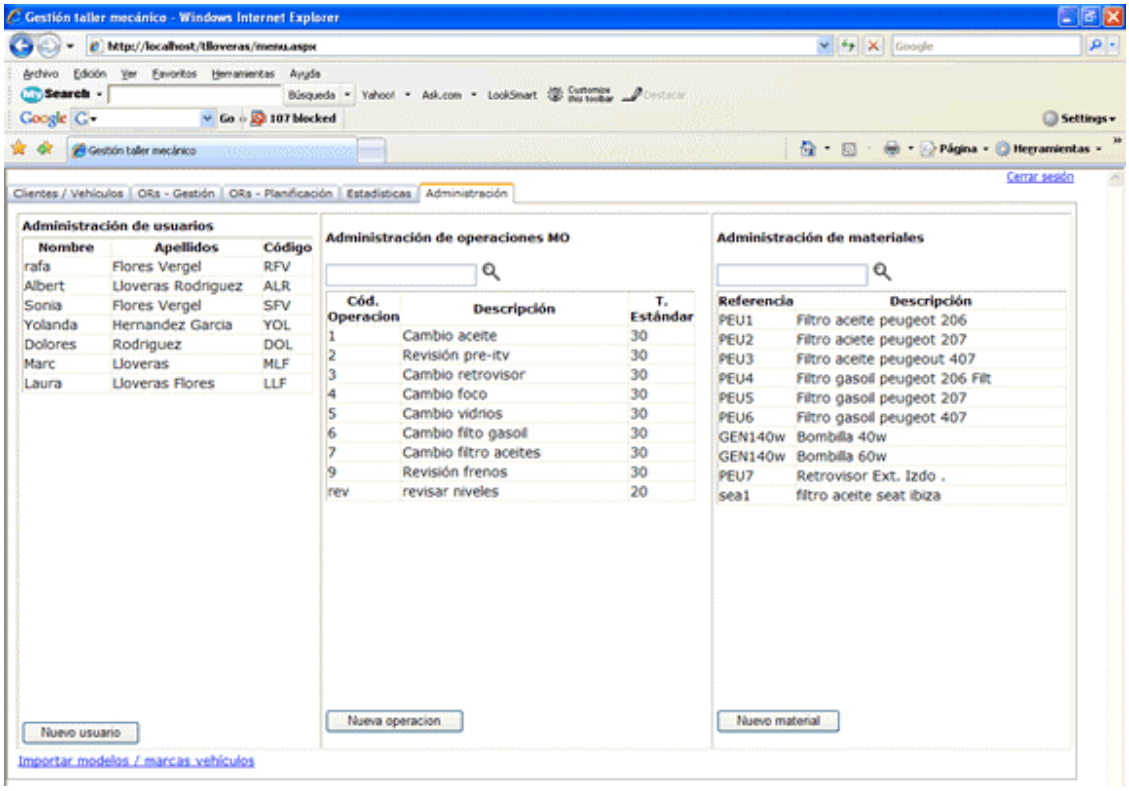


Figura 30: Pantalla d'administració

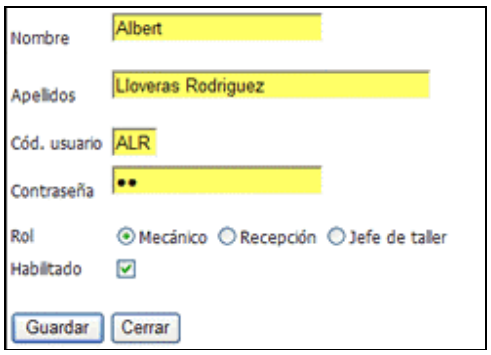


Figura 31: Detall d'usuari

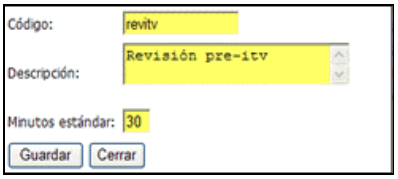


Figura 32: Detall d'operació

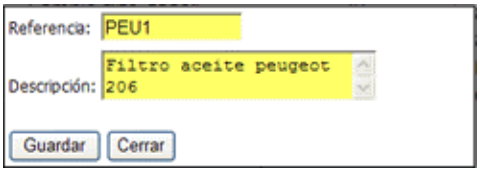


Figura 33: Detall de material

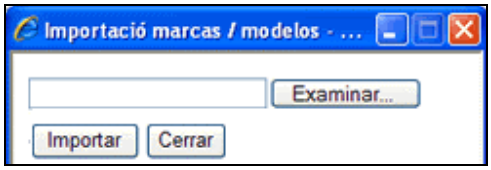


Figura 34: Importació fitxer XML

El motiu pel que he realitzat aquesta administració de manera diferent als anteriors casos és que considero que les marques i models disponibles a l'aplicació haurien de ser tots els existents al mercat, i s'haurien d'obtenir d'una font externa que estigui actualitzada. Per tant, la idea original era utilitzar algun Servei Web disponible de manera gratuïta que ens facilités aquesta informació, i que pogués ser consulta de manera automàtica per la nostra aplicació, mantenint sempre les dades actualitzades, amb les últimes marques i models apareguts al mercat, sense requerir cap acció de l'usuari. Tot i això, no hem trobat cap servei d'aquestes característiques, pel que la solució intermitja adoptada és mantenir un fitxer XML amb tota la informació (la podem obtenir de qualsevol revista especialitzada del món de l'automòbil). Aquest s'haurà de revisar periòdicament, actualitzant-se en cas de ser necessari, i carregant-lo a l'aplicació convenientment. És important destacar dos aspectes que també han ajudat a creure que aquesta resulta la millor manera de realitzar aquest manteniment, sense necessitat d'interfície d'usuari:

1. Les marques i models mai s'esborren, tot i que es deixin de fabricar, ja que els nostres clients poden tenir un vehicle d'alguna d'aquestes marques o models.
2. Les marques i models no canvien la seva denominació.

A més, si en un futur disposem d'una font d'informació com l'anteriorment comentada, la seva integració resultarà més senzilla ja que la pantalla d'administració no requerirà de modificacions.

4.5. Servei Web

