

# **Aplicación de Gestión de Pruebas de decoración (PD)**

**Joan Català Bolet**

Ingeniería Técnica de Informática de Sistemas

**Cristina Pérez Solà**

11 de Enero de 2013

## **Dedicatoria**

Dedico este TFC a todos los estudiantes de la UOC, tanto a los que viven para aprender, como a los que aprenden para vivir.

## **Resumen**

La empresa Guala Closures se dedica a barnizar y litografiar hojas de aluminio para la fabricación de tapas para el sector alimentario. Esta empresa no dispone de un sistema de información digital para gestionar ni las solicitudes ni los diseños o pruebas de decoración (PD) que recibe y desarrolla para sus clientes.

Este proyecto quiere crear una aplicación sobre una plataforma web ya existente, que sea capaz de gestionar las solicitudes de las pruebas de decoración (PD) y que contenga todos los diseños de decoración que recibe y desarrolla para sus clientes. Una aplicación que nos permita la consulta, la modificación y la organización de los datos, controlando también el estado de desarrollo de las PDs para su seguimiento.

Este proyecto se desarrolla mediante software libre, para minimizar al máximo el coste del proyecto. Utilizaremos PHP como lenguaje de programación, frameworks XML para el diseño web, base de datos SQL y Apache como servidor web para la aplicación.

## **Indice**

CAPÍTULO 1: INTRODUCCIÓN.....	5
1.1 Justificación: punto de partida i aportación del TFC	
1.2 Contexto y descripción del problema	
1.3 Objetivos del TFC	
1.4 Enfoque y método seguido	
1.4.1 Metodología	
1.4.2 Herramientas	
1.5 Planificación del proyecto	
1.6 Breve descripción de los otros capítulos	
CAPÍTULO 2: ARQUITECTURA.....	12
CAPÍTULO 3: DISEÑO.....	13
3.1 Como construir la aplicación	
3.2 Estudio de la estructura y las funcionalidades de la aplicación	
3.3 Identificación clases y estructuras de la base de datos	
3.4 Diseño de metadatos con Eclipse	
3.5 Estructura XML	
3.6 Diseño pantallas	
CAPÍTULO 4: TECNOLOGIAS APLICADAS.....	35
4.1 Apache	
4.1.1 Como instalar el Servidor Apache	
4.1.2 Como configurar el Servidor Apache	
4.2 SQL Server	
4.3 Openbiz-Cubi Framework PHP	
4.3.1 Arquitectura Openbiz	
4.3.2 Módulos de Openbiz-Cubi	
CAPÍTULO 5: JUEGOS DE PRUEBAS.....	55
5.1 Prueba Crear solicitud PD	
5.2 Prueba para modificar una PD	
CAPÍTULO 6: CONCLUSIONES.....	60
CAPÍTULO 7: GLOSARIO.....	61
CAPÍTULO 8: BIBLIOGRAFIA.....	61

## 1. INTRODUCCIÓN

### 1.1 Justificación: punto de partida y aportación del TFC

La empresa Guala Closures se dedica a barnizar y litografiar hojas de aluminio para la fabricación de tapas para el sector alimentario. Los clientes envían un diseño a la empresa, y esta desarrolla una prueba de decoración, en adelante PD, como prototipo de cómo quedaría el producto final, y la devuelve al cliente para su aceptación.



Figura 1 – Tapas

Esta Prueba de decoración o PD, en su confección, es una ficha que contiene los datos del diseño, colores, proporciones de pintura y barniz utilizados para el desarrollo de la prueba. Esta información será utilizada para la producción en serie si el cliente da el visto bueno a la prueba enviada. Hay clientes que no solicitan la prueba física, por lo que con un modelo plano escaneado sería suficiente para dar o no el visto bueno a la prueba de decoración.

Actualmente la empresa no digitaliza estas pruebas de decoración. Los diseños o artes litográficos que se desarrollan para sus clientes, están en soporte físico y se plantea la necesidad de, además de conservar las muestras físicas del diseño que se envían a los clientes, digitalizar toda la información referente al desarrollo de las mismas, almacenar los modelos escaneados y poder compartir toda esta información con los usuarios implicados en el proceso productivo y también, para lo más importante, proteger la propia información del proceso productivo en caso de desastre.

Este proyecto quiere crear un módulo sobre una plataforma web, que contenga todos los diseños de decoración que recibe y desarrolla la empresa para su consulta, modificación y organización, controlando también el estado de las PDs, aprovechando bases de datos maestros ya existentes como clientes o artículos, etc.

### 1.2 Contexto y descripción del problema

Para contextualizar el proyecto, haremos una breve descripción a grandes rasgos del proceso productivo de una tapa para el sector de la alimentación. Partimos de hojas de aluminio que barnizamos y litografiamos como muestra la siguiente imagen:

Aluminio

Barniz interior

Decoración litográfica



Figura 2 – Hojas

Después la hoja de aluminio litografiada pasa por un proceso de embutición por prensa, donde la hoja toma progresivamente la forma cilíndrica.



Figura 3 – Embutición

Cuando la forma de la tapa ya es la definitiva, la tapa pasa a su decoración lateral, esta fase puede ser más o menos compleja ya que no solo incluye la decoración, sino también el relieve lateral (bordoneado) de la tapa. La última fase es la colocación del inserto plástico aislante dentro de la tapa. La siguiente imagen muestra la decoración lateral, bordoneado e inserción del inserto.



Figura 4 – Decoración lateral y Offset

El cliente final es quien aplica la tapa al envase definitivo mediante un proceso mecánico con unos cabezales.



Figura 5 – Producto acabado

El proceso productivo de la empresa parte siempre de estas pruebas de decoración (PD), que son las muestras prototipo que se envían a los clientes para su aprobación y también de dónde sacamos los datos necesarios para su fabricación en serie, establecer su coste y complejidad, componentes, etc.

El departamento de desarrollo de PDs, utilizaba hasta el momento unas cartulinas a doble cara con los datos de la prueba: una muestra de la chapa en plano con el diseño, datos referentes al proceso de fabricación litográfica y datos referentes a la decoración lateral de la tapa (Offset o Serigrafía).

Nº 15.598

PRUEBA DECORACIÓN

CLIENTE: RX SOUTH AFRICA

30 X 59.4

**Guala Closures Iberica**  
 C/df. 56-62 - Pol. Ind. St. Pere Molanta  
 Apartado de Correos 65  
 E-08799 Olerdola - Barcelona - Spain

Fecha: 23/09/2011

PRUEBA DE APROBACION

**PROCESO FABRICACIÓN LITOGRAFÍA**

APLICACIÓN	COMPOSICIÓN
1 <input type="checkbox"/> BARNIZADO	
2 <input type="checkbox"/> BARNIZADO	
3 <input type="checkbox"/> BARNIZ <input type="checkbox"/> COUCHE <input type="checkbox"/> LITOGR.	
4 <input type="checkbox"/> BARNIZ <input type="checkbox"/> COUCHE <input type="checkbox"/> LITOGR.	
5 <input type="checkbox"/> BARNIZ <input type="checkbox"/> COUCHE <input type="checkbox"/> LITOGR.	
6 <input type="checkbox"/> BARNIZ	

**DECORACIÓN LATERAL**

<p><b>OFFSET</b></p> <p>Barnizado inicial .....</p> <p>Altura decoración <u>A 5 mm DE LA BASE</u></p> <p>Composición colores</p> <p>*Color <u>NEGRO SF 190</u></p> <p><u>SECANTE 215 - 31</u></p> <p>*Color .....</p> <p>*Color .....</p> <p>Barniz final <u>WET ON WET</u></p> <p><u>5050</u></p> <p>Observaciones .....</p>	<p style="text-align: right;">GCS 4102</p> <p><b>SERIGRAFÍA</b></p> <p>Barnizado inicial .....</p> <p>Altura decoración .....</p> <p>Composición colores</p> <p>1º Color .....</p> <p>2º Color .....</p> <p>3º Color .....</p> <p>Barniz final .....</p> <p>Observaciones .....</p>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Figura 6 – Ficha PD de papel

La cantidad de cartulinas acumuladas con el paso del tiempo ha llegado a ser notable, y si bien es cierto que desde hace un algún tiempo se empezó a escanear estas cartulinas, la organización de estos datos, cantidad de copias, organización y transmisión de la información es un problema.



Figura 7 – Registro de PDs

Actualmente los estados de las PDs son los “Aprobada” y “Anulada”.

### **1.3 Objetivos que se quieren conseguir con la realización del TFC**

El objetivo principal de este TFC es el desarrollo de una aplicación que dé solución al problema planteado, utilizando las plataformas y herramientas tecnológicas nombradas anteriormente.

Quiero crear una aplicación que nos permita tener organizada toda la información que ya existe, introduciendo además nuevos “estados”, en las PD para definir y mejorar el flujo de trabajo entre las partes cuando haya un cambio de estado, notificándolas mediante correo electrónico, cuando este se produzca.

Hay que destacar que la solución tecnológica propuesta, se ha hecho pensando en que tenga un mínimo coste económico dado que todas las herramientas planteadas son software libre sin coste de licencias, a excepción de la base de datos SQL 2008 Enterprise Estándar, que utilizaremos en el entorno productivo dado que ya está instalada, pero podríamos utilizar MySQL y así minimizar al máximo el coste del proyecto.

Durante el desarrollo del proyecto trabajaremos con lenguajes de programación WEB, WEB Services y bases de datos como tecnologías principales.

Por otro lado, pondremos en práctica la aplicación siguiendo las especificaciones y las etapas del proceso de desarrollo, planificación, análisis, implementación y pruebas descritas en el plan docente de TFC, así como todos los formalismos en la entrega del proyecto y la memoria del mismo.

## 1.4 Enfoque y método seguido

### 1.4.1. Metodología

En principio, después de un pequeño análisis, me decidí por desarrollar el TFC siguiendo las etapas previstas en un ciclo de vida clásico o en cascada, puesto que los requisitos de la plataforma estaban completamente definidos y no cambiarían a lo largo del plazo establecido por realizar el TFC. Las etapas de este ciclo de vida son:

1. Análisis previo: En esta fase se han definido a grandes rasgos el sistema de software necesario para realizar el TFC.
2. Análisis de requisitos: He analizado todas las necesidades de la plataforma y definido una especificación detallada de los requisitos.
3. Diseño: He dado solución a todas las necesidades y requisitos de la etapa anterior (arquitectura general, estructuras de datos, interfaces, ...)
4. Programación: He generado el código que refleja las especificaciones de la fase anterior.
5. Prueba: He localizado y corregido todos los errores de la plataforma.
6. Mantenimiento.

En general me he mantenido fiel al ciclo de vida clásico, pero a la parte final he utilizado algo de programación exploratoria, es decir, una vez finalizada la plataforma realicé pequeños cambios para mejorar la navegación en los menús de la aplicación.

### 1.4.2. Herramientas

Para cubrir todos los objetivos de la plataforma, en el análisis previo utilizaremos el lenguaje de programación PHP para crear las páginas web dinámicas, dónde podremos crear nuevas peticiones de pruebas de decoración, hacer un seguimiento de su estado de desarrollo y consultar todos los datos de estas una vez ya aprobadas. También utilizaremos un Framework que se llama Openbiz que nos ayudará a la creación y estructura de la web mediante módulos o estructuras semi desarrolladas en PHP.

Necesitamos un servidor web Apache para generar estas páginas web y una base de datos SQL, aprovechando que ya hay un servidor SQL disponible en la empresa, dónde almacenar toda la información relevante. Eclipse nos ayudará en el diseño y la administración de los metadatos así como también en el mantenimiento de la aplicación como entorno de trabajo.



Eclipse es un software libre para el desarrollo integrado de código abierto multiplataforma.

Para más información acerca de Eclipse, podemos visitar su página web en el siguiente link:

<http://www.eclipse.org>

## **Tecnología empleada:**

PHP Openbiz, PHP, SQL, HTML, CSS, JavaScript

### **1.5 Planificación del proyecto**

A continuación, se describe la planificación del proyecto según el calendario establecido en la primera PAC de la asignatura, así como también el listado de las tareas a realizar para conseguir los objetivos descritos:

#### Entrega PAC1: Definición i planificación

- Justificación
- Objetivos generales y específicos
- Planificación metas y temporización
- Creación MS Project / diagrama Gantt

#### Entrega PAC2: Análisis i diseño

- Documentación PAC1
- Estudio etapas del proyecto y módulos a desarrollar
- Identificación clases y estructuras de la base de datos
- Casos de uso
- Diseño interfaz
  - Análisis interface/Diseño pantallas
  - Documentación

#### Entrega PAC3: Implementación

- Descripción del software utilizado
- Instalación software necesario
- Análisis y creación BBDD
- Implementación (programación)
  - Pruebas
  - Corrección de errores
  - Documentación

#### Entrega PAC4: Entrega final y memoria

- Creación memoria
- Presentación virtual

## La planificación temporal detallada de las tareas y sus dependencias – Diagrama de Gantt

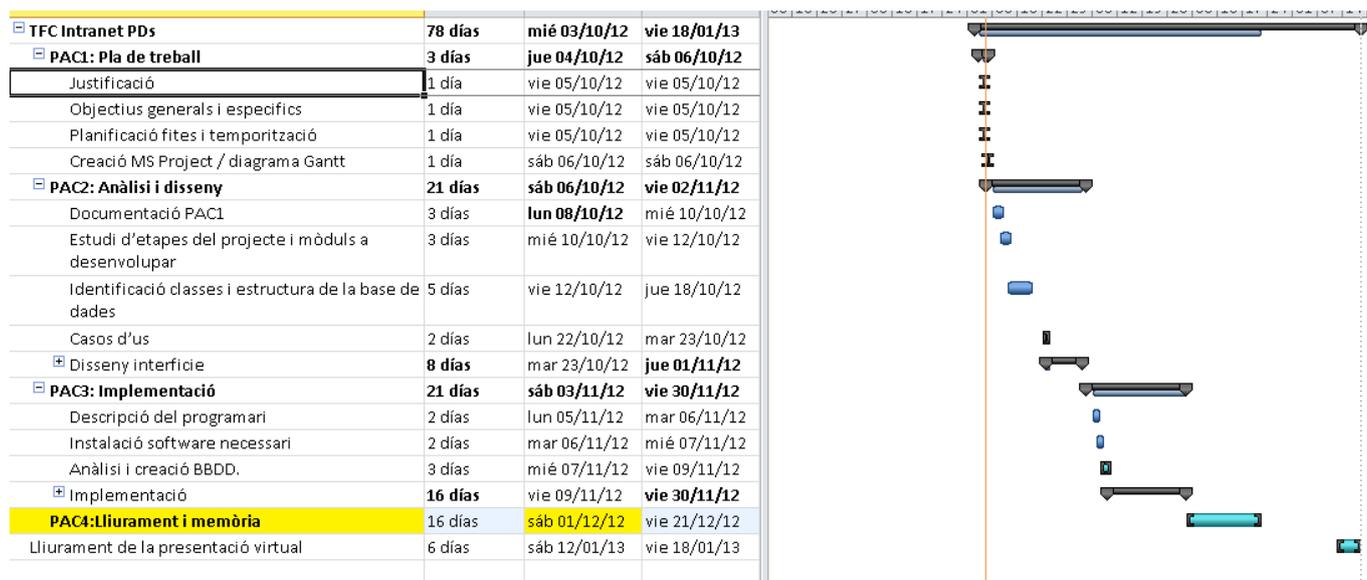


Figura 8 – Diagrama de Gantt

### 1.6 Breve descripción de los otros capítulos

#### CAPÍTULO 2: ARQUITECTURA

En este apartado comentamos la arquitectura de la aplicación e introducimos la organización de la aplicación.

#### CAPÍTULO 3: DISEÑO

En este apartado comentamos el diseño de la aplicación en todas sus fases.

#### CAPÍTULO 4: TECNOLOGIAS APLICADAS

En este apartado comentamos las tecnologías empleadas para el desarrollo de la aplicación, haciendo especial énfasis en los Frameworks Openbiz utilizados para el desarrollo del proyecto.

#### CAPÍTULO 5: JUEGOS DE PRUEBAS

En este apartado comentamos los dos juegos de pruebas realizados tras el desarrollo de la aplicación.

#### CAPÍTULO 6: CONCLUSIONES

En este apartado hay las conclusiones del TFC.

#### CAPÍTULO 7: GLOSARIO

Glosario de algún vocabulario técnico propio del sector.

#### CAPÍTULO 8: BIBLIOGRAFIA

Referencias bibliográficas utilizadas para el desarrollo del TFC.

## 2. ARQUITECTURA

La arquitectura de la solución que planteamos, está basada en la arquitectura cliente-servidor, que permite su fácil acceso desde cualquier puesto de trabajo conectado a la red de la organización. Es una solución distribuida y escalable, que permite la simultaneidad de accesos de gran cantidad de usuarios, dependiendo de la capacidad de los servidores.

Esta solución está pensada para que pueda integrarse con servidores de correo, lo que nos permitirá el envío automático de mensajes de correo electrónico con los datos o informes que generemos en la aplicación.

La aplicación se va a desarrollar en el lenguaje de programación PHP, uno de los lenguajes más generalizados a nivel internacional para desarrollo de aplicaciones web en entornos empresariales con software libre.

Como entorno de trabajo se utilizaremos el framework Openbiz, que es un *framework "open source"* que permite posteriormente un fácil análisis y mantenimiento del sistema, así como su posterior desarrollo y mantenimiento por parte de la propia organización o usuarios avanzados.

Entre las innovaciones principales de la arquitectura de Openbiz está su kernel de **metadatos**. Esto quiere decir que los objetos son construidos basados en la descripción de los archivos de metadatos, construir una aplicación se resume a diseñar y desarrollar archivos de metadatos la mayoría del tiempo. Los archivos de metadatos son descritos en lenguaje XML y es por ello que es tan fácil de mantener, de esa forma el framework Openbiz aprovecha toda la naturaleza escalable de XML.

### ¿Qué es XML?

XML, siglas en inglés de eXtensible Markup Language ('lenguaje de marcas extensible'), es un lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C). Deriva del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones se deben comunicar entre sí o integrar información.

El servidor de base de datos de la instalación es SQL Server 2008 R2 pero el sistema puede trabajar con cualquier base de datos relacional.

En la aplicación, existirán diferentes tipos de usuarios:

**Departamento de atención al cliente / Customer service:** Reciben de los clientes las nuevas peticiones de desarrollo y las introducen en el sistema. Una vez terminada la prueba de decoración, la validan en el sistema.

**Departamento de desarrollo de PD:** Encargados de desarrollar la prueba de decoración e introducir los datos en el sistema.

**Operarios producción:** Pueden consultar los datos del sistema para la producción en serie una vez la prueba de decoración haya sido aprobada por el cliente (PDs aprobadas).

Nuestro propósito es que los diferentes roles de usuarios interactúen de manera diferente con la aplicación, de manera que cada tipo de usuario tendrá acceso sólo a la información o a las partes de la aplicación relevantes en las que intervenga su trabajo. A continuación se muestra una figura que representa los flujos de información, una representación de la infraestructura necesaria y los diferentes tipos de usuarios que interactúan con la aplicación.

