SISTEMA DE INFORMACIÓN CON INTERFAZ WEB PARA UNA GALERÍA DE ARTE

ELKIN ANTONIO CAMARGO GIL
FELIX MIGUEL ROYERO DURAN

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE CIENCIAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2010

SISTEMA DE INFORMACIÓN CON INTERFAZ WEB PARA UNA GALERÍA DE ARTE

ELKIN ANTONIO CAMARGO GIL
FELIX MIGUEL ROYERO DURAN

Proyecto para optar el título de:
Ingenieros de Sistemas

Director:
Doctor. JAIME OCTAVIO ALBARRACIN FERREIRA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE CIENCIAS FÍSICO MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2010

AGRADECIMIENTOS

Queremos dar nuestros más sinceros agradecimientos a:

En primera instancia a Dios nuestro señor, quien ha sido nuestro guía espiritual y compañero primordial en todos los momentos de nuestra vida; permitiéndonos cumplir uno de nuestros sueños mas anhelados, el titulo de Ingeniería de Sistemas en la mejor universidad del país, la Universidad Industrial de Santander.

A todos quienes fueron protagonistas en uno de los logros más grandes de nuestras vidas; nuestros padres, amigos y demás familiares, quienes nos han apoyado de todas las formas posibles; al doctor JAIME OCTAVIO ALBARRACIN FERREIRA nuestro director de proyecto, por ayudarnos a sortear los diversos obstáculos que se nos presentaron durante el desarrollo del proyecto.

DEDICATORIA

Este logro tan importante en vida, se lo quiero dedicar a todos quienes estuvieron a mi lado, apoyándome y brindándome todas las fuerzas necesarias para sortear los diferentes obstáculos que se presentaban, pero por sobre todo a mi gran amor de toda la vida Dayana León.

A mis padres, Evangelina Gil Rueda y José Presentación Camargo Santos, y demás familiares por brindarme todo su apoyo.

A mis tíos Alba Patricia Gil Rueda y Heriberto Camacho, y mi profesora de secundaria Cecilia Jaimes, un apoyo incondicional y moral tras la muerte de mi madre

Elkin

DEDICATORIA

A DIOS por darme todo lo más importante que he necesitado en mi vida y ayudarme a salir adelante.

A mi madre por darme todo su amor y por ser mi alcahueta y salvadora en los momentos que la he necesitado.

A mi padre por darme esa guía y educación llena de temple y sabiduría.

A Shirley por ser la mujer que cambio mi mundo, por haberme dado todo y más de su vida, por ser mi amiga y confidente, por quererme tanto, y por darme el mejor regalo de mi vida, mi hija Laura Valentina a quien también le dedico este éxito y todos los que han de venir.

A mi primo Ismael por su apoyo incondicional.

A las familias Galvis Muñoz y Londoño Bernal por brindarme su apoyo y hacerme sentir como un integrante más de sus hogares.

A todos mis amigos en especial a mis compadres Christian, Jaime, Arturo y Piji, por brindarme su amistad incondicional.

Y a todos los que han estado siempre conmigo y me han permitido lograr muchas cosas.

Felix R.

GLOSARIO

WEB: Acrónimo de World Wide Web. Es un sistema mundial de documentos y/o medios enlazados mediante hiperenlaces accesibles a través de internet por medio de navegadores.

BROWSER / NAVEGADOR: El Browser o Navegador es una aplicación cliente que se ejecuta para navegar en internet. Existen diferentes opciones, sin embargo, el mercado está dividido principalmente entre Microsoft Internet Explorer, Mozilla Firefox, Zafari (de Apple), y uno más reciente: Google Chrome.

HTTP: Acrónimo de Hyper text Transfer Protocol. Método mediante el cual son transferidos recursos informáticos a través de la web.

APACHE: El programa servidor web más difundido a través de internet, de código abierto, implementado y realizado de forma colaborativa, con prestaciones, características y funcionalidades equivalentes a las de cualquier servidor comercial.

PHP: Acrónimo de Personal Hypertext Processor. Lenguaje de scripts del lado del servidor, creado en 1994; inicialmente concebido para el servidor apache. Ha tenido gran aceptación entre los desarrolladores debido a su potencia y sencillez. PHP permite incluir piezas de código dentro de una página web, y realizar determinadas acciones de forma fácil y eficaz.

MYSQL: Sistema manejador de base de datos de libre distribución y código

abierto, creado inicialmente para entornos web, pero ha tenido una gran aceptación en otros ámbitos debido a su portabilidad, velocidad y facilidad de uso.

INTERFAZ: Es el conjunto de elementos y acciones que hacen de puente de comunicación entre dos sistemas. Debido a que usualmente los sistemas se comunican en lenguajes distintos, la interfaz debe traducir lo que cada una de las partes dice, para hacerlo comprensible a la otra.

APPSERV: Paquete de libre distribución para Windows y Linux que contiene el servidor web Apache, el lenguaje de scripts PHP, la base de datos MySQL, el administrador de base de datos phpMyAdmin DataBase Manager y PERL.

ARQUITECTURA CLIENTE / SERVIDOR: Consiste básicamente en que un programa – el cliente- realiza peticiones a otro programa – el servidor- que le da la respuesta, el servidor desarrolla tareas en beneficio del cliente.

UML: Acrónimo de Unified Modeling Language. Lenguaje unificado de modelado que permite analizar y diseñar sistemas de una manera muy completa debido a su capacidad de representar la perspectiva de cada una de las personas involucradas en el mismo, por medio de los diagramas que lo componen.

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN

TÍTULO:
SISTEMA DE INFORMACIÓN CON INTERFAZ WEB PARA UNA GALERÍA DE ARTE*.