Com ja hem anat comentant al llarg d'aquest document, en aquest aplicatiu es fa un ús intensiu de Serveis Web, degut a la capacitat que ens dóna per millorar la experiència de l'usuari en l'ús diari de l'aplicació. Així, per exemple, seguint amb les idees ja explicades a l'apartat corresponent a AJAX, podrem cridar a aquests Web Services des del client, sense necessitat d'enviar tota la pàgina al servidor, i refrescant sols el que convingui d'aquesta amb la informació retornada pel servidor, mitjançant la resposta del Servei Web.

Altre aspecte pel que ja vam comentar que resultava molt interessant l'ús dels Serveis Web és que permet la fàcil interconnexió entre sistemes i mòduls d'aquests. En el nostre cas concret, vam comentar que afavoria l'escalabilitat, donat que podríem integrar nous mòduls funcionals que obtindrien fàcilment informació del sistema mitjançant crides als Servei Web de que disposa.

A l'aplicatiu s'han desenvolupat dos Serveis Web:

- *SvcModelosVehiculo*: Servei Web auxiliar necessari pel correcte funcionament d'algun dels controls específics de la llibreria AJAX utilitzada.

- *svcGestion*: Aquest Servei Web engloba totes aquelles funcionalitats que es duen a terme al servidor, però que són invocades des de la part client del nostre aplicatiu (codi JavaScript), emprant AJAX. A més, aquestes funcionalitats podran ser consumides per qualsevol altre mòdul o aplicatiu capaç de consumir Serveis Web.

Els Web Services, a més de composar-se de mètodes (webmethods), que implementen cadascuna de les funcionalitats que aquests ofereixen, poden contenir altre tipus de codi, com definició de classes. En el nostre cas, s'han definit

una sèrie de classes que faciliten el retorn dels diferents mètodes, utilitzant objectes d'aquestes. Així doncs, s'han definit les classes infoCliente, infoVehiculo, infoOR, infoPlanificacionOR, infoVehiculoRed que contenen informació de client, vehicle, OR, planificació d'OR i detalls bàsics de vehicle, respectivament. A més, aquestes classes contenen informació de control que ens permet comprovar si l'execució dels mètodes ha funcionat correctament o, pel contrari, s'ha produït algun error.