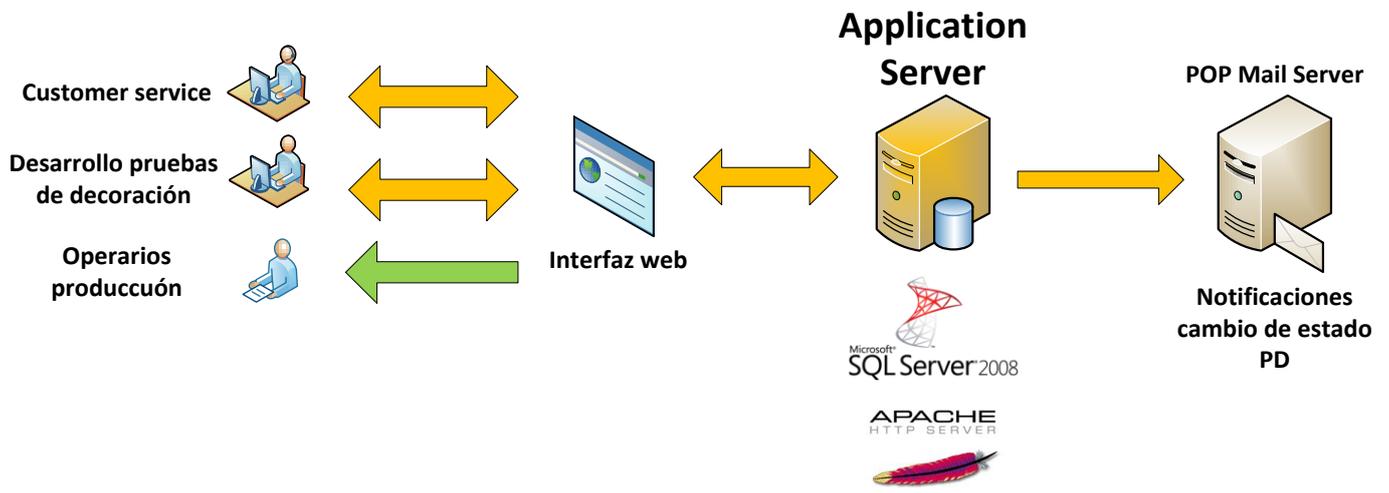


Figura 9 – flujos de información

### 3. DISEÑO

#### 3.1 Como construir la aplicación

Para construir la aplicación, tenemos que seguir los siguientes pasos como nos muestra la siguiente imagen:

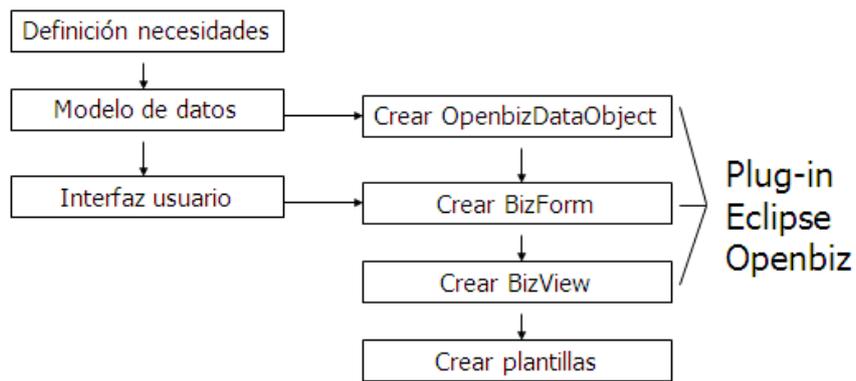


Figura 10 – Pasos desarrollo aplicación

Esto significa, que a la práctica, dentro del nuevo módulo, donde crearemos la aplicación, al que llamaremos "epd", tendremos siempre dentro de cada sección del módulo las siguientes carpetas:

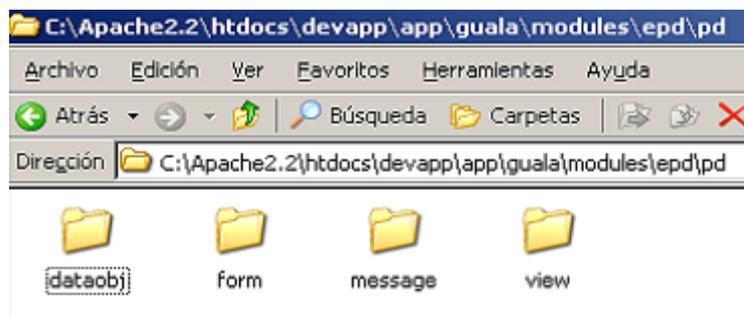


Figura 11 – Estructura módulo

## ◆ Diseño del modelo de datos

Esto incluye analizar los requisitos o necesidades de la aplicación para crear el diseño de los datos. Hay que crear la base de datos, las tablas, los campos, las claves y establecer las relaciones entre las tablas de la base de datos.

- ✓ Creamos las tablas/índices usando los asistentes (GUI tools) o con scripts.
- ✓ Especificaremos el prefijo (Prefix) en el id\_ de las tablas para cada que cada nuevo registro auto-generado contenga la ruta como "Prefix"+secuencia formato. Es muy recomendable y una buena forma de tener bien organizadas las tablas.
- ✓ Añadimos la información de conexión a la base de datos en el config.xml.

```
<?xml version="1.0" standalone="no"?>
<Config>
  <DataSource>
    <Database Name="Default" Driver="Pdo_Mssql" Server="192.168.1.105\GUALA" Port="1433" Charset="UTF-8" DBName="g"
    <Database Name="gualaerp" Driver="Pdo_Mssql" Server="192.168.1.105\GUALA" Port="1433" Charset="UTF-8" DBName="g"
    <Database Name="mail_" Driver="Pdo_Mssql" Server="192.168.1.105\GUALA" Port="1433" Charset="UTF-8" DBName="g"
    <Database Name="erp" Driver="Pdo_Mssql" Server="192.168.1.105\GUALA" Port="1433" Charset="UTF-8" DBName="gual
  </DataSource>
  <ThemeSource Default="Default">
    <Theme Name="mytheme" Description="Default Theme for OpenBiz" Path="theme1" Author="Jixian W." AuthorEmail="j
    <Theme Name="Default" Description="My Theme" Path="theme1" Author="Luis Ernesto." AuthorEmail="jixian2003@gg.
    <Theme Name="Diakrit" Description="Diakrit Style Theme" Path="diakrit" Author="Jixian W." AuthorEmail="jixiar
  </ThemeSource>
</Config>
```

Figura 12 – Config.xml de la aplicación

## ◆ Diseño de los Openbiz Data Objects (DO – business data unit):

Organizado en tres partes:

- ✓ Propiedades del BizDataObject: Hay que ver que propiedades necesita el OpenbizDataObject. Cada OpenbizDO está basado en una database table-base table.
- ✓ Funciones proporcionadas por el BizDataObject: Las funciones básicas ya son proporcionadas por el Openbiz framework. Podemos implementar funciones especiales mediante extender el OpenbizDO (para nuestro caso no requerido).
- ✓ Relaciones entre los BizDataObjects.

A continuación vemos un diseño de datos, en concreto el de una PD, y vemos cómo y dónde definimos todos sus campos necesarios:

```
<?xml version="1.0" standalone="no"?>
<BizDataObj Name="v_d_ga_pd_list" Description="" Class="BizDataObj" DBName="gualaerp" Table="v_ga_pd_list" Search="">
  <BizFieldList>
    <BizField Name="id" Column="id" Length="4" Type="Number"/>
    <BizField Name="num_pd" Column="num_pd" Length="4" Required="Y" Type="Number" SQLExpr=""/>
    <BizField Name="descripcion" OnAudit="Y" Column="descripcion" Length="100" Required="N" Type="Text"/>
    <BizField Name="estado" OnAudit="Y" Column="estado" Required="N" Type="Number"/>
    <BizField Name="fecha_envio" Column="fecha_envio" Required="N" OnAudit="Y" Type="Datetime" Format=""/>
    <BizField Name="fecha_anulacion" Column="fecha_anulacion" Required="N" OnAudit="Y" Type="Datetime" Format=""/>
    <BizField Name="fecha_aprobacion" Column="fecha_aprobacion" Required="N" OnAudit="Y" Type="Datetime" Format=""/>
    <BizField Name="fecha_fin_desarrollo" Column="fecha_fin_desarrollo" Required="N" OnAudit="Y" Type="Datetime" Format=""/>
    <BizField Name="aprobada" OnAudit="Y" Column="aprobada" Required="N" Type="Number"/>
    <BizField Name="anulada" OnAudit="Y" Column="anulada" Required="N" Type="Number"/>
    <BizField Name="motivo_retrazo" OnAudit="Y" Column="motivo_retrazo" Required="N" Type="Number"/>
    <BizField Name="generica" Column="generica" Required="N" Type="Number"/>
    <BizField Name="cliente" Column="cliente"/>
    <BizField Name="ref_cliente" Column="ref_cliente"/>
    <BizField Name="fecha_solicitud" Column="fecha_solicitud" Type="Datetime" Format="%d-%m-%Y"/>
    <BizField Name="comanda_curso" Column="comanda_curso"/>
    <BizField Name="tcabeza_fondo" Column="tcabeza_fondo"/>
    <BizField Name="tlaterral_fondo" Column="tlaterral_fondo"/>
    <BizField Name="fecha_solicitud_entrega" Column="fecha_solicitud_entrega" Type="Datetime" Format=""/>
    <BizField Name="info" Column="info"/>
    <BizField Name="shell" Column="shell"/>
    <BizField Name="shell_id" Column="shell_id"/>
    <BizField Name="tipo" Column="tipo"/>
  </BizFieldList>
</BizDataObj>
```

Figura 13 – Ejemplo diseño DO

## ◆ Diseño de los Formularios Openbiz (Presentation unit)

Los BizForms los organizamos también en tres partes:

- ✓ Un BizForms es el interfaz de usuario (UI) proxy de un BizDataObject: Un BizForm puede tener diferentes tipos de presentación (default, edit, query) asociado con plantillas (templates).
- ✓ Hay que mapear un campo de control desde un campo de un BizDataObject:
  - Mapear un campos de control desde un campo base BizObject.
  - Hay que ver si un campo necesita recoger datos desde otro BizForm.
  - Hay que ver si los campos deben ser editables o visibles.
- ✓ Añadir funciones a los BizForm
  - Mapear funciones proporcionadas por la clase BizForm hacia los botones.
  - Implementar las funciones especiales extendiendo el BizForm (para nuestro caso no requerido).

A continuación vemos un diseño de un formulario que hemos creado, dónde podemos cambiar la fecha requerida de entrega de una PD:

```
<?xml version="1.0" encoding="UTF-8" ?>
<EasyForm Name="f_ga_pd_edit_fecha_requerida_entrega" Class="PickerForm" FormType="Edit" jsClass="jbForm" Title="
  <DataPanel>
    <Element Name="fld_Id" Class="Hidden" FieldName="Id" Label="Id" Sortable="Y" AllowURLParam="Y" />
    <Element Name="fld_num_pd" Class="InputText" FieldName="num_pd" Label="Num_pd" Sortable="Y" Enabled="N" Allo
    <Element Name="fld_descripcion" Class="Textarea" FieldName="descripcion" Label="Descripcion" AllowURLParam="I
    <Element Name="fld_estado" Class="Listbox" SelectFrom="epd.Selection(estado_pd(PD_WORKFLOW))" FieldName="est:
    <Element Name="fld_motivo_retraso" Class="Listbox" SelectFrom="epd.Selection(motivo_retraso)" FieldName="mo
    <Element Name="fld_fecha_entrega_pd" Class="InputDate" DateFormat="%d-%m-%Y" Required="Y" FieldName="fecha_e
    <Element Name="fld_generica" Class="Hidden" FieldName="generica" Label="generica" AllowURLParam="N" CssClass:
  </DataPanel>
  <ActionPanel>
    <Element Name="btn_save" Class="LabelText" Text="Save" CssClass="button">
      <EventHandler Name="onclick" Event="onclick" Function="SaveRecord()" ShortcutKey="Enter"/>
    </Element>
    <Element Name="btn_cancel" Class="LabelText" Text="Cancel" CssClass="button">
      <EventHandler Name="onclick" Event="onclick" Function="Close()"/>
    </Element>
  </ActionPanel>
  <NavPanel>
  </NavPanel>
  <SearchPanel>
  </SearchPanel>
</EasyForm>
```

Figura 14 – Ejemplo diseño Form

## ◆ Diseño de las Vistas Openbiz (Presentation container)

- ✓ Las vistas (BizView) son los contenedores de los BizForms y otros controles HTML, el diseño de la página web está definido en las plantillas (templates).
- ✓ Hay que definir las relaciones entre los BizForms. Un formulario puede contener subformularios con dependencias entre ellos. También pueden tener dependencias los datos de estos entre ellos.

Un ejemplo sencillo es el diseño de una Vista, dónde podemos ver el detalle de una PD genérica:

```

<?xml version="1.0" standalone="no" ?>
<EasyView Name="view.ga_pd_generica_detail" Class="EasyView" Tab="PD" Description="Gestion de PDs"
TemplateEngine="Smarty" Module="EPD" TemplateFile="view_pd_generica_detail.tpl">
  <FormReferences>
    <Reference Name="epd.pd.form.f_ga_pd_generica_detail" SubForms="common.form.f_AttachList;epd.pd.form.f_ga_pd_shell
    <Reference Name="common.form.f_AttachList_generica"/>
    <Reference Name="epd.pd_solicitud.form.f_ga_pd_solicitud_generica_list" SubForms="common.form.f_AttachList_generica
    <Reference Name="common.form.f_AttachList_generica_solicitud"/>
    <Reference Name="epd.pd.form.f_ga_pd_shell_list_generica" SubForms="epd.pd.form.f_ga_pd_shell_pasadas_list_generica
    <Reference Name="epd.pd.form.f_ga_pd_shell_pasadas_list_generica"/>
    <Reference Name="epd.pd_barniz.form.f_ga_pd_barniz_list" />
    <Reference Name="epd.pd_tinta_dl.form.f_ga_pd_tinta_dl_list" />
    <Reference Name="epd.pd_tinta_lito.form.f_ga_pd_tinta_lito_list" />
    <Reference Name="epd.pd.form.f_ga_pd_detail_menu"/>
  </FormReferences>
</EasyView>

```

Figura 15 – Ejemplo diseño Vista

## ◆ Diseño de plantillas HTML (Template design) – GUI

Las plantillas se definen en ficheros .tpl de *template*.

Existen diferentes tipos de plantillas (o modelos) que podemos utilizar. Son las siguientes:

- ✓ Plantillas típicas
  - Plantilla tabla: Una tabla HTML que contiene, como es nuestro caso, varios formularios Openbiz.
  - Plantilla Edit: Una tabla HTML que contiene un solo formulario Openbiz.
- ✓ Plantillas especiales: Como por ejemplo un calendario

Todas las plantillas deben tener el formato `<form id={$name} name={$name}>...</form>`

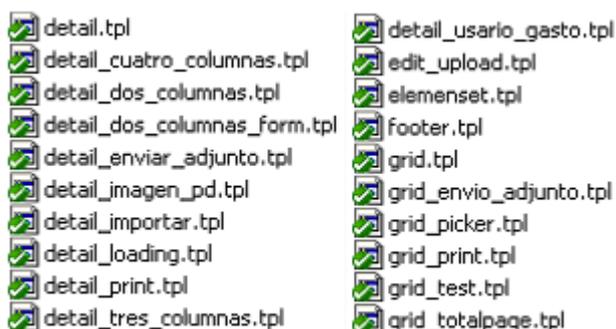
Podemos utilizar la salida de variables desde BizForm.Render para nuestras plantillas:

- {\$title}, {\$name} como valor
- {\$toolbar}, {\$navbar} como array
- {\$columns} como array
- {\$fmttable} como tabla HTML
- {\$fields} como array

Finalmente diseñar el estilo (UI theme) de nuestra aplicación

- Definir el color del tema usando CSS
- El archivo CSS file va referido a la plantilla BizView
- Definir el estilo de los botones

Vemos algunas plantillas utilizadas en la aplicación:



## ¿Qué es CSS?

CSS es el acrónimo de CascadingStyle Sheets (es decir, hojas de estilo en cascada). Es un lenguaje de estilo que define la presentación de los documentos HTML. Por ejemplo, CSS abarca cuestiones relativas a fuentes, colores, márgenes, líneas, altura, anchura, imágenes de fondo, posicionamiento avanzado y muchos otros temas.

### 3.2 Estudio de la estructura y las funcionalidades de la aplicación

Previo al desarrollo y a la programación estricta de la aplicación y creación de la base de datos, hay que ver cuáles son las funcionalidades que deseamos que tenga la aplicación.

Las funcionalidades relevantes que desarrolla la aplicación son:

1. **Informes:** Podremos visualizar los informes del módulo de PDs. Se tratan de informes o listados referentes a la información contenida en el sistema, como por ejemplo un listado de solicitudes de PD, donde su estado es "Pendiente\_Desarrollo".
2. **Solicitudes de PDs - Gestión de las solicitudes de los PDs:** En esta sección podemos crear nuevas solicitudes de PD o modificar las ya existentes. Estas solicitudes son específicas de un solo cliente.

Detalle de Solicitud de PD

Número Solicitud		<b>DECORACIÓN CABEZA</b>
Nombre Cliente	Número de Colores	
Fecha Solicitud	Colores	
Fecha Requerida Entrega		
Estado	Número de Colores	
Genérica	Colores	
Referencia del Cliente	DECORACIÓN INTERIOR	
Solicitud Muestra Física	Barniz Interior	
Solicitud de Plancha	Líquido que Contiene	
Solicitud PDF	Inserto	
	Godranatura	
Pedido en Curso	Shell	
Usuario	Comercial	
Sustituye A	Nombre del Proyecto	
Observaciones		

Figura 16 – Campos de una solicitud de PD

3. **Solicitudes PD Genéricas - Gestión de las solicitudes de los PDs genéricas:** Al igual que en solicitudes de PD, en esta sección podemos crear nuevas solicitudes de PD genéricas o modificar las ya existentes.
4. **Gestión de PDs:** Dónde gestionamos los datos de las PDs ya desarrolladas o creamos una nueva PD genérica. También gestionamos los datos de los aditivos o componentes de las PD.
5. **PDs Aprobadas:** Funcionalidad orientada a los equipos de trabajo que solo podrán visualizar y solicitar los cambios a los PDs, cuyo estatus ya sea "aprobado".
6. **Aprobaciones:** Funcionalidad para aprobar a los cambios propuestos por los operarios sobre las PD. (Test – No desarrollado)

## Nuevos estados de las PDs

Queremos crear nuevos estados en las PD para definir y mejorar el flujo de trabajo entre las partes cuando haya un cambio de estado, notificándolas mediante correo electrónico, cuando este se produzca. Los nuevos estados serán los siguientes:

- ✓ **Pendiente de desarrollo:** Creada la solicitud por parte del departamento comercial, la PD se encuentra a la espera de que el departamento de desarrollo cree la PD de muestra.
- ✓ **Desarrollada:** El departamento crea la PD e introduce el cambio de estado en el sistema.
- ✓ **Pendiente de aprobación:** El departamento comercial, recibe la PD y la reenvía al cliente a la espera de su aprobación. El departamento comercial cambia el estado a “pendiente de aprobación”.
- ✓ **Aprobada:** Se recibe el OK del cliente y pasa a Aprobada. Si no es así, vuelve a “pendiente de desarrollo”.
- ✓ **Anulada:** Estado que invalida la PD. Se puede anular una PD en cualquier momento durante el proceso.