AUTORES: CAMARGO GIL, ELKIN ANTONIO – ROYERO DURAN, FELIX MIGUEL**.

PALABRAS CLAVES: Galeria de Arte, Web, Materia Prima, Sistema de Informacion, Consumo, Inventario.

DESCRIPCIÓN

El mundo actual se encuentra en constante cambio y evolución generando a diario la directa necesidad de formar parte activa del continuo progreso, hecho primordial que condujo a llevar el impulso mercadotécnico y de control de consumos y existencias de una galería de arte a un entorno Web con enormes posibilidades de brindar un impulso publicitario directo a escala mundial, además de un sistema de información que le provee al artista la posibilidad de llevar su propio control de existencias en inventario y debido control de consumos por obra en un sistema práctico, sencillo y amigable para usar.

Es así como se logra de esta manera solucionar los inconvenientes de promoción de las obras, el conocimiento real de materia prima e insumos en inventario, el costo real en la elaboración de las obras, y la reducción de gastos innecesarios que conllevaban a un sobrecosto en los productos, así como la reducción de ganancias y precios poco económicos de las obras en el mercado, producto de la enorme carencia de control del artista en el cuidado de sus materiales.

En conclusión el Sistema de Informacion ofrece una solución práctica a los inconvenientes presentados en el comercio, inventario y control de las obras, así como de los insumos necesarios para su elaboración y costeo real de las mismas.

* Trabajo de Grado
** Facultad de Ingenierías Físico-Mecánicas, Escuela de Ingeniería de Sistemas e Informática.
Director: Albarracin Ferreira, Jaime Octavio

ABSTRACT

TITLE:
INFORMATION SYSTEM WITH INTERFACE WEB FOR AN ART GALLERY*.

AUTHOR: CAMARGO GIL, ELKIN ANTONIO – ROYERO DURAN, FELIX MIGUEL**.

KEY WORDS: Gallery of Art, Web, Raw material, Information system, Consumption, Inventory

DESCRIPTION

The present world is in constant change on a daily basis and evolution generating the direct necessity to comprise active of the continuous progress, fundamental fact that it lead to take the mercadotécnico impulse and of control of consumptions and existence of a gallery from art to surroundings Web with enormous possibilities of offering a direct advertising impulse on world-wide scale, besides an information system that provides to the artist the possibility to him of taking to its own control of existence in inventory and due control of consumptions by work in a practical system, simple and friendly to use.

It is as well as it is managed this way to solve the disadvantages of promotion of works, the real knowledge of raw material and consumptions in inventory, the real cost in the elaboration of works, and the reduction of unnecessary expenses that entailed to a sobrecost in products, as well as the reduction of gains and little economic prices of works in the market, product of the enormous deficiency of control of the artist in the care of his materials.

In conclusion the project offers a practical solution to the disadvantages presented/displayed in the commerce, inventory and control of works, as well as of the necessary consumptions for its elaboration and I pay for real of the same

* Degree project
**Faculty of Physical-Mechanical engineering, School of Systems Engineering and Computer science.
Director: Albarracin Ferreira, Jaime Octavio

INTRODUCCIÓN

El presente documento corresponde al informe del desarrollo del proyecto de grado “SISTEMA DE INFORMACIÓN CON INTERFAZ WEB PARA UNA GALERÍA DE ARTE”

Para el desarrollo optimo del sistema de información, se realizó un análisis concreto de las diferentes necesidades tanto del artista como de los usuarios, quienes requieren una forma sencilla de visualizar las pinturas y una manera optima de organización para el creador de las obras. De ahí que tras un análisis de requisitos, se llegó al desarrollo de un control objetivo de los insumos necesarios para el proceso de elaboración e inventario de las obras, así como un medio optimo de promoción de la galería y sus productos, evitando el desperdicio innecesario de insumos, gasto adicional de los mismos, y un posible estancamiento de mercancía debido a la falta de rotación por una insuficiente promoción de los artículos.

Para tal desarrollo fue necesaria la lectura de artículos y documentos sobre el desarrollo de aplicaciones en lenguaje PHP, la utilización de un sistema manejador de base de datos MySQL y el uso del servidor Web APACHE. Además del estudio, tanto de metodologías de análisis y diseño, como de lenguajes y modelos cómo UML. De ahí que una vez finalizado el proceso de modelado de la aplicación, se procedió a la fase de implementación.

1 PRESENTACIÓN DEL PROYECTO

Para dar una mejor idea del alcance del proyecto, es necesario enunciar la problemática que da inicio a su desarrollo: su justificación, así como los objetivos que guiaron el proceso de análisis diseño e implementación del sistema de información

1.1 TÍTULO

Sistema de Información con interfaz Web para una galería de arte virtual

1.2 OBJETIVOS

1.2.1 Objetivo general

Desarrollar una herramienta software en ambiente Web que permita la exhibición, cotización, encargo, cobro y control de las obras de arte en ejecución y sus insumos.

1.2.2 Objetivos específicos

- Implementar un módulo que permita al administrador del sistema tener un control de insumos y del inventario de las obras.

- ❖ Implementar un módulo que muestre a los visitantes de la vitrina virtual las características de las obras, y permita cotizarlas.

- ❖ Organizar adecuadamente mediante una base de datos el registro y manejo de todo lo relacionado con los módulos mencionados.

- ❖ Diseñar formularios para la captación de los datos acerca de las obras proyectadas y/o encargadas, de las adquisiciones de insumos (compras) y para el descargue de insumos gastados (gastos) y trabajos cumplidos.