4.5.1 Resum de mètodes implementats

A la següent taula mostrem un resum de tots els mètodes implementats al principal Web Service de l'aplicació, que anteriorment hem comentat.

Nom	Paràmetres	Retorn
recuperarDatosCliente	IdCliente:Integer	InfoCliente
Recupera informació del client indicat per paràmetre.		
recuperarInfoVehiculo	idVehiculo:Integer	InfoVehiculo
Recupera informació del vehicle indicat per paràmetre.		
recuperarInfoOR	idVehiculo:Integer idOR:Integer	InfoOR
Recupera informació de la OR indicada per paràmetre.		
insertarOR	IdVehiculo:Integer recepcionada:Boolean estado:String fechaEntrada:String fechaSalida:String horasEstimado:Integer numKms:Integer codGasolina:String descripcion:String	InfoPlanificacionOR
Inserta una nova OR, segons les dades facilitades per paràmetre.		
actualizarOR	idOR:Integer idVehiculo:Integer recepcionada:Boolean estado:String fechaEntrada:String fechaSalida:String horasEstimado:Integer numKms:Integer codGasolina:String descripcion:String	InfoPlanificacionOR
Actualitza la OR que correspon al paràmetre facilitat, amb la informació corresponent.		
actualizarCliente	idCliente:Integer nombreCliente:String apellidosCliente:String direccionCliente:String	Integer

	provinciaCliente:String paisCliente:String nifCliente:String telFijoCliente:String telMovilCliente:String emailCliente:String municipioCliente:string CPCliente:String cuentaCliente:String numeroCliente:String	
Actualitza les dades del client, segons la informació facilitada per paràmetre.		
insertarVehiculo	idCliente:Integer idMarca:Integer idModelo:Integer version:String matricula:String bastidor:String codColor:String codMotor:String	Integer
Insertar un nou vehicle pel client indicat.		
actualizarVehiculo	idVehiculo:Integer idMarca:Integer idModelo:Integer version:String matricula:String bastidor:String codColor:String codMotor:String	Integer
Actualitza les dades d'un vehicle concret.		
actualizarORPlanificada	idOR:Integer idVehiculo:Integer recepcionada:Boolean estado:String fechaEntrada:String fechaSalida:String horasEstimado:Integer numKms:Integer codGasolina:String descripcion:String fechaInicioPrevisto:String fechaFinPrevisto:String horasPlanificadas:Integer idusuario:Integer	InfoPlanificacionOR
Actualitza la OR ja planificada, que correspon al paràmetre facilitat, amb la informació corresponent.		
recuperarListadoORsPlanificadas	fechaInicioIntervalo:String fechaFinIntervalo:String	ArrayList
Recupera relació d'ORs planificades, segons l'interval indicat.		
horasDisponiblesExistente	idOR:Integer fechaInicioIntervalo:String fechaFinIntervalo:String	Boolean

Comprova si l'interval indicat està disponible (no ocupat o ocupat per la OR que intentem planificar en el moment de fer la crida).		
iniciarOR	idOR:Integer idUsuario:integer	(cap)
Inicia la OR i ho registra a l'històric.		
finalizarOR	idOR:Integer idUsuario:integer	(cap)
Finalitza la OR i ho registra a l'històric.		
pararOR	idOR:Integer idUsuario:integer	(cap)
Para la OR i ho registra a l'històric.		
horasDisponibles	fechaInicioIntervalo String fechaFinIntervalo:String	Boolean
Comprova si l'interval facilitat per paràmetre està o no ocupat per alguna OR.		
recuperarEstadisticasGlobales	dtFechaInicio:Date dtFechaFin:Date sMatricula:String sDni:String	estadisticaOR
Recupera les estadístiques globals de les ORs finalitzades, limitant el càlcul segons el valor dels paràmetres.		
insertarORPlanificada	idVehiculo:Integer recepcionada:Boolean estado:String fechaEntrada:String fechaSalida:String horasEstimado:Integer numKms:Integer codGasolina:String descripcion:String fechaInicioPrevisto:String fechaFinPrevisto:String horasPlanificadas:Integer	InfoPlanificacionOR
Inserta i planifica una nova OR.		
insertarCliente	nombreCliente:String apellidosCliente:String direccionCliente:String provinciaCliente:String paisCliente:String nifCliente:String telFijoCliente:String telMovilCliente:String emailCliente:String municipioCliente:String CPCliente:String cuentaCliente:String numeroCliente:String	Integer
Inserta un nou client.		
RecuperaInfoUsuario	idUsuario: integer	Usuario
Recupera el detall d'un usuari concret.		