Para llegar al estado “Aprobada”, una PD tiene que pasar por todos los estados intermedios de manera consecutiva, es decir, sin saltos entre ellos (una PD no puede pasar de “pendiente de desarrollo a Aprobada”).

## Diagramas de casos de uso

Un diagrama de los casos de uso de la aplicación es de gran utilidad para organizar y analizar las funciones y los menús de la aplicación que queremos desarrollar. A continuación mostramos el diagrama de casos de uso para los usuarios del departamento comercial o *customer service*:

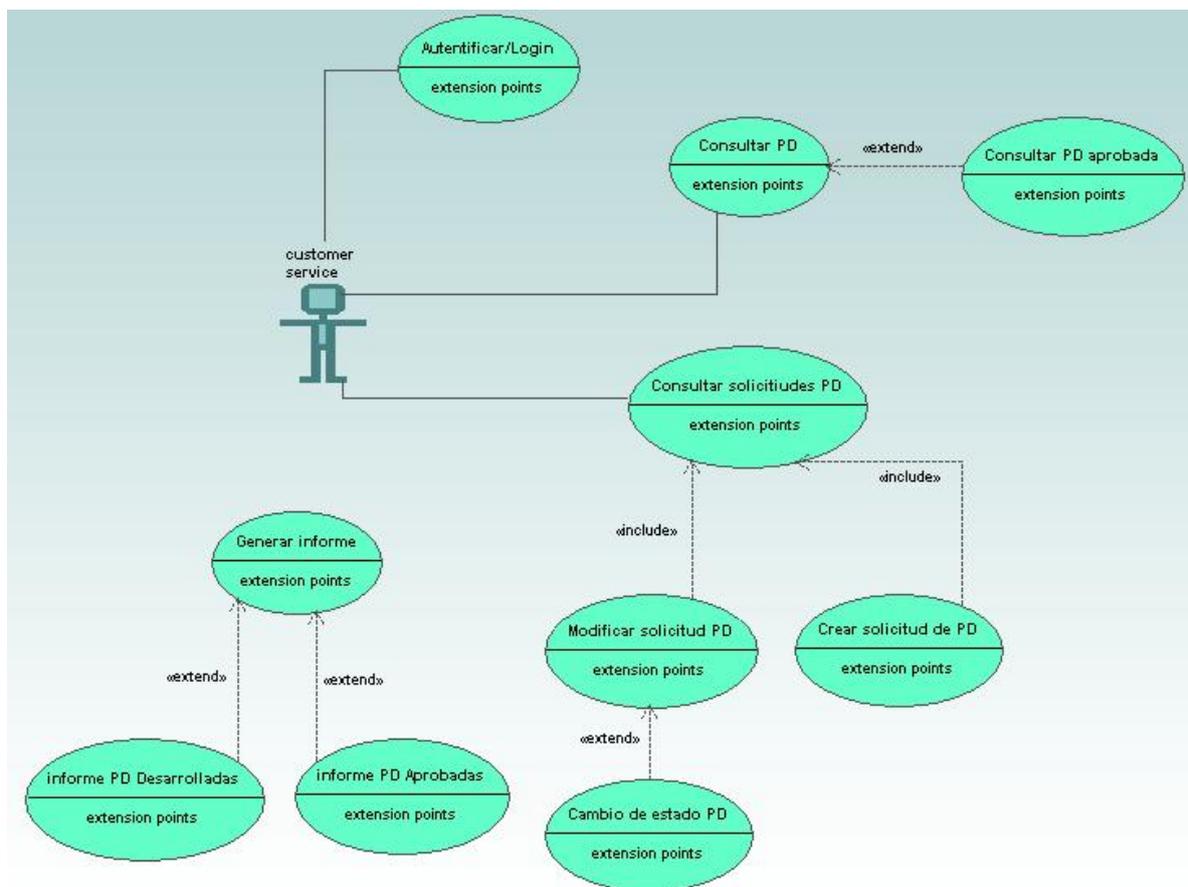


Figura 17 – Diagrama de uso usuario customer service

En este esquema he omitido el caso de las solicitudes de PD genéricas, cuya estructura sería la misma que para las solicitudes de PD normales. A destacar que en el caso de “cambio de estado de PD”, sólo podremos cambiar a estados a los que tengamos asignados (pendiente de aprobación, aprobada o anulada).

A continuación mostramos el diagrama de casos de uso para los usuarios del departamento de desarrollo de PDs:

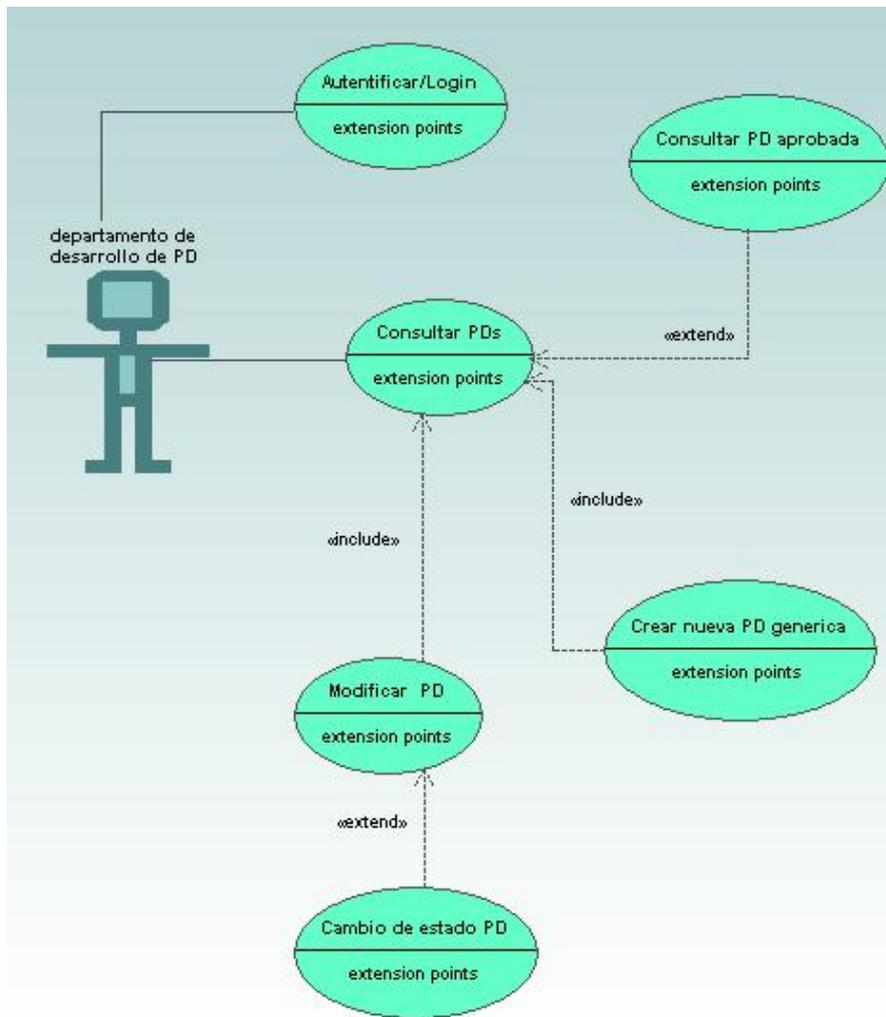


Figura 18 – Diagrama de uso usuario departamento desarrollo PDs

A destacar que, al igual que con el usuario de customer service, en el caso de “cambio de estado de PD”, este perfil, sólo podrá cambiar a estado “desarrollada”, una vez se haya terminado la PD.

El administrador de la aplicación, como súper usuario, tendrá acceso a todos los menús de la aplicación. El perfil más básico, “operario producción” sólo tendrá acceso a consultar los datos de las PDs aprobadas.

### **Organización menú aplicación**

La solución consta de 17 funcionalidades organizadas en 7 sub-menús de explotación. Algunas funcionalidades se encuentran en proceso de desarrollo y perfeccionamiento.

En el siguiente esquema se puede apreciar la estructura de los módulos y las funcionalidades de la aplicación en el menú inicial, ubicado en la página principal del sistema.



Figura 19 – Menú principal de la aplicación

### **3.3 Identificación clases y estructuras de la base de datos**

El diseño de la estructura de la base de datos, está basado en la actual estructura de las fichas de papel que contienen todos los datos necesarios para los desarrollos de las pruebas de decoración (PDs).

Además, también contiene toda la información para la gestión de solicitudes de PD y su circuito de aprobación entre desarrolladores del arte de la prueba y el departamento comercial (customer service), quien con comunicación con el cliente, verifica la aprobación de la PD para su producción en serie.



Una vez hecho el diseño, hay que crear toda la estructura de tablas y las correspondientes vistas en SQL:

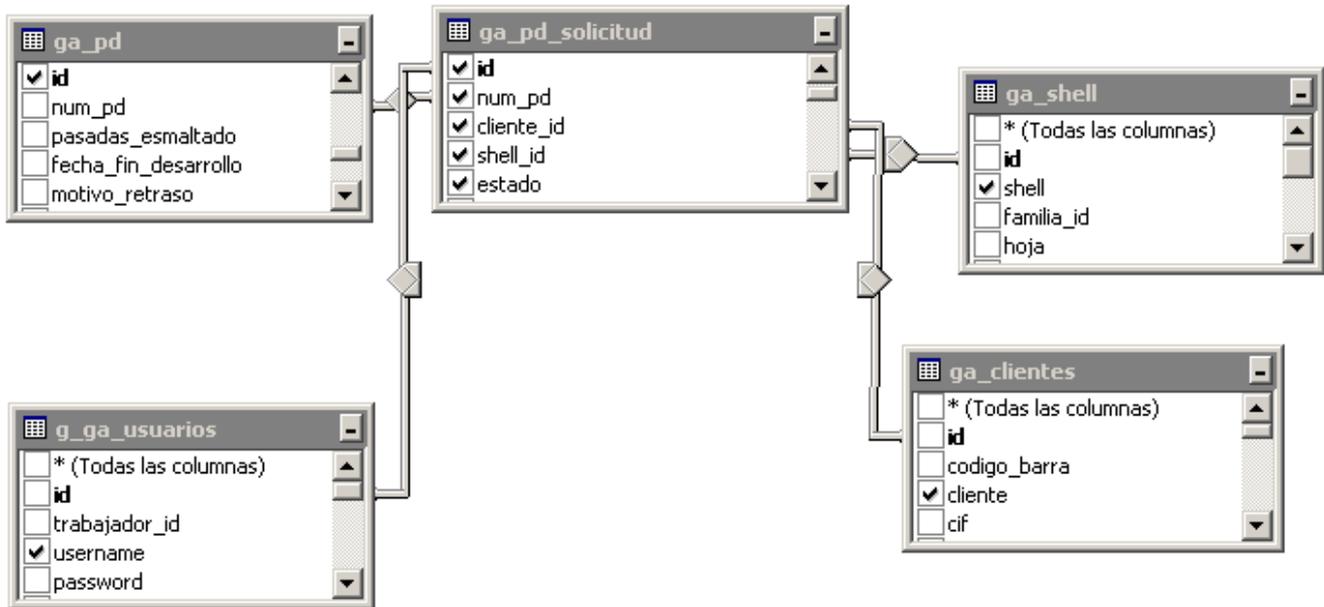


Figura 21 – Vinculación de tablas en SQL

### Descripción tablas principales para la gestión de las PDs

A continuación, describimos en detalle la estructura de las principales tablas de esta aplicación:

## Entity - ga\_pd\_solicitud

### Properties

Name	Value
Documentation	Entidad que registra la solicitud de PD

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave de la Solicitud
<a href="#">num_pd</a>	Número que Identifica al PD de esta solicitud
<a href="#">cliente_id</a>	Se registra el "id" del cliente que se selecciona en el campo "Nombre Cliente", este primer campo aparece inhabilitado por defecto. Campo vinculado a una tabla de clientes.
<a href="#">shell_id</a>	Id que identifica al Shell que se pide en esta solicitud Campo que registra el estado de la solicitud:  Value="1" Text="Pendiente de Desarrollo"  Value="2" Text="Pendiente de Aprobación"  Value="3" Text="Aprobada"  Value="4" Text="Anulada"
<a href="#">Estado</a>	
<a href="#">ref_cliente</a>	Descripción de referencia del cliente
<a href="#">fecha_solicitud</a>	Fecha en la cual se realiza la solicitud
<a href="#">Info</a>	Información o descripción de la solicitud

<a href="#">interior_barniz</a>	Descripción del interior_barniz
<a href="#">tcabeza_fondo</a>	Descripción de tcabeza_fondo
<a href="#">tcabeza_marca</a>	Descripción de tcabeza_marca
<a href="#">tlateral_fondo</a>	Descripción de tlateral_fondo
<a href="#">tlateral_marca</a>	Descripción de tlateral_marca
<a href="#">comanda_curso</a>	Descripción de comanda_curso
<a href="#">user_solicitante_id</a>	Clave del Usuario que registra la solicitud
<a href="#">fecha_solicitud_entrega</a>	Fecha que solicita la entrega
<a href="#">solicitud_muestra_fisica</a>	Campo entero que registra: 1 si lleva muestra física, 0 si no lleva muestra física
<a href="#">solicitud_pdf</a>	Campo entero que registra: 1 si lleva solicitud de pdf, o si no lleva solicitud de pdf
<a href="#">solicitud_plancha</a>	Campo entero que registra: 1 si lleva solicitud plancha, o si no lleva solicitud plancha
<a href="#">generica</a>	Campo entero que registra: 1 si es sobre PD genérica , o si no sobre PD genérica

## Entity - ga\_pd

### Properties

Name	Value
Documentation	Entidad que registra el PD

### Column Summary

Name	Documentation
<a href="#">id</a>	
<a href="#">num_pd</a>	Clave que Identifica al PD
<a href="#">estado</a>	
<a href="#">descripcion</a>	Descripción de la PD
<a href="#">fecha_envio</a>	fecha de envío de la PD
<a href="#">aprobada</a>	Campo que registra: 1 se está aprobada, 0 si no está aprobada
<a href="#">fecha_aprobacion</a>	Fecha de aprobación del PD
<a href="#">anulada</a>	Campo que registra: 1 se está anulada, 0 si no está anulada
<a href="#">fecha_anulacion</a>	Fecha de anulación
<a href="#">wetwet</a>	Campo wetwet
<a href="#">pasadas_barniz</a>	Cantidad de Pasadas Barniz
<a href="#">pasadas_litografia</a>	Cantidad de Pasadas de Litografía
<a href="#">pasadas_serigrafia</a>	Cantidad de Pasadas de Serigrafía
<a href="#">pasadas_offset</a>	Cantidad de Pasadas Offset
<a href="#">pasadas_HotStamping</a>	Cantidad de Pasadas HotStamping
<a href="#">pasadas_Tampografia</a>	Cantidad de Pasadas de Tampografía
<a href="#">generica</a>	Campo entero que registra: 1 si es sobre PD genérica , o si no sobre PD genérica

## Entity - pdc

### Properties

Name	Value
Documentation	Entidad que registra los cambios solicitados al PD

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica los registros de la entidad que registra los cambios solicitados al PD
<a href="#">estado</a>	Estado de la solicitud de cambios en el PD  Value="1" Text="Pendiente de Revisión"  Value="2" Text="Aprobada"  Value="3" Text="Cancelada"  Value="4" Text="Cerrada"
<a href="#">tabla</a>	Tabla o Entidad a la cual pertenece la solicitud de cambio
<a href="#">registro_id</a>	Clave del registro al cual se le está solicitando el cambio
<a href="#">campo</a>	Campo o propiedad del registro al cual se le está solicitando el cambio
<a href="#">valor</a>	Valor final del campo del que se solicita el cambio Tipo de campo o acción que se solicita:  Value="1" Text="INSERT"  Value="2" Text="UPDATE"  Value="3" Text="DELETE"
<a href="#">fecha</a>	Fecha en la que se solicita el cambio
<a href="#">usuario_id</a>	Clave del usuario que solicita el cambio
<a href="#">pd_id</a>	

## Entity - ga\_barniz

### Properties

Name	Value
Documentation	Entidad que registra los Barnices

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica los Barnices
<a href="#">color</a>	Color o descripción del Barniz
<a href="#">referencia</a>	Referencia del Barniz
<a href="#">proveedor</a>	Proveedor del Barniz
<a href="#">npantone</a>	Propiedad del Barniz
<a href="#">nral</a>	Propiedad del Barniz
<a href="#">plazopago</a>	Plazo de pago por el Barniz

## Entity - ga\_attachment

### Properties

Name	Value
Documentation	Entidad que registra los archivos adjuntos de las PD y las solicitudes genéricas

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica el adjunto
<a href="#">parent</a>	Se registra a quien pertenece el adjunto: 1 si es del PD, 2 si es de la solicitud
<a href="#">parent_id</a>	Clave del padre al que pertenece el adjunto
<a href="#">filename</a>	Nombre del adjunto incluyendo la extensión
<a href="#">type</a>	Tipo del adjunto
<a href="#">size</a>	Tamaño del adjunto
<a href="#">time</a>	Fecha en la cual se registró el adjunto
<a href="#">description</a>	Descripción del adjunto
<a href="#">author</a>	Clave del usuario que registra al usuario
<a href="#">ipnr</a>	Propiedad del adjunto
<a href="#">fecha</a>	Fecha en que se registró el adjunto

## Entity - ga\_pd\_log

### Properties

Name	Value
Documentation	Entidad que registra los logs o cambios realizado al PD

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica el log registrado
<a href="#">fecha</a>	Fecha en que se realizó el cambio
<a href="#">usuario_id</a>	Clave que identifica el usuario que realizo el cambio
<a href="#">info</a>	Información del Log registrado (Insertar , Modificar , Eliminar)
<a href="#">pd_id</a>	

## Entity - ga\_pd\_ga\_pd\_solicitud

### Properties

Name	Value
Documentation	Entidad que registra la relación de una solicitud de PD con un PD

### Column Summary

Name	Documentation
<a href="#">solicitud_pd_id</a>	Clave que identifica la solicitud
<a href="#">pd_id</a>	

## Entity - ga\_pd\_ga\_pd\_solicitud

### Properties

Name	Value
Documentation	Entidad que registra la relación de una solicitud de PD con un PD

### Column Summary

Name	Documentation
------	---------------

[solicitud\\_pd\\_id](#)

Clave que identifica la solicitud

[pd\\_id](#)

## Entity - ga\_shell

### Properties

Name	Value
Documentation	Entidad que registra las Shell

### Column Summary

Name	Documentation
<a href="#">id</a>	Identificador de la entidad SHELL , ID asignado
<a href="#">shell</a>	Descripción del shell
<a href="#">hoja</a>	Hoja que utiliza el shell
<a href="#">peso_hoja</a>	Peso de la hoja
<a href="#">num_tapas</a>	Cantidad de tapas por hoja
<a href="#">porc_merma</a>	Puntos porcentuales de la merma que se genera
<a href="#">fabricacionpropia</a>	Fabricación propia : 1 True, 0 False
<a href="#">familia_id</a>	Id de la familia a la que pertenece