1.3 JUSTIFICACIÓN

1.3.1 Descripción del problema

En una galería de arte es posible que no se lleve un control objetivo de los insumos necesarios para el proceso de elaboración e inventario de las obras, así como un medio optimo de promoción de la galería y sus productos, lo que produce un desperdicio innecesario de insumos, gasto adicional de los mismos, un estancamiento de mercancía debido a la falta de rotación por una insuficiente promoción de los artículos, inconformidad de clientes y pérdidas de los mismos, lo cual se pretende solucionar con el sistema de información en ambiente Web.

En una galería que no se cuente con una herramienta adecuada, se le pueden presentar inconvenientes como:

- Para un cliente en muchas ocasiones la posibilidad de observar una obra, conocer aspectos sobre la misma, saber su precio o proponer alguna de

acuerdo a los diversos gustos particulares, se reducía a la ardua tarea de realizar la búsqueda en forma presencial, disipando en muchas ocasiones el interés de los clientes en realizar este tipo de compras, y de las galerías en incrementar sus ventas y listas de clientes.

- Llevar un control reducido de los diferentes aspectos y requerimientos durante el proceso de elaboración, de ahí que su control de costos en cuanto a los implementos de uso, tiempo de elaboración y otros factores no sean tenidos en cuenta, lo que dificulta una estructura lógica en el proceso de contabilidad y asignación de precio de las obras.

- Se reduce el proceso de comercialización logístico y organizado de este tipo de obras, de ahí que muchos clientes potenciales se percaten de su existencia solo cuando transitan ocasionalmente por la ciudad y sus centros comerciales, en la realización de otras compras o el desarrollo de sus diligencias.

1.3.2 Justificación del proyecto

Para una galería de Arte en la que se elaboran artísticamente cuadros y retratos al óleo, de diferentes tamaños y especificaciones. Y en la cual se desea agilizar el manejo de los insumos, y facilitar el control de las obras disponibles, además de la promoción a gran escala de la galería, apunta a la ampliación de sus recursos tecnológicos. Es así como se decide elaborar un Sistema de Información cuya interfaz sea una vitrina virtual, que solucione estos problemas en el proceso de elaboración de las obras, sus insumos y demás factores, ya que de otra forma se retrasaría el proceso de elaboración generando sobrecoseos e inconformidad de la clientela. La galería de arte obtendría enormes beneficios al adquirir esta herramienta que estamos proponiendo.

1.4 IMPACTO

1.4.1 Social

- ✓ Los clientes tendrán la posibilidad de realizar la observación, verificación, propuesta y cotización de las obras sin necesidad de salir de su casa para tal fin. Convirtiéndose así en una herramienta útil, de tal forma que gestione de manera adecuada la información pertinente tanto en el proceso de desarrollo de las obras, como la de interés para los diferentes usuarios.
- ✓ En general, el usuario que tenga acceso al sistema, gozará de un entorno amigable y sencillo que le permitirá su desenvolvimiento óptimo.

1.4.2 Económico

Con la implantación del sistema de información para la galería de arte, se tendrá un control objetivo en el uso de los materiales e insumos. De esta manera se evitará la compra de materia prima innecesaria, además de conocer la real existencia de insumos y materiales, de ahí que se genere un uso apropiado de los mismos, reduciendo así los costos de las obras. Por otro lado ayudará a disminuir los gastos en publicidad, generando además un aumento de la clientela gracias a la Web.

No habrá necesidad de comprar licencias de uso y tampoco licencias de desarrollo, porque se van a usar herramientas de software libre tales como MySQL para las bases de datos y PHP para el código fuente.

1.4.3 Técnico

El resultado del desarrollo de este proyecto se verá reflejado en un aumento del uso de la tecnología por parte de los diversos usuarios del sistema, clientes, visitantes etc, disponiendo así de herramientas que faciliten la promoción y observación realizada por las diferentes personas.

Al terminar éste proyecto, se dispondrá de una herramienta acorde con la tendencia actual del desarrollo de software: parametrizable y orientado a Web.

1.5 VIABILIDAD

La Universidad Industrial de Santander y en especial la Escuela de Ingeniería de Sistemas e Informática, apoyan el desarrollo de herramientas software para la Web, porque éstas representan grandes beneficios a sus usuarios al brindar independencia de tiempo y espacio, y permitirles usar una herramienta de de mayor actualidad.

Actualmente se cuenta con los recursos tecnológicos y de personal necesarios para la ejecución del proyecto. Por tal motivo no habrá necesidad de llevar a cabo ningún tipo de subcontratación.

En la actualidad existen las herramientas como MySQL y PHP que son tecnologías de desarrollo para la creación de sistemas de información aplicados a múltiples campos, permitiendo el desarrollo de software cada vez más robusto y escalable.

2 MARCO TEÓRICO

En éste capítulo se plasman los aspectos relevantes de los conocimientos técnicos necesarios para la realización de éste proyecto.

2.1 APLICACIONES WEB

Una aplicación Web es un sistema informático que los usuarios utilizan accediendo a un servidor navegando a través de internet o de intranet, Una de las razones más relevantes de la popularidad de las aplicaciones Web es su capacidad de actualizarse y mantenerse sin distribuir e instalar software en miles de potenciales clientes.

2.1.1 Usos comunes de las aplicaciones web

Las aplicaciones Web pueden tener diversidad de usos, tanto para los usuarios que acceden a las aplicaciones finales como para los programadores que las desarrollan. Entre estos se tiene:

- Permitir a los usuarios localizar información de manera rápida y sencilla.

- Tomar, guardar y analizar datos suministrados por los visitantes de los sitios

- Actualización de sitios cuyo contenido cambia constantemente

2.1.2 Funcionamiento de una aplicación web

Una aplicación web es un conjunto de páginas estáticas y dinámicas. Una página estática es aquella que no cambia cuando un usuario la solicita: el servidor envía la página al navegador solicitante sin modificarla. Por el contrario el servidor modifica las páginas dinámicas antes de enviarlas al navegador solicitante.