ActualizarUsuario	IdUsuario:Integer nombre:String apellidos:String codigo:String rol:String password:String	(cap)
Actualitza la informació d'un usuari.		
insertarUsuario	nombre:String apellidos:String codigo:String rol:String password:String	Integer
Inserta un nou usuari i retorna l'identificador d'aquest.		
ActualizarOperacion	idOperacion:Integer codigo:String descripcion:String minutos:Integer	(cap)
Actualitza la informació d'una operació predefinida.		
insertarOperacion	codigo:String descripcion:String minutos:Integer	(cap)
Inserta una nova operació.		
ActualizarMaterial	idMaterial:Integer referencia:String descripcion:String	(cap)
Actualitza el detall d'un material concret dels disponibles.		
InsertarMaterial	referencia:String descripcion:String	(cap)
Inserta un nou material.		
recuperarInfoOperacion	idOperacion:Integer	Operacion
Retorna el detall d'una operació predefinida.		
recuperarInfoMaterial	idMaterial:Integer	Material
Retorna el detall d'un material, segons el valor facilitat per paràmetre.		

5. Proves

En aquest capítol explicarem alguns conceptes bàsics relatius a la prova del software que ens facilitaran la comprensió del detall de les proves realitzades al nostre aplicatiu, que també seran àmpliament explicades.

5.1. Introducció

La prova és la mesura més important per al control de qualitat emprada durant el procés de desenvolupament. La funció bàsica de les proves és la detecció d'errors del software, que es poden haver introduït en qualsevol de les fases precedents (anàlisi, disseny o codificació).

La importància de la fase de proves ve determinada per:

- És impossible desenvolupar un software sense errors, durant l'etapa d'especificació, disseny o codificació.
- El cost dels errors és molt gran; la prova representa un 40% de l'esforç de desenvolupament.

És important tenir en compte que la prova no implica directament la qualitat del software, és a dir, si el software era de mala qualitat abans de començar la fase de proves, després d'aquesta, continuarà sent-ho. Això és així ja que la qualitat s'incorpora al software durant el procés d'enginyeria del software i no durant la fase de prova. Com que l'objectiu de la prova és descobrir errors, direm que té èxit si troba errors i per tant un cas de prova és bo si implica una alta probabilitat de detectar un nou error.

Com és lògic, en qualsevol projecte existeix un conflicte inherent d'interessos quan comença l'etapa de prova: es demana a la persona que ha desenvolupat l'aplicació que la provi, però aquesta persona té un gran interès per demostrar l'absència d'errors a l'aplicació i que l'aplicació funciona d'acord amb les especificacions del client. Per tant, es pot fer necessària la prova per part d'un grup independent de proves (GIP), l'objectiu del qual és trobar els errors. Tot i això, la persona que ha desenvolupat l'aplicació treballarà conjuntament amb el GIP per assegurar que es realitzen proves exhaustives i corregir els errors que es van descobrint.

Bàsicament hi ha dues possibilitats d'enfocament de les proves: proves de caixa blanca i proves de caixa negra. Les primeres es basen en el coneixement de la funció específica per la que es va dissenyar el software, i intenten demostrar que cada funció és totalment operativa, buscant a la mateixa vegada errors per cada funció que implementa. Sembla, per tant, que una prova de caixa blanca molt profunda ens portaria a obtenir software 100 % correcte. Per portar-la a terme hauríem de definir tots els camins lògics, dissenyar casos de prova que els executin i avaluar els resultats. A la pràctica aquesta prova exhaustiva és impossible degut a la complexitat que implica la recerca i execució de tots els camins lògics possibles. Per tant, hem d'escollir i executar els camins més importants o suficients per cobrir totes les sentències.
En el nostre projecte dissenyarem casos de prova el més complets possibles i això ens serà facilitat pel fet que l'aplicació està perfectament definida en mòduls que provarem per separat.