## Entity - ga\_pd\_tinta\_dl

### Properties

Name	Value
Documentation	Entidad que registra la tinta dl del PD

### Column Summary

Name	Documentation
<a href="#">id</a>	Clave de la tinta_dl
<a href="#">pd_shell_id</a>	Clave de la entidad que guarda la relación de PD - Shell
<a href="#">color</a>	Descripción del color de tinta dl
<a href="#">utilizacion</a>	Propiedad de utilizacion de la tinta dl Value="1" Text="Litografía" Value="2" Text="Serigrafía" Value="3" Text="Offset"
<a href="#">num_pantone</a>	Propiedad de la tinta dl
<a href="#">num_rai</a>	Propiedad de la tinta dl
<a href="#">observaciones</a>	Observaciones de la tinta dl

## Entity - ga\_pd\_tinta\_lito

### Properties

Name	Value
Documentation	Entidad que registra la tinta lito del PD

## Column Summary

Name	Documentation
<a href="#">id</a>	Clave de la Entidad que registra la tinta dl del PD
<a href="#">pd_shell_id</a>	Clave de la entidad que guarda la relación de PD - Shell
<a href="#">color</a>	Descripción del color de tinta lito
<a href="#">utilizacion</a>	Propiedad de utilizacion de la tinta lito:
	Value="" Text="-"
	Value="1" Text="Positivo"
	Value="2" Text="Negativo"
	Value="3" Text="Cabeza"
<a href="#">num_pantone</a>	Propiedad de la tinta lito
<a href="#">num_ral</a>	Propiedad de la tinta lito
<a href="#">observaciones</a>	Observaciones de la tinta lito

## Entity - ga\_pd\_barniz

### Properties

Name	Value
Documentation	Entidad que registra el barniz del PD

## Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica la Entidad que registra el barniz del PD
<a href="#">pd_shell_id</a>	Clave de la entidad que guarda la relación de PD - Shell
<a href="#">color</a>	Descripción o color del barniz
<a href="#">utilizacion</a>	Propiedad de utilizacion del barniz
	Value="1" Text="Enganche"
	Value="2" Text="Acabado"
	Value="3" Text="Couche"
	Value="4" Text="Interior"
<a href="#">num_pantone</a>	Propiedad del barniz del PD
<a href="#">num_ral</a>	Propiedad del barniz del PD
<a href="#">observaciones</a>	Observaciones del barniz del PD

## Entity - ga\_pd\_tinta\_dl\_comp

### Properties

Name	Value
------	-------

## Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica la Entidad que registra los compuestos que forma la Tinta DL del PD
<a href="#">pd_tinta_dl_id</a>	Clave que identifica la Tinta DL del PD al cual pertenece el compuesto
<a href="#">cantidad</a>	Cantidad en REAL del compuesto
<a href="#">porciento</a>	Puntos Porcentuales que representa la cantidad del compuesto para la fórmula de la Tinta DL del PD
<a href="#">tinta_id</a>	Clave de la Tinta que se utiliza en el compuesto

## Entity - ga\_pd\_tinta\_lito\_comp

### Properties

Name	Value
Documentation	Entidad que registra los compuestos que forma la Tinta LITO del PD

## Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica la Entidad que registra los compuestos que forma la Tinta LITO del PD
<a href="#">pd_tinta_lito_id</a>	Clave que identifica la Tinta LITO del PD al cual pertenece el compuesto
<a href="#">cantidad</a>	Cantidad en REAL del compuesto
<a href="#">porciento</a>	Puntos Porcentuales que representa la cantidad del compuesto para la fórmula de la Tinta LITO del PD
<a href="#">tinta_id</a>	Clave de la Tinta que se utiliza en el compuesto

## Entity - ga\_pd\_barniz\_comp

### Properties

Name	Value
Documentation	Entidad que registra los compuestos que forma el Barniz del PD

## Column Summary

Name	Documentation
<a href="#">id</a>	Clave que identifica la Entidad que registra los compuestos que forma el Barniz del PD
<a href="#">pd_barniz_id</a>	Clave que identifica Barniz del PD al cual pertenece el compuesto
<a href="#">barniz_id</a>	Clave del Barniz que se utiliza en el compuesto
<a href="#">cantidad</a>	Cantidad en REAL del compuesto
<a href="#">porciento</a>	Puntos Porcentuales que representa la cantidad del compuesto para la fórmula del Barniz del PD

## Entity - ga\_tintas

### Properties

Name	Value
Documentation	Entidad que registra la información de las Tintas

## Column Summary

Name	Documentation
<a href="#">id</a>	ID que identifica cada registro de tinta

<a href="#">color</a>	Descripcion o color de la Tinta
<a href="#">referencia</a>	Referencia de la Tinta
<a href="#">proveedor</a>	Proveedor de la tinta
<a href="#">npantone</a>	Propiedad de la tinta npantone
<a href="#">nral</a>	Propiedad de la tinta nral
<a href="#">plazopago</a>	Plazo de pago de tinta

## Entity - ga\_familias

### Properties

Name	Value
Documentation	Entidad que registra las familia de las Shell

### Column Summary

Name	Documentation
<a href="#">id</a>	Identificador de las familias, ID asignado
<a href="#">familia</a>	Descripción de la Familia

### 3.4 Diseño de metadatos con Eclipse

#### ¿Qué son los metadatos?

Los metadatos son «datos sobre datos». Son toda aquella información descriptiva sobre el contexto, calidad, condición o características de un recurso, dato u objeto que tiene la finalidad de facilitar su recuperación, autenticación, evaluación, preservación y/o interoperabilidad. Los metadatos son de suma importancia para el sistema de información que maneja los datos. Los metadatos son enviados junto con la información cuando se realiza alguna petición o actualización de la misma.

En Eclipse podemos crear fácilmente los metadatos siguiendo el asistente que el mismo Eclipse nos ofrece para este fin, siempre que hayamos instalado previamente el plug-in de Openbiz para esta Eclipse que podemos descargar del siguiente link: <http://www.phpopenbiz.org/document/release.html>.

En la siguiente imagen, podemos ver como creamos un metadato con el asistente de Eclipse:

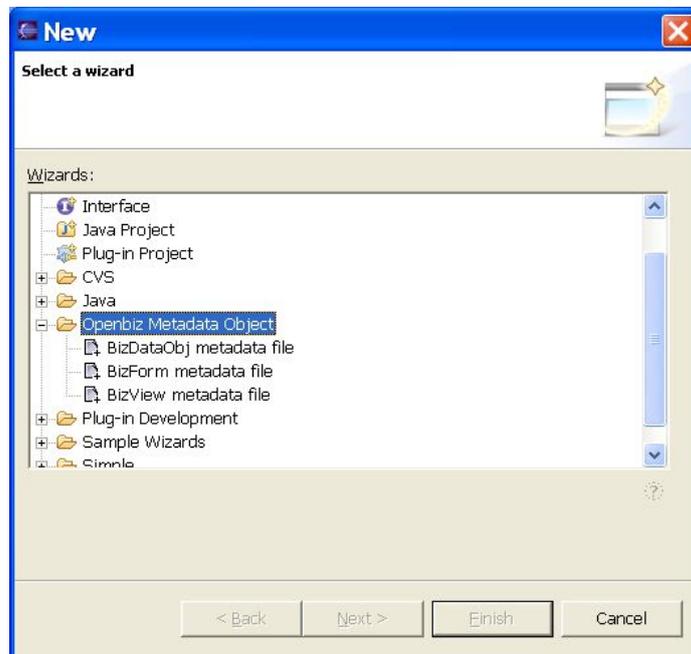


Figura 22 – Creación metadato en Eclipse

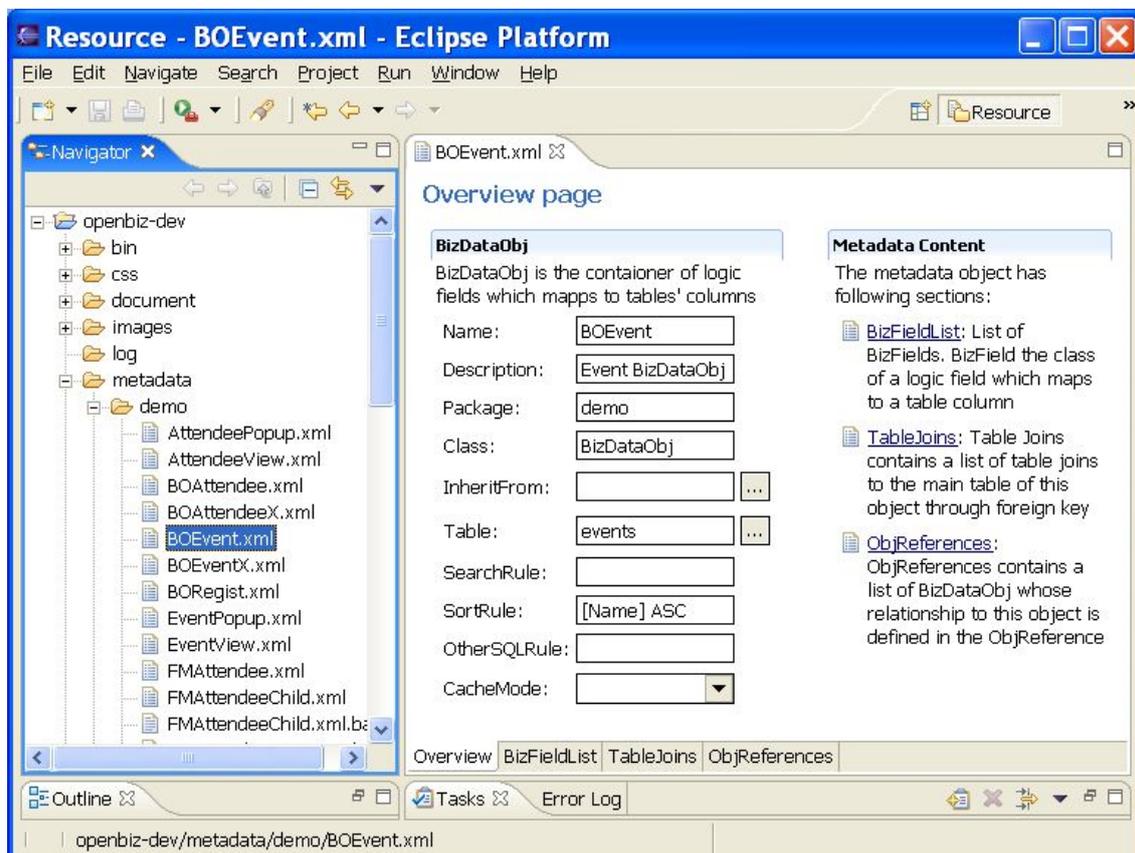


Figura 23 – Creación metadato en Eclipse

En la siguiente imagen, podemos ver como vemos un formulario con Eclipse:

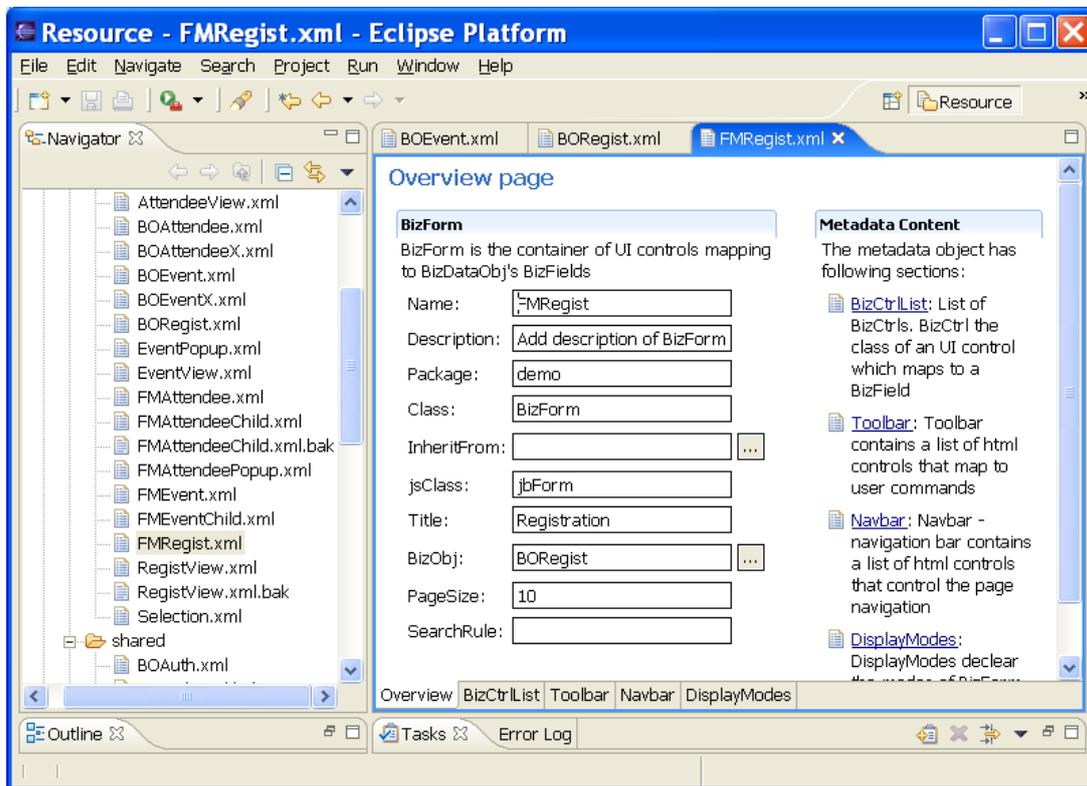


Figura 24 – Formulario en Eclipse

En la siguiente imagen, podemos ver como vemos una Vista con Eclipse:

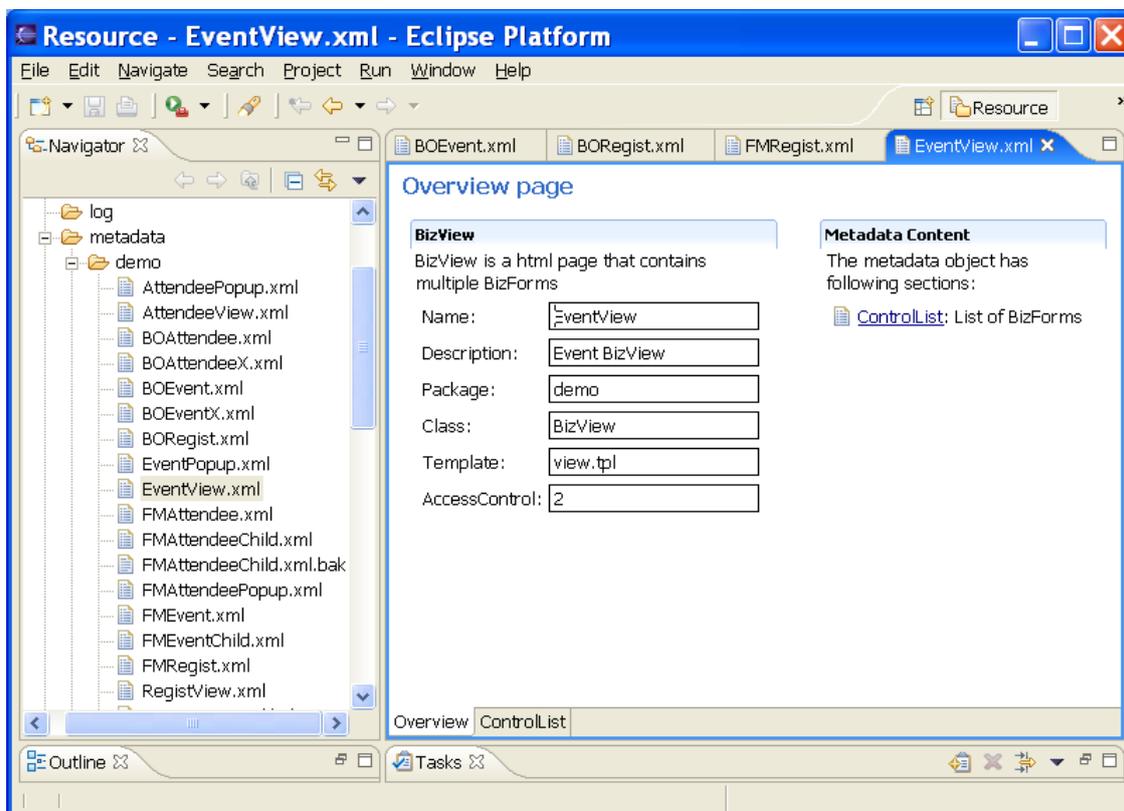


Figura 25 – Vista en Eclipse

### 3.5 Estructura XML

A continuación mostramos a modo de ejemplo dos de las estructuras XML diseñadas para la aplicación. Los documentos XML deben seguir una estructura estrictamente jerárquica con lo que respecta a las etiquetas que delimitan sus elementos. Una etiqueta debe estar correctamente "incluida" en otra. Además, los elementos con contenido, deben estar correctamente "cerrados". A continuación mostramos a modo de ejemplo dos de las estructuras XML diseñadas para la aplicación. El `d_ga_pd_estado.xml` muestra la estructura de los diferentes estados por los que puede estar una PD (aprobada, anulada) así como otros campos referentes a las fechas relevantes del proceso de desarrollo, envío al cliente y aprobación de la misma.

```
<?xml version="1.0" standalone="no"?>
<BizDataObj Name="d_ga_pd_estado" Description="" Class="d_ga_pd" DBName="gualaerp" Table="ga_pd"
SearchRule="[planta_id]='{@profile:planta_activa}'" SortRule="[Id] ASC" OtherSQLRule="" Uniqueness="num_pd"
Stateless="N" IdGeneration="Identity" CacheLifeTime="0">
  <BizFieldList>
    <BizField Name="Id" Column="id" Length="4" Type="Number"/>
    <BizField Name="num_pd" Column="num_pd" Length="4" Required="Y" Type="Number"/>
    <BizField Name="descripcion" OnAudit="Y" Column="descripcion" Length="100" Required="N"
Type="Text"/>
    <BizField Name="estado" OnAudit="Y" Column="estado" Required="N" Type="Number"/>
    <BizField Name="fecha_envio" Column="fecha_envio" Required="N" OnAudit="Y" Type="Datetime"
Format="%d-%m-%Y"/>
    <BizField Name="fecha_anulacion" Column="fecha_anulacion" Required="N" OnAudit="Y" Type="Datetime"
Format="%d-%m-%Y"/>
    <BizField Name="fecha_aprobacion" Column="fecha_aprobacion" Required="N" OnAudit="Y" Type="Datetime"
Format="%d-%m-%Y"/>
    <BizField Name="fecha" Required="N" Type="Datetime" Format="%d-%m-%Y" Value="{date('d-m-Y')}"/>
    <BizField Name="fecha_fin_desarrollo" Column="fecha_fin_desarrollo" Required="N" OnAudit="Y"
Type="Datetime" Format="%d-%m-%Y"/>
    <BizField Name="fecha_entrega_pd" Column="fecha_entrega_pd" Required="N" OnAudit="Y" Type="Datetime"
Format="%d-%m-%Y"/>

    <BizField Name="aprobada" OnAudit="Y" Column="aprobada" Required="N" Type="Number"/>
    <BizField Name="anulada" OnAudit="Y" Column="anulada" Required="N" Type="Number"/>
    <BizField Name="motivo_retrazo" OnAudit="Y" Column="motivo_retrazo" Required="N" Type="Number"/>
    <BizField Name="generica" Column="generica" Required="N" Type="Number"/>

    <BizField Name="fecha_entrega_pdf" Column="fecha_entrega_pdf" Required="N" OnAudit="Y"
Type="Datetime" Format="%d-%m-%Y"/>
    <BizField Name="motivo_retrazo_pdf" OnAudit="Y" Column="motivo_retrazo_pdf" Required="N"
Type="Number"/>
  </BizFieldList>
</TableJoins>
```