2.1.3 Procesamiento de páginas web estáticas

Un sitio web estático consta de un conjunto de páginas y de archivos HTML relacionados y alojados en un equipo que ejecuta un servidor web, definido como un software que suministra páginas en respuesta a las peticiones de los navegadores. El contenido final de una página estática lo determina el diseñador de la página, y no cambia cuando se solicita la página.

2.1.4 Procesamiento de páginas web dinámicas

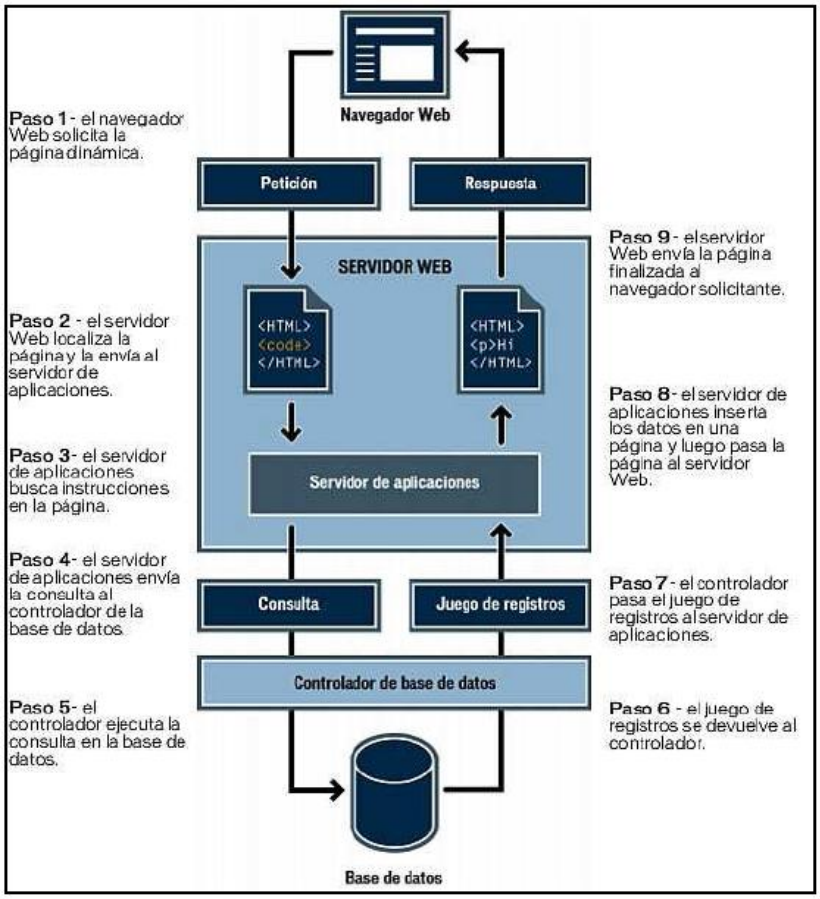
Cuando el servidor web recibe una petición para mostrar una página dinámica, la transfiere a un software encargado de finalizar la página con los datos que se obtienen de la solicitud denominado servidor de aplicaciones, que lee el código de la página, finaliza la página en función de las instrucciones del código y elimina el código de la página, el resultado es una página estática que el servidor de aplicaciones devuelve al servidor web, que a su vez la envía al navegador solicitante. Lo único que el navegador recibe cuando llega la página es el código HTML puro.

2.1.5 Procesamiento de páginas web con acceso a una base de datos

Un servidor de aplicaciones permite trabajar con recursos de lado del servidor como las bases de datos. Por ejemplo, una página dinámica puede indicarle que extraiga datos de una base de datos y los inserte en el código de la página.

Una consulta SQL (Structured Query Language) de una base de datos se escribe del lado del servidor de la página. El servidor de aplicaciones no puede ejecutar directamente una consulta en la base de datos. Para poder obtener los datos de la consulta, se comunica con un controlador que actúa de inmediato. Una vez que el controlador establece la comunicación, la consulta se ejecuta en la base de datos y se extraen un conjunto de registros, el conjunto de registros se devuelve al servidor de aplicaciones, que emplea los datos para completar la página con consulta a BD.

Figura 1: Procesamiento de una página web dinámica



Se puede utilizar prácticamente cualquier base de datos en una aplicación web siempre y cuando se haya instalado el controlador adecuado en el servidor.

2.1.6 Diseño de sitios web

Los sitios web se clasifican en generaciones de acuerdo a las características visuales predominantes en ellos.

Primera generación

En 1993 se realiza el primer diseño de un sitio web, tenía por nombre Mosaic y en menos de un año ya tenía 2’000.000 de visitantes. El explorador tenía capacidad de mostrar imágenes y textos, aunque a la hora de diagramar información era muy limitado. El diseño de sitios web era lineal. Y la tecnología de los navegadores limitaba la capacidad de proveer información gráfica para la comunicación visual. En 1994, se estableció un consorcio llamado w3c para poner metas y normas para el desarrollo futuro, comenzando a diseñar estándares de código de HTML para el diseño Web.

Segunda generación

Esta generación está basada en los conceptos de la primera, pero con algunas diferencias, como íconos que reemplazan las palabras, imágenes para los fondos de página, botones con bordes en relieve, el uso de navegación de arriba hacia abajo con menús para presentar una información jerárquica; muchas de estas características las trajo el avance del diseño web con HTML. El diseño web estaba limitado por la tecnología en constante cambio, por ejemplo, el diseñador necesitaba saber si los monitores eran de 8 o 24 bits. Existía otro problema, internet Explorer y Netscape Navigator tenían sus propias reglas y la visualización de estos sitios era distinta en cada uno, lo que representaba un reto a los diseñadores.