Per altra banda, les proves de caixa negra es basen en el coneixement del funcionament del software i intenten provar que totes les peces encaixen, centrant-se en el compliment dels requisits i ignorant l'estructura lògica interna de l'aplicació. Aquestes proves no són una alternativa a les proves de caixa blanca sinó que intenten trobar altres tipus d'errors diferents. Es busquen tècniques de prova de caixa negra que redueixin el nombre de casos de prova per obtenir una prova raonablement completa, i que denotin en general la presència d'errors a l'aplicació. Aquests tipus de prova s'adapten perfectament al projecte ja que s'han utilitzat per comprovar errors d'interfície, de navegació, d'exposició de resultats, etc... Passant, per exemple, per totes les pantalles de l'aplicació i comprovant que els resultats eren els esperats així com la seva disposició a pantalla.

Considerant les proves de caixa blanca i les de caixa negra a continuació comento molt breument els diferents nivells de prova depenent de l'element del software que es prova:

1. proves d'unitat: centra el procés de verificació en la menor unitat del disseny del software: un mòdul, una classe, …
2. proves d'integració: és una tècnica per construir el software a partir de diferents mòduls, a l'hora que es van portant a terme proves per detectar errors deguts a la seva interacció. Concretament es tracta d'una integració incremental.

3. proves del sistema: tracten de provar el software ja incorporat amb la resta d'elements que formen el sistema, ja siguin de hardware o de software. Es tracta de verificar que s'han integrat adequadament tots els elements del sistema i que aquest realitza les funcions apropiades.
4. proves d'acceptació: es realitzen per demostrar al client, amb dades i en temps real, l'operativitat del sistema.

5.2. Proves Realitzades

Tal i com vam comentar en la planificació, quan ens vàrem referir a les proves, aquestes s'han realitzat tant durant el desenvolupament, com en una fase posterior dedicada íntegrament a aquestes.

Al realitzar proves durant el desenvolupament, ens resultarà més senzill resoldre els errors que puguin sorgir (tenim més recent el coneixement de com s'ha implementat). A més, evitarem posposar la detecció d'errors greus que podrien implicar costoses modificacions a l'aplicació. Es tractaria doncs de les proves d'unitat anteriorment comentades.

A més, al dedicar tota una fase a les proves, quan s'hagi acabat el desenvolupament, s'afavoreix la realització de proves més generals, al no estar capficats en una funcionalitat concreta, de manera que podem validar el comportament global de l'aplicatiu. En aquest cas parlaríem de les proves d'integració, sistema i acceptació.

Per altra banda, tal i com hem vist anteriorment, les normes de desenvolupament de software en referència a les proves, recomanen que les proves no les faci únicament la persona que ha desenvolupat l'aplicatiu. Així doncs, les proves no les he portat únicament jo a terme sinó que altres persones també l'han provat, per validar el correcte funcionament i detectar possibles errors que per mi potser són inapreciables o que sorgeixen amb casos de prova que jo no havia tingut en compte. A més, com més persones proven una aplicació més possibilitats hi ha de trobar errors, i per tant que les proves tinguin èxit.

5.2.1. Proves del programador

Com a únic programador de l'aplicació, aquestes proves les he realitzat jo mateix, i tot seguit passo a resumir-les, ja que algunes d'elles ja s'han comentat més àmpliament al punt de la memòria on es parla del disseny funcional de l'aplicació:

- només tenen accés a l'aplicatiu els usuaris convenientment autenticats i sols estan autoritzats a veure aquella informació i a executar aquelles accions que els hi pertoqui en funció del seu rol.

- els usuaris autoritzats han de poder cercar clients i vehicles, i els resultats que es mostren són els que s'esperaven, segons els criteris utilitzats.

- els usuaris autoritzats han de poder editar els clients i vehicles, així com crear-ne de nous, i la informació mostrada de cadascun és la correcta, havent-se d'informar amb el format correcte els camps obligatoris.

- els usuaris autoritzats poden cercar ORs, i els resultats mostrats són els esperats, segons els criteris utilitzats.

- els usuaris autoritzats poden modificar les ORs, segons les restriccions d'estat, així com crear noves, i la informació de cadascuna de les ORs és correcta, validant-se que s'informen els camps obligatoris.