Figura 26 – Diseño XML I

El segundo ejemplo de estructura XML es `d_ga_pd_shell_pdf.xml` y nos muestra la estructura de datos con los detalles técnicos de la producción de la PD, como dimensiones de la tapa(shell), pasadas necesarias de litografía, pasadas de barniz, maquina donde se produce entre otros muchas características en función de cómo sea el proceso de creación de la tapa.

```

<?xml version="1.0" standalone="no" ?>
<BizDataObj Name="d_ga_pd_shell_pdf" Description="" Class="d_ga_pd_shell" DBName="gualaerp" Table="ga_pd_shell"
SearchRule="" SortRule="" OtherSQLRule="" Uniqueness="" Stateless="N" IdGeneration="Identity" CacheLifeTime="0">
  <BizFieldList>
    <BizField Name="Id" Column="id" Length="4" Type="Number"/>
    <BizField Name="pd_id" Column="pd_id" OnAudit="N" Length="4" Required="Y" Type="Number"/>
    <BizField Name="shell_id" Column="shell_id" OnAudit="Y" Length="40" Required="Y" Type="Text"/>
    <BizField Name="shell" Join="shell" Column="shell"/>
    <BizField Name="hoja" Join="shell" Column="hoja"/>
    <BizField Name="peso_hoja" Join="shell" Column="peso_hoja"/>
    <BizField Name="pasadas_barniz" OnAudit="Y" Column="pasadas_barniz" Required="N" Type="Number"/>
    <BizField Name="pasadas_litografia" OnAudit="Y" Column="pasadas_litografia" Required="N" Type="Number"/>
    <BizField Name="pasadas_serigrafia" OnAudit="Y" Column="pasadas_serigrafia" Required="N" Type="Number"/>
    <BizField Name="pasadas_offset" OnAudit="Y" Column="pasadas_offset" Required="N" Type="Number"/>
    <BizField Name="pasadas_HotStamping" OnAudit="Y" Column="pasadas_HotStamping" Required="N" Type="Number"/>
    <BizField Name="pasadas_Tampografia" OnAudit="Y" Column="pasadas_Tampografia" Required="N" Type="Number"/>
    <BizField Name="pasadas_esmaltado" OnAudit="Y" Column="pasadas_esmaltado" Required="N" Type="Number"/>
    <BizField Name="wetwet" OnAudit="Y" Column="wetwet" Required="N" Type="Number"/>
    <BizField Name="maquina_id" OnAudit="Y" Column="maquina_id" Required="N" Type="Number"/>
    <BizField Name="identificador" Column="codigo" Join="maquinas"/>
  </BizFieldList>
  <TableJoins>
    <Join Name="shell" Table="ga_shell" Column="id" ColumnRef="shell_id" JoinType="INNER JOIN"/>
    <Join Name="maquinas" Table="ga_prod_maquina" Column="id" ColumnRef="maquina_id" JoinType="LEFT OUTER JOIN"/>
  </TableJoins>
</BizDataObj>

```

Figura 27 – Diseño XML 2

### 3.6 Diseño pantallas

El diseño de las interfaces gráficas (páginas web) de la plataforma está orientado a facilitar al máximo la navegación a través de la misma para que sea lo intuitiva y sencilla posible. En este sentido, he estructurado la página en 3 partes principales. Un menú principal de pestañas (1), un menú secundario de pestañas (2) y el frame principal (3) de la página web donde está el contenido principal de la aplicación. El motivo de estructurar la página en dos menús de pestañas en vez de uno, es dejar la estructura montada para que pueda ser ampliada si se desean añadir otros módulos o funcionalidades a la aplicación.



Figura 28 – Diseño pantalla

## 1 Pestanya superior (tab\_principal.xml)

```
<?xml version="1.0" standalone="no" ?>
<Tabs Name="PRINCIPAL" class="HTMLTabs" TemplateFile="tabs_desplegable_ocultar.tpl">
  <TabViews>
    <View Name="PDs" Module="EPD" URL="controller.php?view=epd.v_Home" View="epd.v_Home" security="epd.v_Home" View:
    <View Name="Informe_epd" Parent="PDs" security="epd.reporte.view.v_reporte_inicio" URL="controller.php?view=ep
    <View Name="Solicitud" Parent="PDs" security="ga_pd_solicitud.access" URL="controller.php?view=epd.pd_solicitud
    <View Name="Solicitud_Generica" Parent="PDs" security="ga_pd_solicitud_generica.access" URL="controller.php?vi
    <View Name="PD" Parent="PDs" security="ga_pd.access" URL="controller.php?view=epd.pd.view.ga_pd_list" ViewSet="
    <View Name="Pdc" Parent="PDs" security="ga_pdc.access" URL="controller.php?view=epd.pdc.view.ga_pd&form=epd
    <View Name="PDL" Parent="PDs" security="pdc_cambiado.access" URL="controller.php?view=epd.cambios.view.pdc_camb
    <View Name="Ayuda_pd" Parent="PDs" security="epd.ayuda.view.ayuda" URL="controller.php?view=epd.ayuda.view.ayu

    <View Name="Inicio" Module="DESKTOP" security="desktop.v_Frontpage" URL="controller.php?view=desktop.v_Frontpa
  </TabViews>
</Tabs>
```

Figura 29 – Código menú superior

## 2 Menu arbol izquierda(menu\_sitemap.xml)

```
<?xml version="1.0" standalone="no" ?>
<Tree Name="Tree" Class="HTMLTree" Template="" URL="">
  <Node security="desktop.v_Frontpage" URL="controller.php?view=desktop.v_Frontpage" Target="" Caption="Inicio"/>

  <!--Access de PDs OK-->
  <Node security="epd.v_Home" Module="EPD" URL="controller.php?view=epd.v_Home" Target="" Caption="PDs">
    <Node URL="controller.php?view=epd.reporte.view.v_reporte_inicio" security="epd.reporte.view.v_reporte_inicio" Ta
    <Node URL="controller.php?view=epd.reporte.view.report_pd_general" Target="" security="ga_pd_general_report
    <Node URL="controller.php?view=epd.reporte.view.report_pd_no_escaneada" Target="" security="ga_pd_no_escane
    <Node URL="controller.php?view=epd.reporte.view.report_pd_pdesarrollo" Target="" security="ga_pd_pendiente_
    <Node URL="controller.php?view=epd.reporte.view.report_pd_pjecucion" Target="" security="ga_pd_plazo_ejecuc
    <Node URL="controller.php?view=epd.reporte.view.report_pd_paprobacion" Target="ga_pd_pendiente_aprobacion_r
    <Node URL="controller.php?view=epd.reporte.view.report_pd_tinta" Target="" security="ga_pd_tinta_dl_reporte
    <Node URL="controller.php?view=epd.reporte.view.report_pd_tinta_lito" Target="" security="ga_pd_tinta_lito_
    <Node URL="controller.php?view=epd.reporte.view.report_pd_barniz" Target="" security="ga_pd_barniz_reporte.
    <Node URL="controller.php?view=epd.reporte.view.report_pd_cambios_solicitados" Target="" security="ga_pd_ca
    <Node URL="controller.php?view=epd.reporte.view.report_pd_cantidad_barniz_tinta" Target="" security="ga_pd_
  </Node>
  <Node URL="controller.php?view=epd.pd_solicitud.view.ga_pd_solicitud_list" Target="" security="ga_pd_solicitud.ac
  <Node URL="controller.php?view=epd.pd_solicitud.view.ga_pd_solicitud_generica_list" Target="" security="ga_pd_sol
  <Node URL="controller.php?view=epd.pd.view.ga_pd_list" Target="" security="ga_pd.access" Caption="Gestion de PD
    <Node URL="controller.php?view=epd.cambio_masivo.view.ga_pd_barniz_cm" Target="" security="ga_cambio_masivo
  </Node>
  <Node URL="controller.php?view=epd.pdc.view.ga_pd&form=epd.pdc.form.f_ga_pd_list" Target="" security="ga_pdc.
  <Node URL="controller.php?view=epd.cambios.view.pdc_cambiado" Target="" security="pdc_cambios.access" Caption="Ap
  <Node URL="controller.php?view=epd.ayuda.view.ayuda" Target="" security="epd.ayuda.view.ayuda" Caption="Ayuda"/>
</Node>
```

Figura 30 – Código menú árbol lateral

## **4. TECNOLOGIAS APLICADAS**

### **4.1 Apache**

APACHE  
HTTP SERVER



#### **¿Qué es un servidor web?**

Un servidor web como su nombre lo indica, es un software instalado en una computadora con todas las condiciones necesarias para servir o entregar páginas web que le sean solicitadas por un navegador, asegurando que se muestren y representen todos los elementos necesarios para su correcto funcionamiento y visualización.

Existen varios tipos de servidores web, Apache es un software de código abierto, libre de uso y totalmente configurable, es en este momento el más utilizado en la red, ya sea en plataformas Linux o Windows Al instalarlo en nuestra PC dispondremos de un servidor completo con todos los requisitos para ejecutarlo de forma local.

#### **Ventajas que nos aporta instalar el servidor Apache en el equipo**

Con la instalación de Apache es posible disponer en nuestro PC de un pequeño servidor que nos posibilitará entre otras tareas:

- 1- Probar y ver las páginas web como verdaderamente van a mostrarse desde internet antes de subirlas a un host o servidor en la red útil e indispensable si tienes o vas a crear tu sitio por modesto que este sea.
- 2- Crear mediante el modulo Virtual Host múltiples sitios web en nuestra PC, que podemos descargar con wget y acceder a ellos igual que en la red pero esta vez de forma local.
- 3- Poder ver localmente páginas web hechas en lenguaje php.
- 4- Servir nuestras páginas o sitio web directamente a internet, a los que puede acceder y conectarse cualquier persona desde el exterior, en este caso lógicamente el funcionamiento del servidor estará limitado al tiempo que tengamos funcionando la PC y a las posibilidades de nuestra conexión. Puede constituir una experiencia muy alentadora para cualquier aficionado, esta posibilidad da la ventaja de que no es necesario depender de ninguna compañía ni servidor remoto para subir a la red el contenido que queremos mostrar. Es como montar una pequeña estación de radio y empezar a transmitir, (una similitud) pero en este caso el alcance es global.
- 5- Puede actuar como intermediario entre nuestra PC e internet lo que nos da varias ventajas en el ámbito de la seguridad.
- 6- A través de él podemos servir internet a varias PC conectadas en una red local.
- 7- Es posible activar un módulo que permite guardar en cache todas las páginas cargadas lo que mejorará el rendimiento de nuestra navegación.

### 4.1.1 Como instalar el Servidor Apache

La instalación es sencilla, descarga de Apache.org la última versión para Windows u otro sistema operativo, desde la siguiente ruta: <http://httpd.apache.org/download.cgi#apache24>

#### Nuevo Apache HTTP Server 24

En febrero de 2012, coincidiendo con el 17 aniversario del lanzamiento de Apache, ha sido presentada la nueva versión 24, se dice que es la mejor hasta la fecha. Entre los cambios y características están un menor consumo de recursos, reducción de la memoria usada, soporte I/O (Entrada/Salida) asíncrona, configuración de proxy dinámico inverso y soporte de ajuste detallado del caché para servidores.

Más información en: <http://httpd.apache.org>

Creamos dos carpetas en la unidad C, la primera de nombre Apache2 y la segunda servidor\_web. Descomprimos el archivo descargado y ejecutamos, seguimos los pasos de la instalación y de los datos que nos piden solo se escoge el destino de la instalación, que será la carpeta que creaste en C:\Apache2, los otros datos los dejamos de la forma predeterminada para configurarlos más tarde.

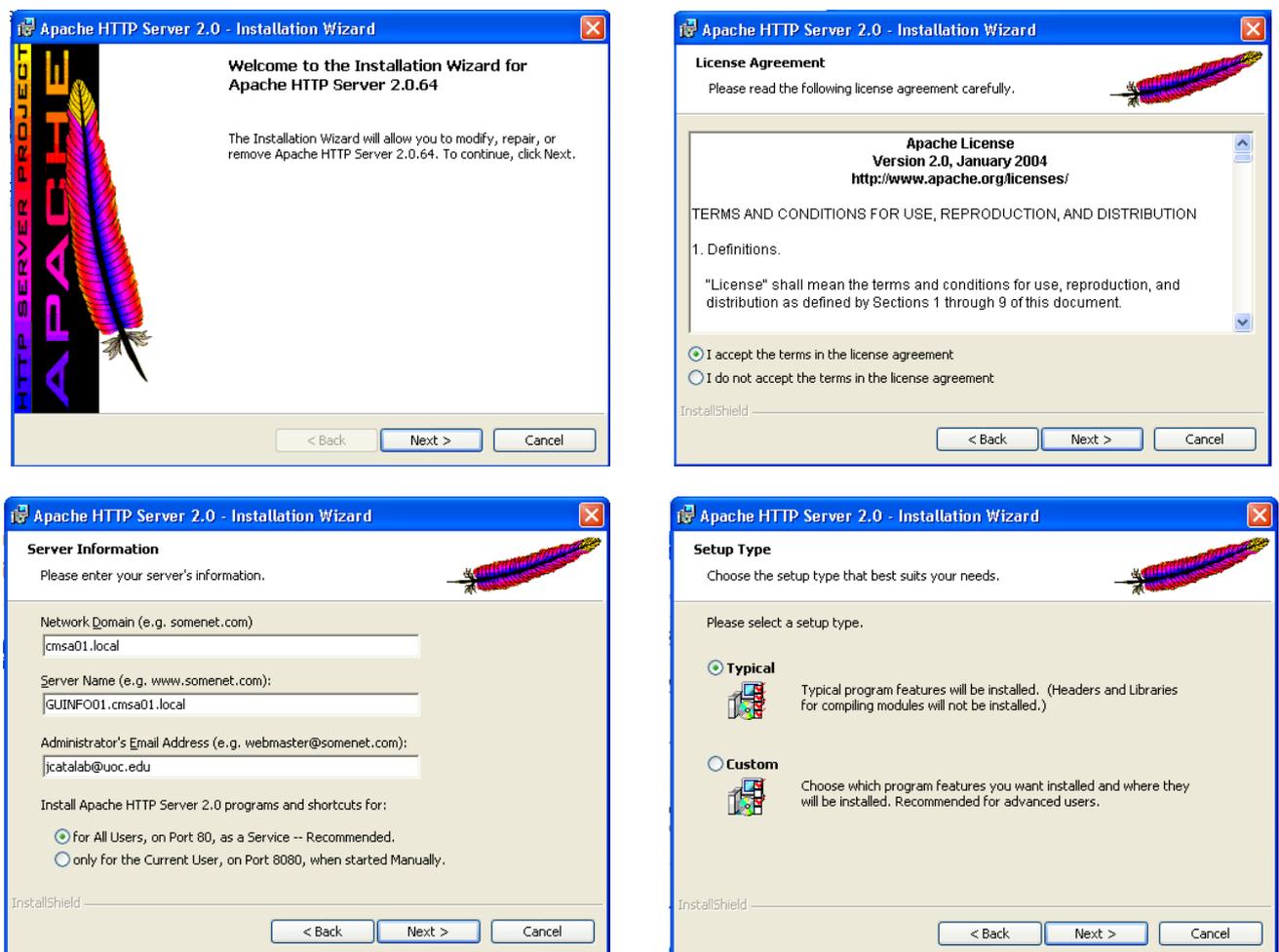


Figura 31 – Instalación Apache

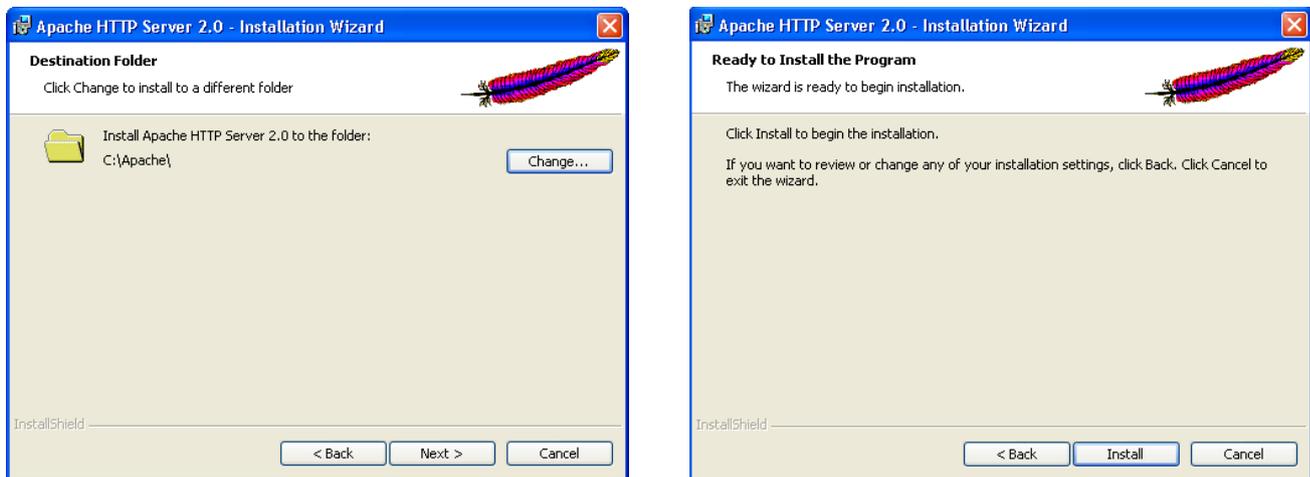


Figura 31 – Instalación Apache

El programa al instalarse crea un icono en el área de notificación que te permitirá: iniciar, detener y reiniciar Apache; tienes que tener en cuenta que cualquier cambio que hagas en el archivo de configuración no tendrá efecto hasta que reinicies el servidor.



Figura 32 – Instalación Apache

#### 4.1.2 Como configurar el Servidor Apache

Toda la configuración para el funcionamiento de Apache se guarda en un archivo de texto nombrado: httpd.conf que se encuentra en la ruta C:\Apache2\conf, lo podemos editar en cualquier editor de texto, aunque recomiendo el Notepad++.

##### Notepad++

Es muy recomendable utilizar este editor de texto. Se trata de software libre, muy útil y recomendable para cualquier tipo de usuario. Se puede descargar Notepad++ desde el siguiente la siguiente dirección web: <http://notepad-plus-plus.org/>

A continuación hay que abrir el archivo httpd.conf y editar manualmente el fichero para habilitar el php, configurar los puertos de escucha, etc.