Tercera generación

El contenido dinámico en las aplicaciones web marcó ésta etapa, así como la incorporación del plugin de Macromedia flash en los navegadores; lo cual revolucionó la presentación de los contenidos. La filosofía del diseño web cambió hacia la utilización de contenidos para estrategias de mercadeo y publicidad.

Cuarta generación

Los diseños web de los sitios están basados en la multimedia. Las nuevas versiones de los navegadores de HTML poseen el control que los diseñadores han estado buscando, ahora pueden utilizar elementos con mayor libertad que las versiones anteriores de los navegadores.

2.2 HERRAMIENTAS DE DESARROLLO

2.2.1 HTML

HTML, siglas de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<…>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento, y puede incluir un script (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

2.2.2 JAVASCRIPT

JavaScript es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web, con una sintaxis semejante a la del lenguaje Java y el lenguaje C.

2.2.3 Cliente / Servidor

Esta arquitectura consiste básicamente en que un programa -el cliente- realiza peticiones a otro programa -el servidor- que le da respuesta. Aunque esta idea se puede aplicar a programas que se ejecutan sobre una sola computadora es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras.

La separación entre cliente y servidor es una separación de tipo lógico, donde el servidor no se ejecuta necesariamente sobre una sola máquina ni es necesariamente un sólo programa. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo, los servidores del correo, etc. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma.

Características de un cliente

En la arquitectura C/S el remitente de una solicitud es conocido como cliente. Sus características son:

- Es quien inicia solicitudes o peticiones, tienen por tanto un papel activo en la comunicación (dispositivo maestro o amo).

- Espera y recibe las respuestas del servidor.

- Por lo general, puede conectarse a varios servidores a la vez.

- Normalmente interactúa directamente con los usuarios finales mediante una interfaz gráfica de usuario.

Características de un servidor

En los sistemas C/S el receptor de la solicitud enviada por cliente se conoce como servidor. Sus características son:

- Al iniciarse esperan a que lleguen las solicitudes de los clientes, desempeñan entonces un papel pasivo en la comunicación (dispositivo esclavo).

- Tras la recepción de una solicitud, la procesan y luego envían la respuesta al cliente.

- Por lo general, aceptan conexiones desde un gran número de clientes (en ciertos casos el número máximo de peticiones puede estar limitado).

- No es frecuente que interactúen directamente con los usuarios finales.

2.2.4 PHP

PHP es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting).

PHP es un acrónimo recursivo que significa Hypertext Pre-processor (inicialmente PHP Tools, o, Personal Home Page Tools). Fue creado originalmente por Rasmus Lerdof en 1994; sin embargo la implementación principal de PHP es producida ahora por The PHP Group y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la PHP License, la Free Software Foundation considera esta licencia como software libre.

PHP es un lenguaje que está diseñado especialmente para desarrollo web y puede ser embebido dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno. PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores, aunque el número de sitios en PHP ha declinado desde agosto de 2005. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor web. La más reciente versión principal del PHP fue la versión 5.2.6 de 1 de mayo de 2008.

Ventajas

- ✓ Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL

- ✓ Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.

- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.

- ✓ Permite las técnicas de Programación Orientada a Objetos.

- ✓ No requiere definición de tipos de variables.

- ✓ Tiene manejo de excepciones (desde php5).

- ✓ Tiene funciones de encriptación de un solo sentido predefinidas.

Desventajas

- No posee adecuado manejo de internacionalización, unicode, etc.
- Por sus características favorece la creación de código desordenado y complejo de mantener.
- Si bien PHP no obliga a quien lo usa a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aún estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable.

2.2.5 MYSQL

MySQL es muy utilizado en aplicaciones Web, como Drupal o phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación Web está muy ligada a PHP, que a menudo aparece en combinación con MySQL. MySQL es una base de datos muy rápida en la lectura cuando. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones.

2.3 FUNDAMENTOS DE SEGURIDAD

La seguridad informática consiste en asegurar que los recursos del sistema de información (material informático o programas) de una organización, sean utilizados de la manera que se decidió y, que el acceso a la información allí contenida así como su modificación, sólo sea posible a las personas que se encuentren acreditadas y dentro de los límites de su autorización.

Se puede entender como seguridad, un estado de cualquier sistema (informático o no) que nos indica que ese sistema está libre de peligro, daño o riesgo. Se entiende como peligro o daño todo aquello que pueda afectar su funcionamiento directo o los resultados que se obtienen del mismo. Para la mayoría de los expertos el concepto de seguridad en la informática es utópico porque no existe un sistema 100% seguro. Para que un sistema se pueda definir como seguro debe tener estas cuatro características:

Integridad: La información sólo puede ser modificada por quien está autorizado.

Confidencialidad: La información sólo debe ser legible para los autorizados.

Disponibilidad: Debe estar disponible cuando se necesita.

Irrefutabilidad: (No-Rechazo o No Repudio) Que no se pueda negar la autoría.

Dependiendo de las fuentes de amenaza, la seguridad puede dividirse en seguridad lógica y seguridad física.

En estos momentos la seguridad informática es un tema de dominio obligado por cualquier usuario de Internet, para no permitir que su información sea robada.

2.3.1 Términos relacionados con la seguridad informática

Activo: Recurso del sistema de información o relacionado con éste, necesario para que la organización funcione correctamente y alcance los objetivos propuestos.

Amenaza: Es un evento que pueden desencadenar un incidente en la organización, produciendo daños materiales o pérdidas inmateriales en sus activos.

Impacto: Medir la consecuencia al materializarse una amenaza.

Riesgo: Posibilidad de que se produzca un impacto determinado en un Activo, en un Dominio o en toda la Organización.