- els usuaris autoritzats poden planificar tant noves ORs com antigues, i la planificació té en compte de manera correcta limitacions tals com disponibilitat horària o final de jornada laboral.

- els usuaris que estan autoritzats, poden revisar i modificar les planificacions desplaçant-se convenientment per les diferents setmanes.

- els usuaris que estan autoritzats, poden realitzar les accions que els pertoqui sobre les ORs, com iniciar-les, parar-les o finalitzar-les i aquestes queden convenientment enregistrades a l'històric.

- la informació del detall de la OR es mostra convenientment, en funció de l'usuari que la intenta visualitzar, deshabilitant l'edició si convé.

- els usuaris que estan autoritzats, poden accedir a les estadístiques, i aquestes mostren els resultats esperats, segons els criteris esperats.

- els usuaris que estan autoritzats poden modificar qualsevol dels paràmetres que permet l'administració, i un cop han estat modificats, es mostren convenientment actualitzats a la resta de l'aplicatiu.

- la visualització de tots els elements és correcta, i l'estil de l'aplicació és homogeni.

5.2.2. Proves dels Usuaris

Com ja hem comentat, han estat diversos els usuaris que han provat l'aplicació, el qual és molt interessant donat que cadascun té un punt de vista diferent, i entén de manera diferent com hauria de ser el funcionament dels diferents blocs de l'aplicació, així com la navegació per les diferents pàgines. Per tant, gràcies a ells augmenten les possibilitats de que les nostres proves tinguin èxit, és a dir, de que trobin errors.

Les proves dels usuaris, com ja hem comentat, s'han dut a terme en el que vàrem comentar a la planificació com Bloc de proves, és a dir, un cop acabat el desenvolupament i realitzades les proves del programador, de manera que l'aplicació resultés el més fiable possible.

Els errors detectats han estat bàsicament alguna falta d'ortografia i validacions de dades, però cap d'ells d'especial importància. Considero que el més important que m'han aportat aquestes proves és una altre visió en quant a la ubicació i disseny dels elements a les diferents pàgines, modificacions que m'han portat a una navegació potser més lògica i que podria fer el treball diari dels usuaris més còmode. Tot i això, s'ha de dir que ha comportat únicament petits canvis, sense implicar cap tipus de desviació respecte a la planificació inicial.

Per últim, agrair a aquests usuaris, que bàsicament han estat familiars i companys de feina, el temps que m'han dedicat ja que gràcies a aquests he pogut perfeccionar l'aplicació amb petits detalls que han contribuït a millorar-la.

6. Conclusions

En aquest capítol faré una última reflexió respecte als coneixements que la realització del projecte m'ha permès posar en pràctica així com si s'han aconseguit els objectius definits inicialment i les possibles ampliacions i millores que es podrien realitzar.

6.1. Aplicació de Coneixements Adquirits a la Universitat

Durant aquests anys de carrera he adquirit una sèrie de coneixements, tant teòrics com pràctics que he pogut aplicar en molts casos a la realització d'aquest projecte.

He aplicat la base de la programació obtinguda amb la matèria d'Algorismes i Programació, els coneixements adquirits en Bases de Dades per dissenyar la meva pròpia base de dades així com coneixements de Xarxes, que m'han ajudat a entendre més fàcilment alguns conceptes relacionats amb el desenvolupament web. Tot i això, potser l'assignatura que ha tingut més aplicació ha estat Enginyeria del software ja que en ella s'aprèn a desenvolupar un software des de l'estudi de requeriments, fins a la planificació del manteniment d'aquest.

Cal dir, però, que al projecte he treballat amb tecnologies i conceptes que mai havia vist a la universitat, i que per tant, he hagut d'aprendre de manera autònoma, mitjançant autoaprenentatge. Val a dir, però, que això m'ha estat possible gràcies a que aquests anys de carrera m'han fet madurar intel·lectualment, fent-me molt més fàcil adquirir nous coneixements. Personalment crec que això m'ha aportat molt més que no pas coneixements específics, que amb el temps podrien quedar antiquats.