Todas las líneas que comienzan con el símbolo # son comentarios, explican en cada sección las distintas opciones pero se encuentran en inglés.

La línea 46 **Listen** indica el puerto y dirección IP por el que el servidor va a recibir las peticiones, se puede usar de las siguientes maneras:

- 1- El servidor va recibir peticiones solo de la misma PC: Listen localhost:80
- 2- Recibirá peticiones de otras máquinas en una red local: Listen 80

```

38 # Listen: Allows you to bind Apache to specific IP addresses and/or
39 # ports, instead of the default. See also the <VirtualHost>
40 # directive.
41 #
42 # Change this to Listen on specific IP addresses as shown below to
43 # prevent Apache from glomming onto all bound IP addresses.
44 #
45 #Listen 12.34.56.78:80
46 Listen 80
47
48 PHPIniDir "C:/PHP5"

```

Figura 33 – Instalación Apache

En la línea 48 **PHPIniDir** inicializamos la ruta de las librerías de PHP en "C:/PHP5"

En la línea 181 **DocumentRoot** es necesario especificar la ruta de la carpeta local que contendrá las páginas y archivos a servir, en tu caso será la carpeta que creaste en C:/servidor\_web, quedaría de la siguiente forma:

DocumentRoot "C:/Apache2/htdocs"

```

176 #
177 # DocumentRoot: The directory out of which you will serve your
178 # documents. By default, all requests are taken from this directory, but
179 # symbolic links and aliases may be used to point to other locations.
180 #
181 DocumentRoot "C:/Apache2/htdocs"

```

Figura 34 – Instalación Apache

La línea 208 **<Directory>** establece los permisos necesarios al directorio anterior, quedaría:

```

191 <Directory />
192     Options FollowSymLinks
193     AllowOverride None
194     Order deny,allow
195     Deny from all
196 </Directory>
197
198 #
199 # Note that from this point forward you must specifically allow
200 # particular features to be enabled - so if something's not working as
201 # you might expect, make sure that you have specifically enabled it
202 # below.
203 #
204 #
205 #
206 # This should be changed to whatever you set DocumentRoot to.
207 #
208 <Directory "C:/Apache2/htdocs">

```

Figura 35 – Instalación Apache

Esta es la configuración con los parámetros esenciales para comenzar a utilizar Apache. Guardamos los cambios realizados y reiniciamos el servidor dando clic en el icono del área de notificación.

## **Comenzar a utilizar Apache**

Copia en la carpeta C:/servidor\_web alguna página web o cualquier archivo y accede a él escribiendo en la barra de direcciones de tu navegador 127.0.0.1 o localhost. En el caso de que sea una página web que estas diseñando la verás cómo realmente se mostrará en Internet.

Cualquier problema del servidor estará reflejado en los logs que guarda en C:\Apache\logs, los puedes abrir con el bloc de notas, en el archivo access.log se registran todos los accesos hechos al servidor, tanto de tu PC como de internet, en error.log se registran todos los errores de su funcionamiento, te ayudará mucho analizarlos periódicamente.

## **Acceder al servidor Apache desde la red local**

Para acceder al servidor desde otra computadora conectada en una red local solo es necesario escribir en la barra de direcciones la dirección IP de la computadora que sirve de host, es decir la que tiene el servidor Apache instalado.

Para conocer la dirección IP necesaria solo abre una ventana de CMD, escribiendo en Inicio >CMD, escribimos en la ventana que se abre IPCONFIG /ALL y presionamos la tecla "Enter", buscamos la línea "Puerta de enlace", el número a continuación es la dirección IP del proxy.

En caso de que el servidor escuche en otro puerto diferente al 80 (predeterminado) escribe: "dirección ip:puerto", por ejemplo: 192.168.13:8080

En caso de conflictos, si tratamos de conectarnos a un equipo usando una red local, verificamos lo siguiente:

- La dirección IP del equipo al que deseas conectarte
- Si Apache está escuchando en el puerto al que se efectúa la petición
- Si el firewall de Windows está bloqueando la conexión
- Si se recibe un mensaje de error con el código 403 significa que no se cuenta con los permisos necesarios para acceder al directorio, en ese caso lo establecemos de la siguiente forma:

```
<Directory " ruta al directorio">  
Options Indexes FollowSymLinks  
AllowOverride None  
Order allow,deny  
Allow from all  
</Directory>
```

## **Como conectarse a internet a través de Apache desde tu PC**

Para que la conexión a internet pase a través del servidor será necesario configurarlo como un proxy forward. Para eso, en el archivo de configuración "httpd.conf" des comentamos, (quitamos el signo #) de las siguientes líneas:

```
LoadModule proxy_module modules/mod_proxyso
```

```
LoadModule proxy_http_module modules/mod_proxy_httpso
```

Después agregamos en el final del archivo la siguiente línea: ProxyRequests On

Cerramos y guardamos los cambios

Accedemos a las “Opciones de internet” mediante el “Panel de Control”, en la pestaña “Conexiones” y pulsamos el botón Configuración de LAN y marcamos la casilla “Usar un servidor Proxi”, escribimos en “Dirección”: 127001 y en Puerto: 80 o el que vayamos a usar Aceptamos en todas las ventanas A continuación, reiniciamos la máquina.

Lo anterior se aplica si usamos el navegador Internet Explorer y Google Chrome, si usamos Firefox las opciones anteriores las tenemos que ingresar en:

Opciones >Configuración >Configurar como Firefox se conecta a Internet

A partir de ahora toda la conexión pasa a través de Apache, ya sean direcciones locales o externas.

### **Como guardar en la cache del servidor web las páginas**

Para guardar en la cache del servidor web las páginas web hacemos lo siguiente:

Creamos una carpeta en la unidad “C:\” de nombre cacheroot, será la que almacene los archivos de la cache, también podemos crearla en otra unidad, en ese caso tendríamos que indicar su ruta en el archivo de configuración.

En el archivo httpd.conf des comentamos, (quitamos el signo #) de la siguiente línea:

```
LoadModule expires_module modules/mod_expireso
```

Y agregamos las siguientes líneas:

```
LoadModule cache_module modules/mod_cacheso
```

```
LoadModule disk_cache_module modules/mod_disk_cacheso
```

Copiamos y pegamos las siguientes líneas de código al final del archivo:

El valor ExpiresDefault A604800 especifica el tiempo en segundos que persistirá tu cache, podemos modificarlo de acuerdo a nuestras necesidades. Todos los demás valores los podemos ajustar a nuestra conveniencia.

Podemos leer las siguientes páginas en la carpeta de manuales:

```
C:/Apache/manual/mod/mod_disk_cachehtml
```

```
C:/Apache/manual/mod/mod_expireshtml
```

Reiniciamos el servidor.

### **Servir páginas en internet y permitir el acceso al servidor desde el exterior**

Para servir páginas web en internet, primero es necesario configurar la línea 147 ServerName, Nombre del servidor, tenemos las siguientes opciones:

1- Disponemos de una conexión a internet fija, sustituimos localhost por la dirección IP que tenemos asignada seguida de dos puntos y el puerto a usar, el predeterminado en el protocolo http es el 80 Si nos

conectamos a través de un router sería la dirección IP de este ya que la nuestra es una dirección local en este caso.

2- Tenemos una conexión a internet dinámica, lo más común, significa que cada vez que nos conectamos nuestro proveedor del servicio de internet (ISP) nos asigna una dirección IP diferente. En este caso cada vez que nos instalemos necesitamos ingresar en nuestro archivo `httpd.conf` la dirección IP asignada en ese momento, lo que es bastante incómodo.

3- La otra opción es acceder al servicio que ofrece DynDNS que va a sincronizar la dirección IP que nos asignen con nuestro servidor. Podemos leer más a continuación de un extracto de Wikipedia.

### *Usar el Servicio de DynDNS*

DynDNS (Dynamic Network Services, Inc.) es una compañía de Internet de Estados Unidos dedicada a soluciones de DNS en direcciones IP dinámicas Ofrece servicios gratuitos de redirección a IP de subdominios de una gran lista de nombres disponibles, ofreciendo a particulares la oportunidad de crear un servidor en Internet gratuitamente con una dirección como 'wikipediadynDNScom', Debe ser actualizado mediante algún cliente en la máquina del cliente, tales como el que la misma compañía ofrece

Podemos acceder a DynDNS, registrarnos y a continuación escogemos un nombre de dominio por el que podremos acceder a nuestro servidor desde internet, tenemos que descargar un cliente de actualización: <https://www.dyndns.com/support/clients.html> que ellos nos ofrecen (es sencillo sólo pesa alrededor de 700 KB) y lo instalamos en nuestro PC.

En este caso, en nuestro archivo `httpd.conf` no tenemos que modificar nada, dejamos intacto el `ServerName` con el valor `localhost:80` , a partir de ahora DynDNS se encargará de sincronizar nuestra dirección IP.

El Nombre del servidor lo agregamos al final de nuestro archivo `httpd.conf` la siguiente línea:

Listen 80 para acceder mediante el puerto 80 que es el predeterminado en la navegación web.

Hay proveedores de internet que cierran este puerto, podemos probar si ese es el caso tuyo accediendo a esta página, previo registro:

<https://accountdyncom/entrance/?return=%2Ftools%2Fopenport.html>

Probamos puertos introduciendo nuestra dirección IP y el puerto, si es así cambia probando otro puerto, por ejemplo el 8080 y específícalo en el archivo `httpd.conf` de esta forma: Listen 8080

Es todo, en caso de que así funcione para acceder desde el exterior a nuestra página, tendríamos que escribir en el navegador: `direccionIP:8080/`

Si hemos instalado el cliente de actualización de DynDNS, no tendría que ser la necesariamente la dirección IP, si no la dirección asignada por ese servicio.

### *Como agregar tu página o blog al índice de Google*

Solo te faltara agregar nuestro sitio en Google u otros buscadores, para eso es interesante leer la siguiente página:

<http://norfipc.com/articulos/google.html>

## 4.2 SQL Server



### ¿Qué es SQL?

El lenguaje de consulta estructurado o SQL (por sus siglas en inglés Structured Query Language) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas. Una de sus características es el manejo del álgebra y el cálculo relacional que permiten efectuar consultas con el fin de recuperar de forma sencilla información de interés de bases de datos, así como hacer cambios en ella.

### ¿Qué es SQL Server?

Microsoft SQL Server es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. Sus lenguajes para consultas son T-SQL y ANSI SQL. Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

SQL Server 2008 es un elemento fundamental de la Plataforma de Datos de Microsoft, capaz de gestionar cualquier tipo de datos, en cualquier sitio y en cualquier momento. Nos permite almacenar datos de documentos estructurados, semi estructurados o no estructurados como son las imágenes, música y archivos directamente dentro de la base de datos. SQL Server 2008 nos ayuda a obtener más rendimiento de los datos, poniendo a su disposición una amplia gama de servicios integrados como son consultas, búsquedas, sincronizaciones, informes y análisis. Sus datos pueden almacenarse y recuperarse desde sus servidores más potentes del Data Center hasta los desktops y dispositivos móviles, permitiéndole tener un mayor control sobre la información sin importar dónde se almacena físicamente.

SQL Server 2008 nos permite utilizar sus datos en aplicaciones a medida desarrolladas con Microsoft® .NET y Visual Studio y también desde su propia Arquitectura Orientada a Servicio (SOA) y los procesos empresariales empleando Microsoft® BizTalk® Server.

Además, las personas que gestionan la información pueden acceder directamente a los datos con las herramientas que utilizan habitualmente como Microsoft® Office 2007. SQL Server 2008 nos ofrece una plataforma de datos, fiable, productiva e inteligente para cubrir todas las necesidades.

### Otras características o funcionalidades según el fabricante

#### **Business Intelligence**

SQL Server 2008 es una plataforma escalable de Business Intelligence optimizada para la integración de datos, elaboración de informes y análisis.

#### **OLTP**

SQL Server 2008 es el motor de base de datos escalable y de alto rendimiento que necesitan las aplicaciones de misión crítica con las mayores exigencias de disponibilidad y seguridad. SQL Server reduce el coste total de propiedad gracias a su mayor manejabilidad en entornos corporativos.

#### **Data Warehouse**

SQL Server nos ofrece una plataforma de data warehouse completa y escalable que nos permite integrar datos dentro del DW más rápidamente, escalar y gestionar volúmenes de datos y usuarios cada vez mayores facilitando a todos las vistas de síntesis que necesitan.

## Desarrollo de aplicaciones

SQL Server 2008 constituye el eje central de una plataforma completa de programación de datos que nos permite acceder y manipular datos críticos de negocio desde toda clase de dispositivos, plataformas y orígenes de datos.

### 4.3 Openbiz-Cubi Framework



#### ¿Qué es un Framework PHP?

El uso de un Framework facilita el desarrollo de software, permitiendo a los diseñadores y programadores pasar más tiempo identificando requerimientos de software que tratando con los detalles de bajo nivel para obtener un sistema funcional. A esto le sumamos la capacidad de extenderse sin prejuicios para diversificar las aplicaciones.

En otras palabras, un Framework establece una manera sencilla y rápida de desarrollar aplicaciones, un sistema basado en soluciones predefinidas y reutilizables capaces de construir una aplicación a través de una modelo de trabajo que libera al programador de la escritura de código de bajo nivel.

#### ¿Qué es Openbiz-Cubi?

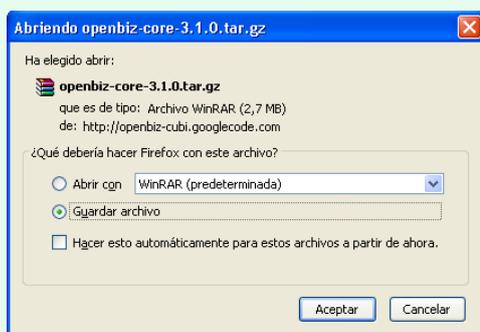
Openbiz es un framework o herramienta con la que se pueden desarrollar fácilmente y con bastante calidad todo tipo de aplicaciones web profesionales. Programada en PHP, está especialmente diseñada para consultoras de software y empresas de IT.

#### Características

Género: Plataforma de desarrollo  
Lenguaje de programación: PHP  
Sistema operativo: Multiplataforma  
Licencia: BSD  
Multidioma: Sí  
Sitio Web: [www.phpopenbiz.org](http://www.phpopenbiz.org)

#### Descarga del Framework Openbiz

El Framework de Openbiz se puede descargar del siguiente enlace:



<http://www.openbiz.me/developer/framework.php>

### 4.3.1 Arquitectura Openbiz

El objetivo del Openbiz framework es de hacer el diseño, el desarrollo y el mantenimiento de aplicativos web de fáciles y rápidos La innovación principal en la arquitectura Openbiz es su núcleo de metadata (metadatos) Esto quiere decir que los objetos de Openbiz son construidos basados en la descripción en sus archivos metadata Construir una aplicación significa diseñar y desarrollar archivos de metadatos la mayor parte del tiempo debido a la naturaleza de auto explicación de lenguaje XML, la aplicación web es fácil para mantener Mientras tanto Openbiz es un framework extensible debido a la naturaleza extensible de XML.

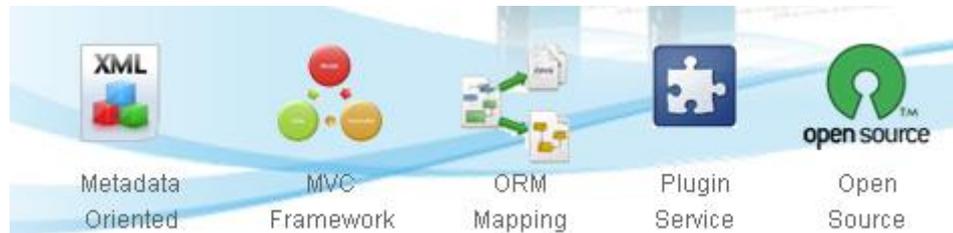


Figura 36 – Arquitectura Openbiz

### Objetos del núcleo de Openbiz

Cualquier aplicación se compone de dos partes (dos capas) – la parte trasera y la frontal La lógica principal de la aplicación típicamente se desarrolla en la parte trasera (back end), mientras el interfaz de usuario está en la parte o capa frontal de la aplicación En Openbiz, en la parte trasera o capa back end es donde hay los Objetos de datos o los servicios de objetos La parte o capa frontal tendrá los objetos formulario y las vistas de los objetos.

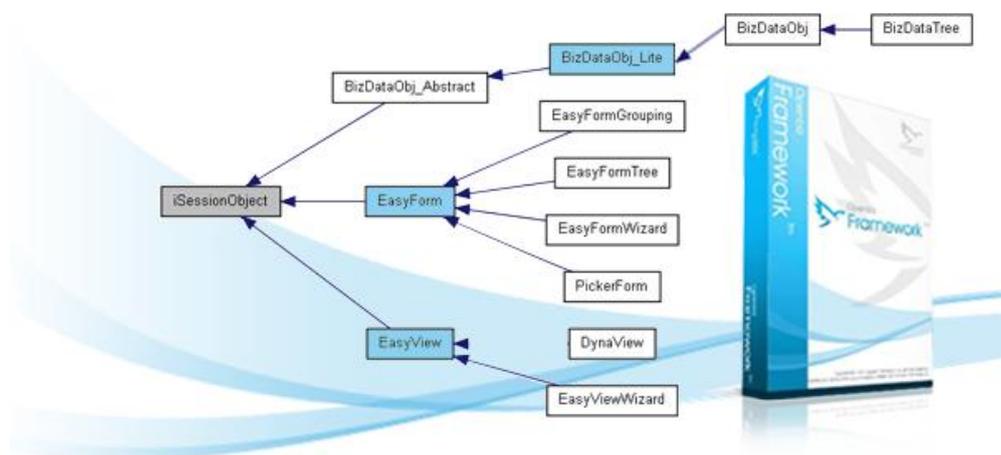


Figura 37 – Arquitectura Openbiz

### Objeto de datos (Data Object)

Un objeto de datos o “DO” es la unidad básica de datos Los Openbiz DO mapean las tablas de la base de datos y las relaciones con los objetos Se encapsula la operación CRUD (create, read, update and delete) en el objeto y también se proporciona la interfaz para búsquedas avanzadas. Un DO contiene un cierto número de campos. Habitualmente un DO se mapea hacia las tablas de la base de datos, y un campo se mapea hacia una tabla columna o expresión SQL.