Vulnerabilidad: Posibilidad de ocurrencia de la materialización de una amenaza sobre un Activo.

Ataque: Evento, exitoso o no, que atenta sobre el buen funcionamiento del sistema.

Desastre o Contingencia: Interrupción de la capacidad de acceso a información y procesamiento de la misma a través de computadoras necesarias para la operación normal de un negocio.

2.3.2 Análisis de riesgo

El activo más importante que se posee es la información y, por lo tanto, deben existir técnicas que la aseguren, más allá de la seguridad física que se establezca sobre los equipos en los cuales se almacena. Estas técnicas las brinda la seguridad lógica que consiste en la aplicación de barreras y procedimientos que resguardan el acceso a los datos y sólo permiten acceder a ellos a las personas autorizadas para hacerlo.

Los medios para conseguirlo son:

- Restringir el acceso (de personas de la organización y de las que no lo son) a los programas y archivos.

- Asegurar que los operadores puedan trabajar pero que no puedan modificar los programas ni los archivos que no correspondan (sin una supervisión minuciosa).

- Asegurar que se utilicen los datos, archivos y programas correctos en / y /por el procedimiento elegido.

- Asegurar que la información transmitida sea la misma que reciba el destinatario al cual se ha enviado y que no le llegue a otro.

- Asegurar que existan sistemas y pasos de emergencia alternativos de transmisión entre diferentes puntos.

- Organizar a cada uno de los empleados por jerarquía informática, con claves distintas y permisos bien establecidos, en todos y cada uno de los sistemas o aplicaciones empleadas.

- Actualizar constantemente las contraseñas de accesos a los sistemas de cómputo.

2.3.3 Puesta en marcha de una política de seguridad

La seguridad informática debe ser estudiada para que no impida el trabajo de los operadores en lo que les es necesario y que puedan utilizar el sistema informático con toda confianza. Por eso en lo referente a elaborar una política de seguridad, conviene:

- Elaborar reglas y procedimientos para cada servicio de la organización.

- Definir las acciones a emprender y elegir las personas a contactar en caso de detectar una posible intrusión

- Sensibilizar a los operadores con los problemas ligados con la seguridad de los sistemas informáticos.

2.3.4 Las amenazas

Una vez que la programación y el funcionamiento de un dispositivo de almacenamiento (o transmisión) de la información se consideran seguras, todavía deben ser tenidos en cuenta las circunstancias "no informáticas" que pueden afectar a los datos, las cuales son a menudo imprevisibles o inevitables, de modo que la única protección posible es la redundancia (en el caso de los datos) y la descentralización -por ejemplo mediante estructura de redes- (en el caso de las comunicaciones).

Estos fenómenos pueden ser causados por:

- **El usuario:** Causa del mayor problema ligado a la seguridad de un sistema informático (porque no le importa, no se da cuenta o a propósito).

- **Programas maliciosos:** Programas destinados a perjudicar o a hacer un uso ilícito de los recursos del sistema. Es instalado (por inatención o maldad) en el ordenador abriendo una puerta a intrusos o bien modificando los datos. Estos programas pueden ser un virus informático, un gusano informático, un troyano, una bomba lógica o un programa espía o Spyware.

- **Un intruso:** Persona que consigue acceder a los datos o programas de los cuales no tiene acceso permitido (cracker, defacer, script kiddie o Script boy, viruxer, etc.).

- **Un siniestro** (robo, incendio, inundación): una mala manipulación o una malintención derivan a la pérdida del material o de los archivos.

- **El personal interno de Sistemas:** Las pujas de poder que llevan a disociaciones entre los sectores y soluciones incompatibles para la seguridad informática.

2.3.5 Técnicas de aseguramiento del sistema

- Codificar la información: Criptología, Criptografía y Criptociencia, contraseñas difíciles de averiguar a partir de datos personales del individuo.

- Vigilancia de red.

- Tecnologías repelentes o protectoras: cortafuegos, sistema de detección de intrusos - antispyware, antivirus, llaves para protección de software, etc. Mantener los sistemas de información con las actualizaciones que más impacten en la seguridad.

2.3.6 Consideraciones de software

Tener instalado en la máquina únicamente el software necesario reduce riesgos. Así mismo tener controlado el software asegura la calidad de la procedencia del mismo (el software pirata o sin garantías aumenta los riesgos). En todo caso un inventario de software proporciona un método correcto de asegurar la reinstalación en caso de desastre. El software con métodos de instalación rápidos facilita también la reinstalación en caso de contingencia.

Existe software que es conocido por la cantidad de agujeros de seguridad que introduce. Se pueden buscar alternativas que proporcionen iguales funcionalidades pero permitiendo una seguridad extra

2.3.7 Consideraciones de una red

Los puntos de entrada en la red son generalmente el correo, las páginas Web y la entrada de ficheros desde discos, o de ordenadores ajenos, como portátiles.

Mantener al máximo el número de recursos de red sólo en modo lectura, impide que ordenadores infectados propaguen virus. En el mismo sentido se pueden reducir los permisos de los usuarios al mínimo.

2.3.8 Afirmaciones erróneas acerca de la seguridad

Mi sistema no es importante para un cracker. Esta afirmación se basa en la idea de que no introducir contraseñas seguras en una empresa no entraña riesgos pues ¿quién va a querer obtener información mía? Sin embargo, dado que los métodos de contagio se realizan por medio de programas automáticos, desde unas máquinas a otras, estos no distinguen buenos de malos, interesantes de no interesantes, etc. Por tanto abrir sistemas y dejarlos sin claves es facilitar la vida a los virus.

- Como tengo antivirus estoy protegido. En general los programas antivirus no son capaces de detectar todas las posibles formas de contagio existentes, ni las nuevas que pudieran aparecer conforme los ordenadores aumenten las capacidades de comunicación, además los antivirus son vulnerables a desbordamientos de búfer que hacen que la seguridad del sistema operativo se vea más afectada aún.