6.2. Objectius

Els objectius establerts per aquest projecte s'han complert, i tots els requeriments marcats des d'un principi s'han tractat i desenvolupat. Es pretenia fer una eina que millorés la gestió d'un taller mecànic, proveint-lo de mecanismes per planificar els treballs i fer-ne seguiment, així com per realitzar altres tipus de tasques, i així ha estat. A més, es pretenia que totes les tasques que es puguin dur a terme siguin fàcils d'utilitzar, aconseguint una aplicació usable, que permeti augmentar la productivitat dels usuaris en les seves tasques diàries.

Arribats a aquest punt, cal dir que s'han implementat detalls que en un principi no havien estat considerats com ara l'administració d'usuaris, que ajuden a tenir una eina més completa.

Referent a la interfície, comentar que partíem de la problemàtica que la creació d'una bona interfície implica més a un dissenyador gràfic que no pas a un programador. Tot i això, el procés que vàrem utilitzar per la creació de la mateixa, la construcció de prototipus, ens ha permès obtenir una implementació final eficient, així com d'una estètica satisfactòria, el qual no vol dir que sigui perfecta, sinó simplement que és coherent amb el nostre objectiu primordial, una interfície que ens faciliti l'eficiència i facilitat d'ús de l'aplicació.

Per últim comentar que a nivell personal em trobo satisfet amb el resultat obtingut, tant per la funcionalitat del sistema com per l'oportunitat que m'ha donat aquest d'endinsar-me en tecnologies que fins ara desconeixia, havent estat capaç de combinar la realització del projecte amb la meva feina.

Com a última conclusió, crec que tant el temps dedicat com la complexitat final de l'aplicació, responen satisfactòriament a les expectatives de la realització d'un projecte de final de carrera.

6.3. Ampliacions i Millores

Tot i que com ja hem comentat amb la realització del projecte s'han assolit tots els requeriments inicials, pensant en la implantació en un taller real, podrien resultar útils algunes funcionalitats addicionals. Es tracta en alguns casos de funcionalitats que donarien resposta a organitzacions d'un tamany major a l'inicialment considerat en aquest projecte, com consta a l'apartat on es defineix l'àmbit, així com funcionalitats que puguin complementar la solució, donant resposta a altres àmbits de gestió.

Concretament, les ampliacions que considero més adients són:

- Contemplar la possibilitat de que una organització tingui vàries sucursals. És a dir, dintre d'una mateixa organització, taller mecànic, hi podria haver un centre de treball que fos planxista, un altre electrònica, un altre mecànica general, etc...D'aquesta manera podrien existir varis calendaris,

un per cada centre. Donada la naturalesa web de l'aplicació, aquests centres podrien estar fins i tot distribuïts geogràficament, sense implicar gaires problemes.

- Contemplar la possibilitat de varis equips de treball dins d'un taller. D'aquesta manera, en una mateixa franja horària podrien coexistir diverses ORs, que serien assumides per diferents equips.

- Inclusió de nous mòduls funcionals com per exemple gestió de stock o mòdul de facturació. Ambdós mòduls podrien utilitzar el Servei Web desenvolupat per tal de recuperar la informació necessària, i en cas de requerir-ho, s'afegiria algun mètode addicional. En qualsevol cas, la inclusió d'aquest mòduls és totalment factible.

Bibliografia

- Bill Evjen, Scott Hanselman. "Professional ASP.NET 2.0", Wrox, 2007
- Francisco Charte Ojeda. "SQL Server 2000", Anaya Multimedia, 2001.
- Jorge Serrano Pérez. "Programación con ASP.NET", Anaya Multimedia, 2003.
- Matt Gibbs. "Professional ASP.NET 2.0 AJAX", Wrox Press, 2007

Pàgines web consultades:

- http://www.asp.net/ajax
- http://www.codeproject.com
- http://www.elguille.info
- http://www.4guysfromrolla.com
- http://www.sql-server-performance.com/articles/dba/stored_procedures_basics_p2.aspx
- http://technet.microsoft.com/es-es/library/ms203721.aspx
- http://www.desarrolloweb.com
- http://www.aspheute.com/english/
- http://www.w3schools.com
- http://p2p.wrox.com/forum.asp?FORUM_ID=96
- http://www.velocityreviews.com/forums/f29-asp-net.html
- http://www.webestilo.com/css/