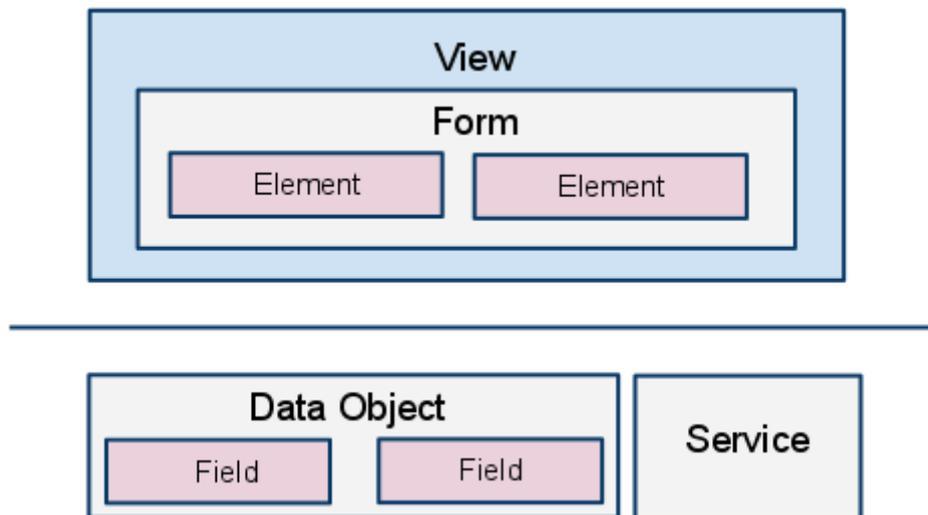


Figura 38 – Openbiz DO

### **Servicio Openbiz (Service)**

Un servicio Openbiz es una unidad que contiene la implementación de una lógica de la aplicación o una serie de funciones. Un ejemplo de servicios Openbiz son las autenticaciones de usuarios para acceder a la aplicación o servicios de email. A continuación vemos la “pantalla de Login” de nuestra aplicación:



Figura 39 – Login de aplicación

### **Formulario (Form)**

Un formulario es una unidad de UI (user interface) que contiene una serie de elementos relacionados entre sí. Puede ser un formulario estándar HTML, una tabla HTML con una barra de herramientas o de paginación, una lista de imágenes, o algo similar. Un formulario contiene un número de elementos que pueden ser controles HTML simples o avanzados. Habitualmente se usan formularios mapeados a objetos DO, y un elemento mapea a un campo del DO.

### **Vista (View)**

Una vista es una página web generada. Una vista es un contenedor de formularios. Podemos considerar una vista (página web) como un mosaico y un formulario sería una pieza de él.

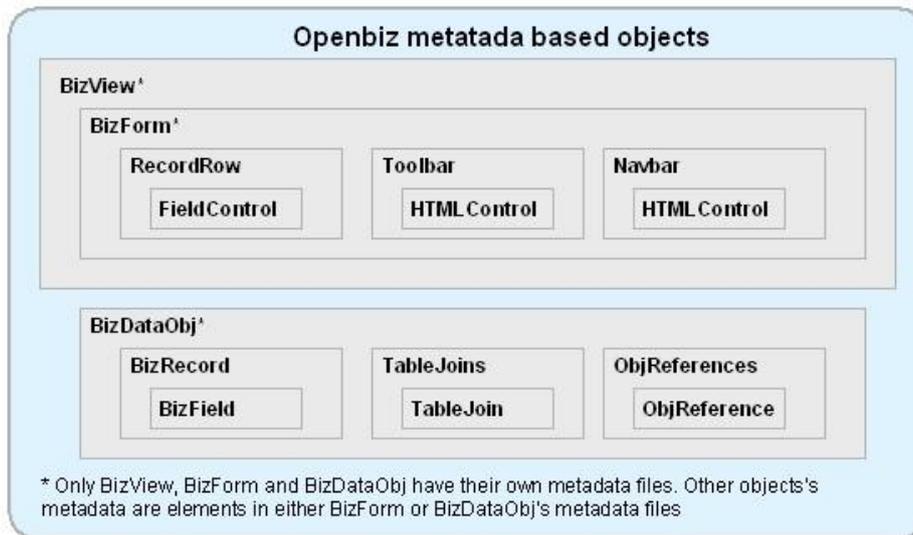


Figura 40 – Esquema estructura Openbiz

### **Objetos del núcleo (Core Objects) y metadatos**

Los objetos del núcleo Openbiz objects son definidos por los metadatos Openbiz. El siguiente extracto de código representa una parte parcial de un DO de metadatos.

```
<?xml version="1.0" standalone="no"?>
<BizDataObj Name="UserDO" class="BizDataObj" DBName="Default" Table="user"
SearchRule="" SortRule="" OtherSQLRule="" IdGeneration="Identity">
  <BizFieldList>
    <BizField Name="Id" Column="id" Type=""/>
    <BizField Name="username" Column="username" Type=""/>
    <BizField Name="password" Column="password" Type=""/>
    <BizField Name="email" Column="email" Type=""/>
  </BizFieldList>
</BizDataObj>
```

Openbiz lee los metadatos y crea objetos online desde el generador de objetos (ObjectFactory).

### **Características de la arquitectura Openbiz**

#### **Diseño orientado a objetos multicapa**

Openbiz está diseñado con una arquitectura orientada a objetos multicapa La aplicación Openbiz está organizado en tres capas: capa de presentación, capa de Lógica de la aplicación y capa de integración de datos:

- La capa de presentación esta implementada por Openbiz Vista y Formulario Openbiz tiene librerías adicionales de JavaScript que comunican con servidores de presentación de objetos
- La capa de Lógica esta implementada con Openbiz DO así como también Servicios Openbiz
- El paquete Zend\_DB maneja la tercera capa de integración de datos

Zend Framework 2 es un framework de software libre para el desarrollo de servicios y aplicaciones web usando PHP 5.3+ Zend Framework 2 usa 100% código orientado a objetos y utiliza las últimas novedades del PHP 5.3: *namely namespaces, late static binding, lambda functions y closures*

<http://framework.zend.com/>

## **Openbiz MVC**

Una de las ventajas clave de Openbiz es que es un framework que sigue el modelo Model-View-Controller (MVC) patrón de diseño. Esto hace que las aplicaciones Openbiz sean mucho más manejables ya que su código de presentación (Vista) está limpiamente separado de los datos de la aplicación y la Lógica (Model). Todas las interacciones del usuario son manejables desde el controlador de la parte frontal (front-end controller).

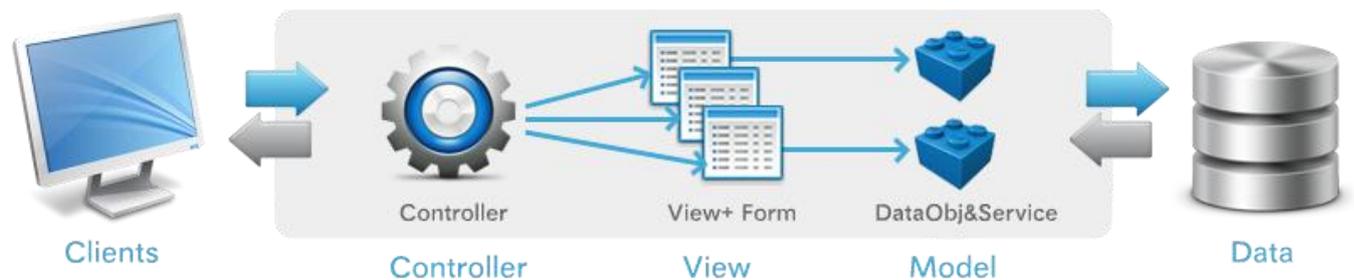


Figura 41 – Modelo MVC

Comparando Openbiz MVC con otros MVC frameworks existentes en el mercado, como JSF y Struts, Openbiz es más parecido a JSF ya que ambos son arquitecturas basadas en componentes. La capa de Vista de Openbiz integra componentes de Vista, Formulario y Elemento. Estos componentes son accesibles durante el proceso de petición.

## **Openbiz ORM**

Openbiz implementa las características necesarias de mapeo relacional de objetos (ORM) para permitir a los objetos DO, que representan los datos, las relaciones de las tablas de la base de datos. A continuación describimos las características de Openbiz ORM:

### Mapeo flexible

- Tabla de soporte para el mapeo de objetos Tabla por clase, N tablas para 1 clase
- Soporte para el mapeo de relaciones: Muchos a uno, uno a muchos, muchos a muchos, personalizados

### Utilidades para consultas (query)

- Soporte SQL para query language
- Soporte SQL funciones y operadores
- Soporte SQL para agregar funciones
- Soporte por grupo, pudiendo ordenar

### Utilidades para metadatos

- XML metadatos describen el mapeo

## **Abstracción de la Base de datos**

Openbiz DO se puede conectar hacia todo tipo de base de datos relacionales soportadas por Zend\_DB quien proporciona a la base de datos la capa de abstracción por encima del PDO y el cliente nativo de la base de datos. Los desarrolladores pueden usar Openbiz DO API para la mayoría de las operaciones así como también Zend\_DB API para funciones avanzadas.

Para conectar con diferentes tipos de bases de datos, el usuario solo necesita especificar la conexión en la application.xml bajo la carpeta raíz de la instalación. Openbiz DO invocará los controladores (drivers) necesarios para conectar al servidor de la base de datos. Openbiz soporta actualmente MySQL, MSSQL,

Oracle, PostgreSQL, Sqlite entre otros. Cada Openbiz DO puede tener sus propias referencias a bases de datos. Esta característica habilita las múltiples conexiones a bases de datos en una conexión o una misma página web.

### **Motor de plantillas (Template Engine)**

El Formulario Openbiz (Form) y la Vista (View) usan el motor de plantillas Smarty template engine para dar salida gráfica por defecto. Como Smarty es la más popular de las plantillas, los desarrolladores pueden fácilmente cómo generar páginas Openbiz. En caso de salidas complejas de código en Smarty template, Openbiz permite usar plantillas PHP, para generar páginas incluso más rápidamente que con Smarty.

### **Estructura del código**

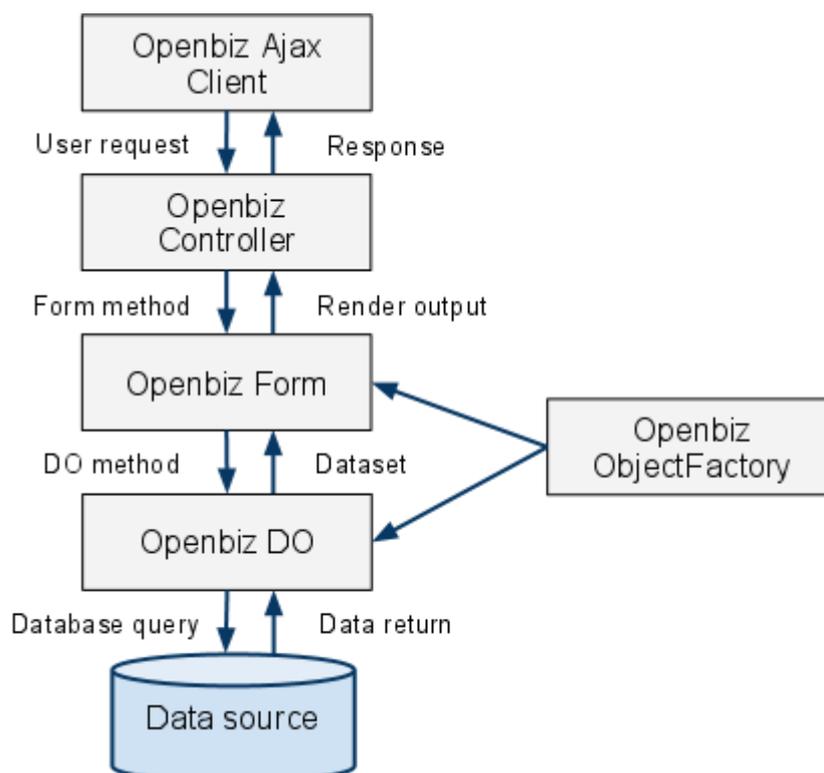
Estructura de librerías del Núcleo Openbiz:

```

openbiz_root/
--- bin/                (openbiz core php source)
----- data/          (data layer classes)
----- private/      (data layer non-public classes)
----- easy/         (presentation layer classes)
----- element/      (html element classes)
----- service/      (openbiz core service classes)
----- util/         (utility helper classes)
--- message/          (openbiz message files)
--- metadata/        (openbiz metadata files)
----- service/      (openbiz service package)
--- others/          (third party libraries)
----- Smarty/       (smarty package)
----- zend/         (Zend framework)
  
```

### **Flujo de ejecución**

El siguiente diagrama, es un típico flujo de ejecución de una consulta de datos hecha por un usuario en una aplicación Openbiz:



### **Openbiz y el tercer grupo de librerías (3rd party libraries)**

Openbiz intentó al reforzar las mejores librerías “extras” del mercado. Las librerías que se usan principalmente en la mayoría de aplicaciones Openbiz son:

- **Zend Framework:**

Openbiz usa Zend Framework en:

- Interacciones Base de datos
- Soporte idiomas
- Validación de datos
- Servicios de Email
- Gestión de Cache
- JSON decode and encode
- Plantillas PHP

- **Smarty:**

Smarty es la principal plantilla motor que utiliza Openbiz Formulario y Vista (Form and View) Para ciertas plantillas que necesitan más complejidad gráfica, Openbiz Formularios utiliza plantillas PHP proporcionadas con Zend Framework. Esta plantilla se utiliza mucho en la aplicación, como muestra el ejemplo a continuación:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<EasyView Name="view.report_pd_pdesarrollo" Class="EasyView" Description="Guala - Reportes - PDs"
Tab="Pendiente Desarrollo" ViewSet="Informe_epd_set" TemplateEngine="Smarty" Module="EPD"
TemplateFile="view_reporte.tpl" >
  <FormReferences>
    <Reference Name="epd.reporte.form.f_ga_pd_list_report_pendiente_desarrollo"/>
  </FormReferences>
</EasyView>
```

Figura 43 – Ejemplo Smarty



Para más información acerca de Smarty, podemos visitar su página web en el siguiente link:

<http://www.smarty.net/>

Su última versión es la Smarty 3.1.12 que salió el pasado 25 de setiembre.

- **Librerías JavaScript**

- Prototype Openbiz Ajax client usa librerías Prototype para heredar clases y comunicaciones Ajax
- jQuery se usa en Openbiz Cubi para elementos UI avanzados

### **Openbiz Cubi Menú**

El menú Cubi Menú es el núcleo de los componentes que gestiona la navegación por la aplicación.

Normalmente, una aplicación web(o sitio web) maneja diferente código de elementos de navegación, como

menús, botones o iconos, estructura de carpetas (Breadcrumb) etc. Esta era la manera cómo funcionaba en aplicaciones con versiones anteriores de Openbiz – añadiendo manualmente menús a la izquierda para cada vista o plantilla.

A continuación, comentamos cómo trabaja ahora el nuevo menú en Openbiz Cubi:

## 1. Definición del menú

- El Menú es una estructura árbol
- Cada parte del manu incluye atributos de navegación, como nombre, titulo, url, icono...

## 2. Presentación del menú en la página web

Como hemos comentado, el menú tiene estructura de árbol. Puede ser presentado como estructura de carpetas (Breadcrumb), mapa de sitio (Sitemap), etiquetas de aplicación (Application tab) o con un menú de navegación en la página web (Navigation menu).

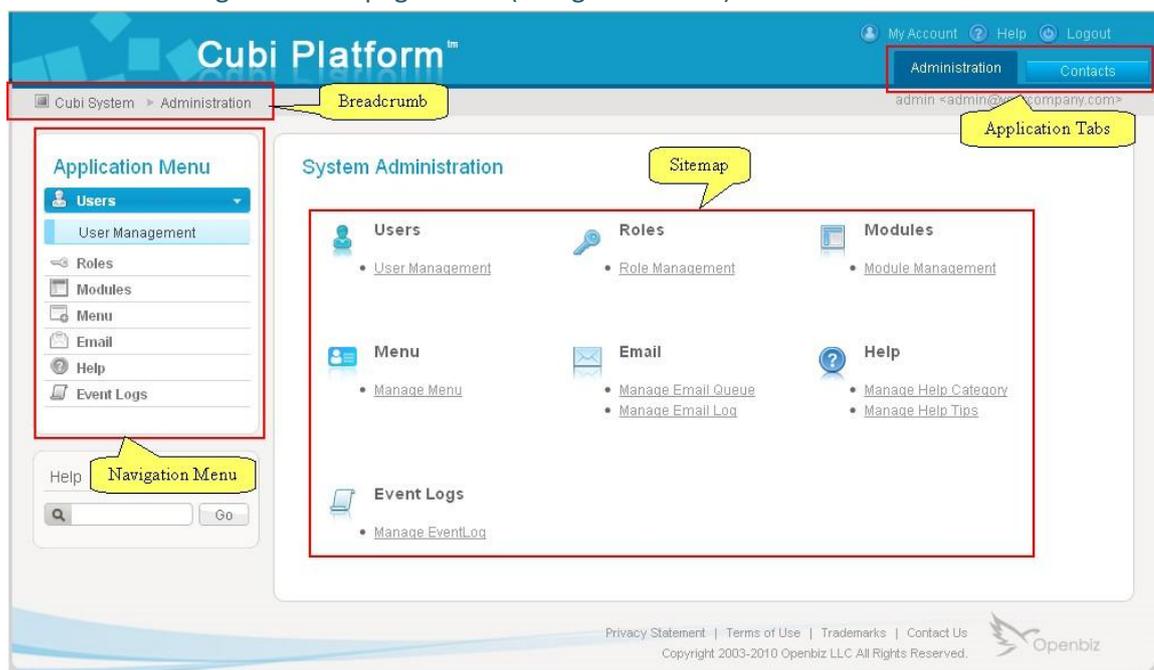


Figura 44 – plataforma Cubi

El siguiente diagrama nos muestra como el menú árbol mapea hacia los elementos del interfaz de usuario (UI).

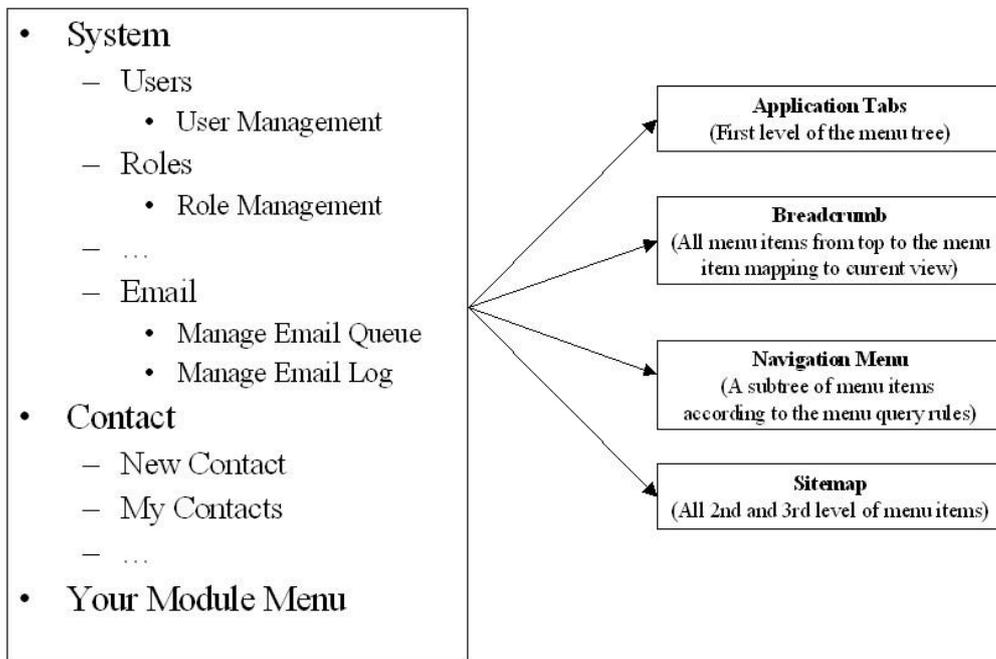


Figura 45 – mapeo interfaz

### 3. Cómo añadir menús en nuestro propio módulo

Cada módulo puede añadir entradas en un menú Cubi. En módulo mod.xml, hay una sección de menús para añadir elementos. La clave para añadir los módulos de menú al sistema son:

- En el elemento superior del menú, indicar Parent="existing\_menuitem". Esto añadirá un menú por defecto.
- Por el contrario si dejamos Parent="" Esto hará este objeto como menú de primer nivel o menú raíz.

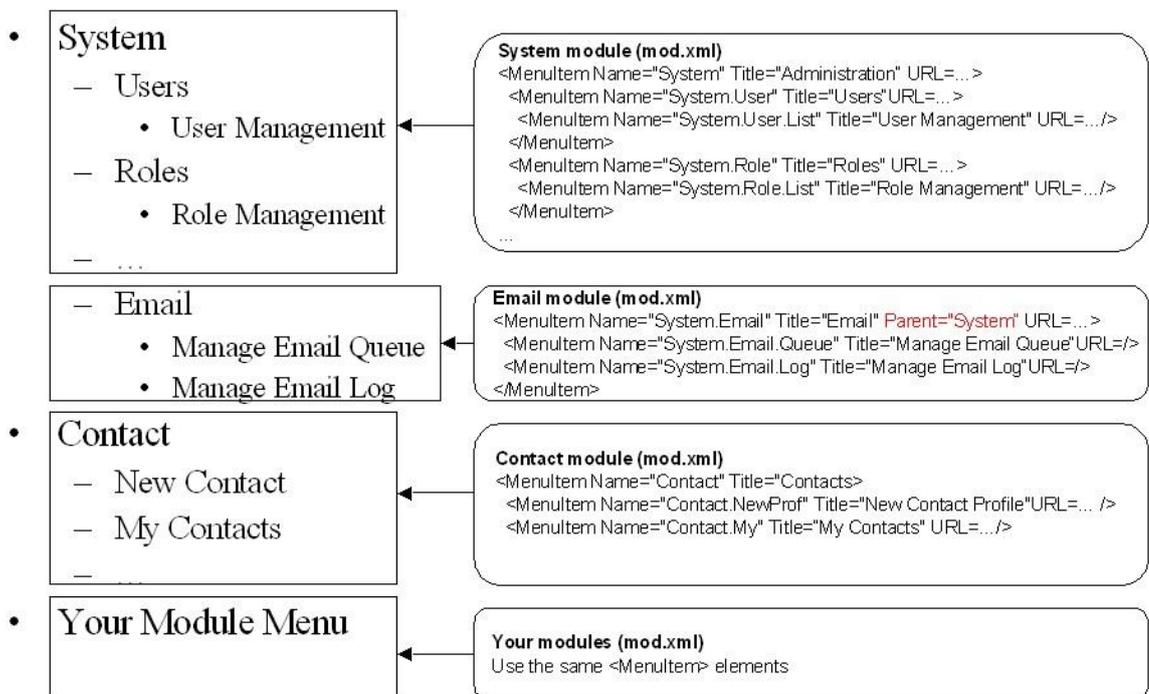


Figura 46 – mapeo interfaz

### 4. Como gestionar el menú en la aplicación Cubi

Una vez el modulo está cargado en el sistema, su menú también es insertado. Entonces podemos ver que los elementos del menú aparecen en el lugar indicado como el menú de navegación o las

etiquetas de la aplicación (application tab). Como administradores de la aplicación, podemos gestionar los elementos del menú en la vista de Gestión de Menús (Menú Management).

### **Módulo Cubi en detalle**

El objetivo del módulo Cubi es permitir que los desarrolladores puedan escribir sus propios componentes y conectarlos a la aplicación Cubi. Así que cada módulo debe tener su propio:

- Módulo de descripción de la información
- Tablas en la base de datos y pre-carga inicial de datos
- Definiciones de los menús Los menús indican a los usuarios dónde acceder y a que funcionalidad
- Módulos dependientes quienes necesitan ser instalados previamente

Veamos ahora en detalle el módulo de gestión de Cubi. En el archivo [mod.xml](#) están todas las descripciones principales Su estructura básicamente tiene 3 partes:

1. ACL
2. Menú
3. Dependencias

Un ejemplo de un mod.xml lo podemos encontrar en `cubi/modules/help/mod.xml`  
Podemos añadir módulos específicos SQL en:

- `modinstallsql` y
- `moduninstallsql`

Cubi proporciona una herramienta de línea de comandos para cargar módulos. Esta herramienta es el comando: `cubi/install/load_module.php`