- Como dispongo de un firewall no me contagio. Esto únicamente proporciona una limitada capacidad de respuesta. Las formas de infectarse en una red

son múltiples. Unas provienen directamente de accesos al sistema (de lo que protege un firewall) y otras de conexiones que se realizan (de las que no me protege). Emplear usuarios con altos privilegios para realizar conexiones puede entrañar riesgos, además los firewalls de aplicación (los más usados) no brindan protección suficiente contra el spoofing.

- Tengo un servidor Web cuyo sistema operativo es un unix actualizado a la fecha. Puede que este protegido contra ataques directamente hacia el núcleo, pero si alguna de las aplicaciones web (PHP, Perl, Cpanel, etc.) está desactualizada, un ataque sobre algún script de dicha aplicación puede permitir que el atacante abra una shell y por ende ejecutar comandos en el unix.

3 PROCESO DE DESARROLLO DE SOFTWARE

3.1 INTRODUCCIÓN

Un sistema informático está compuesto por hardware y software. En cuanto al hardware, su producción se realiza sistemáticamente y la base de conocimiento para el desarrollo de dicha actividad está claramente definida. La fiabilidad del hardware es, en principio, equiparable a la de cualquier otra máquina construida por el hombre.

El primer reconocimiento público de la existencia de problemas en la

producción de software tuvo lugar en la conferencia organizada en 1968 . La **crisis del software** dificultad en escribir programas libres de defectos, fácilmente comprensibles, y que sean verificables. Las causas son, entre otras, la complejidad que supone la tarea de programar, y los cambios a los que se tiene que ver sometido un programa para ser continuamente adaptado a las necesidades de los usuarios. Además, no existen todavía herramientas que permitan estimar de una manera exacta, antes de comenzar el proyecto, cuál es el esfuerzo que se necesitará para desarrollar un programa. Este hecho provoca que la mayoría de las veces no sea posible estimar cuánto tiempo llevará un proyecto, ni cuánto personal será necesario. Cuando se fijan plazos normalmente no se cumplen por este hecho. Del mismo modo, en muchas ocasiones el personal asignado a un proyecto se incrementa con la esperanza de disminuir el plazo de ejecución. Por último, las aplicaciones de hoy en día son programas muy complejos, inabordables por una sola persona. En sus comienzos se valoró como causa también la inmadurez de la ingeniería de software, aunque todavía hoy en día no es posible realizar estimaciones precisas del coste y tiempo que necesitará un proyecto de software. Algunos de los sucesos que se observaban tras el desarrollo del software, eran los siguientes.

- Los proyectos no terminaban en plazo.
- Los proyectos no se ajustaban al presupuesto inicial.
- Baja calidad del software generado.
- Software que no cumplía las especificaciones.
- Código inmantenible que dificultaba la gestión y evolución del proyecto.

3.2 PASOS A SEGUIR DURANTE EL DESARROLLO DE SOFTWARE

Un proceso de desarrollo de software tiene como propósito la producción eficaz y eficiente de un producto software que reúna los requisitos del cliente.

Aunque un proyecto de desarrollo de software es equiparable en muchos aspectos a cualquier otro proyecto de ingeniería, en el desarrollo de software hay una serie de desafíos adicionales, relativos esencialmente a la naturaleza del producto obtenido. A continuación se explican algunas particularidades asociadas al desarrollo de software y que influyen en su proceso de construcción.

Existe una inmensa combinación de factores que impiden una verificación exhaustiva de las todas posibles situaciones de ejecución que se puedan presentar (entradas, valores de variables, datos almacenados, software del sistema, otras aplicaciones que intervienen, el hardware sobre el cual se ejecuta, etc.).

De hay que su desarrollo debe estar enfocado en un proceso evolutivo con miras mejorarlo progresivamente.

Un producto software es intangible y por lo general muy abstracto, esto dificulta la definición del producto y sus requisitos, sobre todo cuando no se tienen precedentes en productos software similares. Esto hace que los requisitos sean difíciles de consolidar tempranamente. Así, los cambios en los requisitos son inevitables, no sólo después de entregado en producto sino también durante el proceso de desarrollo.

Además, de las dos anteriores, siempre puede señalarse la inmadurez de la ingeniería del software como disciplina, justificada por su corta vida comparada con otras disciplinas de la ingeniería. Sin embargo, esto no es más que un inútil consuelo.

El proceso de desarrollo de software no es único. No existe un proceso de software universal que sea efectivo para todos los contextos de proyectos de desarrollo. Debido a esta diversidad, es difícil automatizar todo un proceso de desarrollo de software.

Pressman caracteriza un proceso de desarrollo de software como se muestra en la Figura 2. Los elementos involucrados se describen a continuación:

Figura 2: Elementos del proceso del software

- **Un marco común del proceso**, definiendo un pequeño número de actividades del marco de trabajo que son aplicables a todos los proyectos de software, con independencia del tamaño o complejidad.
- **Un conjunto de tareas**, cada uno es una colección de tareas de ingeniería del software, hitos de proyectos, entregas y productos de trabajo del software, y puntos de garantía de calidad, que permiten que las actividades del marco de trabajo se adapten a las características del proyecto de software y los requisitos del equipo del proyecto.

- **Las actividades de protección**, tales como garantía de calidad del software, gestión de configuración del software y medición, abarcan el modelo del proceso. Las actividades de protección son independientes de cualquier actividad del marco de trabajo y aparecen durante todo el proceso.

3.3 MODELOS DEL PROCESO DE SOFTWARE

Los modelos genéricos no son descripciones definitivas de procesos de software; sin embargo, son abstracciones útiles que pueden ser utilizadas para explicar diferentes enfoques del desarrollo de software.