```
# php load_module.php module_name
```

Para ayudar a preparar una instalación de datos limpia de Cubi, podemos utilizar el comando (script): `cubi/install/create_cubidb.php`

```
# php create_cubidb.php
```

Este script crea una base de datos inicial limpia, llamada "cubi\_install". Podemos verter la base de datos en `cubi/install/cubisql`. La base de datos será usada en el asistente de instalación inicial.

### **Actualizaciones Openbiz**

Cada semana Openbiz avanza en la implementación de nuevas características y la resolución de problemas de programación (bug fixes).

### **4.3.2 Módulos de Openbiz-Cubi**

El framework Openbiz-Cubi tiene bastantes módulos de funcionalidades, entre los cuales destacamos el módulo autenticación de usuarios (Cubi User module) y el de planificación de tareas llamado Cronjob.

#### **Cubi User module**

El módulo de usuario de Cubi, proporciona las siguientes funcionalidades a los usuarios:

- Registro

- Login

- Reset de password, en caso de olvido

### **Planificador de tareas de Cubi (Cronjob)**

Cronjob manager es un planificador de tareas que se ha añadido a Cubi con las siguientes características:

- Proporciona un panel de control para permitir al administrador añadir/corregir/suprimir comandos en formato de Cronjob.
- Una tarea de gestión de scripts que lee y analiza las tareas de la lista de Cronjob, para luego ejecutarlas cuando estén planificadas.



Figura 47 – Menu Cronjob

Una tarea Cronjob definida en Cubi tiene los siguientes atributos:

- Nombre de la tarea
- minuto, hora, día, mes, día de la semana
- Comando: Este es el comando para ejecutar la tarea
- Sendmail: Este campo incluye los emails para enviar una vez completada la ejecución de la tarea
- max\_run: Nos indica cuantas ejecuciones concurrentes. Por defecto este valor es 1, lo que significa que sólo una tarea se puede ejecutar a la vez. Si max\_run=0, entonces no hay límite de ejecución de tareas de manera simultánea.
- Descripción

Cubi proporciona un script para ejecutar las tareas que hay en la lista de la pantalla de gestión de Cronjob. Para iniciar el Cubi Cronjob, tenemos que añadir "cubi/bin/cronjob/cron.php" en la tabla del sistema Cronjob y dejarlo ejecutar cada cierto tiempo, por ejemplo cada minuto.

### Más información

Podemos encontrar más información en el siguiente enlace: <http://en.wikipedia.org/wiki/Cron>

Otros módulos de Openbiz-Cubi son:

- [Cubi User Module](#)
- [Cubi Theme Module](#)
- [Cubi System Module](#)
- [Cubi Session Module](#)
- [Cubi Service Module](#)
- [Cubi Security Module](#)
- [Cubi Package Module](#)
- [Cubi Menú Module](#)
- [Cubi Help Module](#)
- [Cubi EventLog Module](#)
- [Cubi Email Module](#)
- [Cubi Cache Module](#)
- [Cubi Attachment Module](#)
- [Cubi Backup Module](#)

### Más información

Podemos encontrar más información en el siguiente enlace: <http://code.google.com/p/openbiz-cubi/>

## 5. JUEGOS DE PRUEBAS

El juego de las pruebas tiene como objetivo recoger las principales pruebas a las cuales ha sido sometida la plataforma. En estas pruebas s'intenta recoger la casuística más amplia posible. Para hacer las pruebas de la aplicación he utilizado un portátil con las siguientes características:



Figura 48 – Características sistema de prueba

### 5.1 Prueba de creación de solicitud de PD

Vamos a crear una nueva solicitud de PD. Para la gestión de PDs debemos ir inicialmente, una vez ya estamos identificados y dentro de la aplicación, a la pestaña "Solicitudes de PDs" en el menú de PDs, ubicado en la parte superior izquierda de la pantalla.



Figura 49 – Prueba creación de solicitud de PD

En esta pantalla aparece el listado con todas las solicitudes de PDs registradas en el sistema. En la parte superior aparece un filtro sobre el cual podremos hacer clic y se desplegará un buscador.



Figura 50 – Prueba creación de solicitud de PD

En la parte superior derecha del listado de solicitudes de PDs aparecen varias opciones como: "Modificar Solicitud de PD" y "Nueva Solicitud de PD".



PDs > Solicitudes de PDs

[Inicio](#) [Informes](#) [Solicitudes de PDs](#) [Solicitudes PD Genéricas](#) [Gestión de PDs](#) [PDs Aprobadas](#) [Aprobaciones](#) [Ayuda](#)

## Listado de Solicitud de PD

Filtros:

[Modificar Solicitud de PD](#)

[Nueva Solicitud PD](#)

Número Solicitud	Num Pd	Genérica	Ref. Cliente	Cliente	Código Shell	Fecha Solicitud	Estado	Pedido en Curso	Colores Cabeza	Colores Lateral	Fecha Requerida Entrega
6602	9009	No	aswd	KIMOTHOI S.A.	30 x 44	05-01-2013	Anulada	No	1	0	06-02-2013

Figura 51 – Prueba creación de solicitud de PD

El botón "Nueva Solicitud de PD" nos permitirá acceder a la pantalla Nueva Solicitud de PD.

Si hacemos clic encima de un Número Solicitud, vamos a la vista Detalle de Solicitud de PD.

En esta pantalla se podremos registrar una nueva solicitud de PD. Para ello deberá llenar cada uno de los datos que se piden y que se describen a continuación:

- "Núm. PD": El número de PD puede ingresar uno manualmente o dejarlo vacío para que el propio sistema le asigne uno automáticamente.
- "Código Cliente": se registra el "id" del cliente que se seleccione en el campo "Nombre Cliente", este primer campo aparece inhabilitado por defecto.
- "Nombre Cliente": debemos seleccionar un cliente haciendo clic en el cuadro que aparece al lado del campo. Este listado corresponde a las entradas definidas en la pestaña de "Clientes".
- "Referencia del Cliente": este campo es opcional, en caso de que desee llenarlo será con la referencia que tenga del cliente
- "Shell": debemos seleccionar el Shell en la lista desplegable.
- "Líquido que contenemos": debemos seleccionar Líquido de la lista desplegable.
- "Inserto": debemos seleccionar el Inserto en la lista desplegable.
- "Fecha Solicitud": debemos elegir la fecha de solicitud en el calendario que se activa cuando se hace clic encima del icono que aparece al lado del campo.
- "Fecha requerida Entrega": debemos elegir la fecha que se requiere entregar el pedido en el calendario que se activa cuando se hace clic encima del icono que aparece al lado del campo.
- "Barniz interior": barniz interior del PD.
- "Número de Colores": cantidad de colores deseados para la decoración de la cabeza.
- "Colores": colores deseados para la decoración de la cabeza.
- "Número de Colores": cantidad de colores deseados para la decoración lateral.
- "Colores": colores deseados para la decoración lateral.
- "Decoración interior": este campo es opcional, en caso que lo rellenemos, debemos poner cómo queremos la decoración interior.
- "Pedido en Curso": debemos seleccionar SI o NO en la lista desplegable.
- "Solicitud Muestra Física": debemos seleccionar SI o NO en la lista desplegable.
- "Solicitud PDF": debemos seleccionar SI o NO en la lista desplegable.
- "Solicitud de Plancha": debemos seleccionar SI o NO en la lista desplegable.
- "Comercial": debemos seleccionar SI o NO en la lista desplegable.
- "Nombre del proyecto": debemos seleccionar SI o NO en la lista desplegable.

- "Observaciones": observaciones de la solicitud del PD.
- "Sustituye a": En caso que este pedido sea para sustituir una PD existente se debe colocar el número de la PD a la cual sustituye.

El botón que aparece en al lado de "Nombre Cliente", es para seleccionar un cliente. Cuando se hace clic en él, saldrá una ventana emergente dónde lo podemos seleccionar.

Figura 52 – Prueba creación de solicitud de PD

En la siguiente imagen, vemos la ventana emergente debemos seleccionar un cliente, para ello debemos hacer clic encima del cliente deseado (poniéndose de color azul la fila seleccionada). Cuando se haga la selección, debemos hacer clic en el botón "Selección". Una vez que se realice esta acción, volveremos a la pantalla anterior y se habrán llenado los campos: "Clave Cliente" y "Nombre Cliente".

Figura 53 – Prueba creación de solicitud de PD

Una vez que llene todos los campos de la **Nueva solicitud** debemos hacer clic en el botón "Guardar" que se encuentran en la parte superior izquierda de la pantalla.

Figura 54 – Prueba creación de solicitud de PD

El botón "Cancelar" es para cancelar la acción, por tanto no se registrará ningún dato en el sistema.

Completamos el proceso de creación de la solicitud con **éxito**. Podemos ver en el listado de solicitudes nuestra recién creada solicitud (6603).

### Listado de Solicitud de PD

Número Solicitud	Num Pd	Genérica	Ref. Cliente	Cliente	Código Shell	Fecha Solicitud	Estado	Pedido en Curso
6603	9010	No		PERNOD RICARD ESPAÑA, S.A.	31,5 x 24	06-01-2013	Pendiente de Desarrollo	No
6602	9009	No	aswd	KIMOTHOI S.A.	30 x 44	05-01-2013	Anulada	No

Figura 55 – Prueba creación de solicitud de PD

## 5.2 Prueba para modificar una PD

Vamos a modificar una PD, en este caso la PD 1000, modificando el número de pasadas a 4. Si queremos modificar una PD debemos ir al menú "Gestión de PDs", seleccionarla en la lista o buscar con la ayuda del filtro y hacer clic en el botón "Modificar PD".

Utilidades Guala Closures - Gestión de PDs

PDs > **Gestión de PDs**

Inicio | Informes | Solicitudes de PDs | Solicitudes PD Genéricas | **Gestión de PDs**

### Lista de PDs

Filtros:

**Modificar PD** | Nueva PD Genérica | Cambio Masivo de Aditivos

Num PD	Genérica	Ref. Cliente	Cliente	Descripción	Shell
0	Sí	Plata	*hellas*	Aluminio	Unknown
1	Sí				

Figura 56 – Prueba modificación PD

En esta pantalla aparecerán cargados los datos de la PD seleccionada para que sean modificados. En este caso nos desplazamos hacia la parte inferior de la página y cuando encontramos el campo de "pasadas", hacemos clic en modificar.

**PASADAS**

**Modificar**

Pasadas Barniz	Pasadas Litografía	Pasadas Offset	Pasadas Serigrafía	Pasadas HotStamping	Pasadas Tampografía
3	1	0	0	0	0

Figura 57 – Prueba modificación PD

Modificamos el número de pasadas de barniz a “4” y hacemos clic en el botón de “Guardar”.

**Modifica**

Código	6189
Pasadas Barniz	4
Pasadas Offset	0
Pasadas HotStamping	0
Pasadas Esmaltado	0

Figura 58 – Prueba modificación PD

Podemos verificar que los datos han sido modificados correctamente.

**PASADAS**

Pasadas Barniz	Pasadas Litografía	Pasadas Offset	Pasadas Serigrafía	Pasadas HotStamping	Pasadas Tampografía
4	1	0	0	0	0

Figura 59 – Prueba modificación PD

## **6. CONCLUSIONES**

En este TFC se ha diseñado e implementado una aplicación que gestiona las solicitudes de las pruebas de decoración (PD) y que contiene los diseños de decoración. Una aplicación que nos permita la consulta, la modificación y la organización de los datos, controlando también el estado de desarrollo de las PDs para su seguimiento.

Si bien es cierto que algunas y concretas funcionalidades todavía están en fase de desarrollo y no están completas, como por ejemplo las funcionalidades del menú de “Aprobaciones” o el menú de “Ayuda” de la aplicación, que debería contener detallada documentación acerca del funcionamiento de la misma. También sería necesaria la creación de un módulo para la gestión de los usuarios.

Podemos concluir que se han cumplido los objetivos del proyecto y puesto en marcha las principales funcionalidades de gestión de solicitudes y gestión de PDs, dejando por delante una sólida base para terminar algunos detalles de la aplicación en un futuro.

Como conclusión, puedo decir que a nivel personal este trabajo ha sido duro. El tiempo pasa volando y he notado las prisas de última hora. Confieso pero que también ha sido enriquecedor, ya que gracias a él, he podido participar en todas las fases de un proyecto informático y he tenido la oportunidad de poner en práctica y consolidar los conocimientos adquiridos a diferentes asignaturas desarrolladas en el transcurso de mis estudios. También creo que gracias al TFC he ampliado mis conocimientos en el mundo del desarrollo web con software libre, mundo que encuentro de lo más interesante y provechoso para el mundo empresarial, dada la situación actual y la imperiosa necesidad de reducir costes en todos los ámbitos.

## 7. GLOSARIO

- **Litografía:** La litografía es un procedimiento de impresión. Consiste en la adhesión de las tintas grasas y resinosas sobre las hojas del material litografiado.
- **Offset:** Es un método de reproducción de documentos e imágenes sobre papel o materiales similares, que consiste en aplicar una tinta, generalmente oleosa, sobre una plancha metálica, compuesta generalmente de una aleación de aluminio. Constituye un proceso similar al de la litografía.
- **PD Genéricas:** Prueba de decoración que por su decoración básica puede ser utilizada por varios clientes (tapa anónima).
- **Shell:** Término que se refiere al modelo en cuanto a dimensiones de una tapa, diámetro por altura. p.e. 28x15 o 30x60.
- **Inserto:** Parte de plástico que se introduce dentro de la tapa para evitar el contacto directo entre el producto del envase y la tapa de cierre de aluminio.
- **Merma:** Cantidad de aluminio desechado al cortar la hoja de aluminio para crear la forma final de la tapa (cilíndrica).
- **Wet&wet:** Sistema por el cual se pueden aplicar dos capas de pintura a la vez (ej. dos colores), sin realizar dos pasadas por litografía.

## 8. BIBLIOGRAFIA

- Asociación española de documentación e información  
<http://www.sedic.es>
- Wikipedia: La enciclopedia de contenido libre  
<http://es.wikipedia.org>
- Norfi: Bloc de diseño web  
<http://norfipc.com/internet/instalar-servidor-apache.html>
- Portal de Openbiz  
<http://www.openbiz.me/>
- Maestros del web: Portal de desarrollo web con software libre.  
<http://www.maestrosdelweb.com/>
- Portal de Openbiz-Cubi  
<http://code.google.com/p/openbiz-cubi/>
- Desarrolloweb.com: Portal de ayuda para el diseño de páginas web.  
<http://www.desarrolloweb.com>
- Phpopenbiz.org: Sitio de documentación de Openbiz.  
<http://www.phpopenbiz.org/document/>
- HTML.net: Tutoriales sobre HTML y CSS  
<http://es.html.net>
- Cristalab.com: Página de tutoriales diseño web.  
<http://www.cristalab.com/tutoriales/>