Modelos que se van a discutir a continuación:

- Codificar y corregir
- Modelo en cascada
- Desarrollo evolutivo
- Desarrollo formal de sistemas
- Desarrollo basado en reutilización
- Desarrollo incremental
- Desarrollo en espiral

3.3.1 Codificar y corregir

Este es el modelo básico utilizado en los inicios del desarrollo de software. Contiene dos pasos:

- Escribir código.
- Corregir problemas en el código.

Se trata primero de implementar algo de código y luego pensar acerca de requisitos, diseño, validación, y mantenimiento.

Este modelo tiene tres problemas principales:

- Después de un número de correcciones, el código puede tener una muy mala estructura, hace que los arreglos sean muy costosos.

- Frecuentemente, aún el software bien diseñado, no se ajusta a las necesidades del usuario, por lo que es rechazado o su reconstrucción es muy cara.

- El código es difícil de reparar por su pobre preparación para probar y modificar.

3.3.2 Modelo en cascada

El primer modelo de desarrollo de software que se publicó, se derivó de otros procesos de ingeniería. Éste toma las actividades fundamentales del proceso de especificación, desarrollo, validación y evolución y las representa como fases separadas del proceso.

Figura3: Modelo en cascada

El modelo en cascada consta de las siguientes fases:

1. **Definición de los requisitos:** Los servicios, restricciones y objetivos son establecidos con los usuarios del sistema. Se busca hacer esta definición en detalle.
2. **Diseño de software**: Se particiona el sistema en sistemas de software o hardware. Se establece la arquitectura total del sistema. Se identifican y describen las abstracciones y relaciones de los componentes del sistema.
3. **Implementación y pruebas unitarias:** Construcción de los módulos y unidades de software. Se realizan pruebas de cada unidad.
4. **Integración y pruebas del sistema:** Se integran todas las unidades. Se prueban en conjunto. Se entrega el conjunto probado al cliente.
5. **Operación y mantenimiento:** Generalmente es la fase más larga. El sistema es puesto en marcha y se realiza la corrección de errores descubiertos. Se realizan mejoras de implementación. Se identifican nuevos requisitos.

En la práctica, este modelo no es lineal, e involucra varias iteraciones e interacciones entre las distintas fases de desarrollo. Algunos problemas que se observan en el modelo de cascada son:

- Las iteraciones son costosas e implican rehacer trabajo debido a la producción y aprobación de documentos.
- Aunque son pocas iteraciones, es usual congelar parte del desarrollo y continuar con las siguientes fases.
- Los problemas se dejan para su posterior resolución, lo que lleva a que estos sean ignorados o corregidos de una forma poco elegante.
- Existe una alta probabilidad de que el software no cumpla con los requisitos del usuario por el largo tiempo de entrega del producto.
- Es inflexible a la hora de evolucionar para incorporar nuevos requisitos. Es difícil responder a cambios en los requisitos.

Este modelo sólo debe usarse si se entienden a plenitud los requisitos. Aún se utiliza como parte de proyectos grandes.

3.3.3 Desarrollo evolutivo

Como el modelo de desarrollo incremental, el modelo de desarrollo evolutivo (algunas veces denominado como prototipado evolutivo) construye una serie de grandes versiones sucesivas de un producto. Sin embargo, mientras que la aproximación incremental presupone que el conjunto completo de requerimientos es conocido al comenzar, el modelo evolutivo asume que los requerimientos no son completamente conocidos al inicio del proyecto. En el modelo evolutivo, los requerimientos son cuidadosamente examinados, y sólo esos que son bien comprendidos son seleccionados para el primer incremento. Los desarrolladores

construyen una implementación parcial del sistema que recibe sólo estos requerimientos. El sistema es entonces desarrollado, los usuarios lo usan, y proveen retroalimentación a los desarrolladores. Basada en esta retroalimentación, la especificación de requerimientos es actualizada, y una segunda versión del producto es desarrollada y desplegada. El proceso se repite indefinidamente. Es de suma importancia entender que el desarrollo evolutivo es 100% compatible con el modelo cascada. El desarrollo evolutivo no demanda una forma específica de observar el desarrollo de algún incremento. Así, el modelo cascada puede ser usado para administrar cada esfuerzo de desarrollo. Obviamente, el desarrollo incremental y evolutivo puede ser combinado también. Todo lo que uno tiene que hacer es construir un subconjunto de requerimientos conocidos (incremental), y comprender al principio que muchos nuevos requerimientos es probable que aparezcan cuando el sistema sea desplegado o desarrollado. El desarrollo de software en forma evolutiva requiere un especial cuidado en la manipulación de documentos, programas, datos de test, etc. desarrollados para distintas versiones del software. Cada paso debe ser registrado, la documentación debe ser recuperada con facilidad, los cambios deben ser efectuados de una manera controlada.

El prototipado de requerimientos es la creación de una implementación parcial de un sistema, para el propósito explícito de aprender sobre los requerimientos del sistema. Un prototipo es construido de una manera rápida tal como sea posible. Esto es dado a los usuarios, clientes o representantes de ellos, posibilitando que ellos experimenten con el prototipo. Estos individuos luego proveen la retroalimentación sobre lo que a ellos les gustó y no les gustó acerca del prototipo proporcionado, quienes capturan en la documentación actual de la especificación de requerimientos la información entregada por los usuarios para el desarrollo del sistema real. El prototipado puede ser usado como parte de la fase de requerimientos (determinar requerimientos) o justo antes de la fase de requerimientos (como predecesor de requerimientos). En otro caso, el prototipado puede servir su papel inmediatamente antes de algún o todo el desarrollo

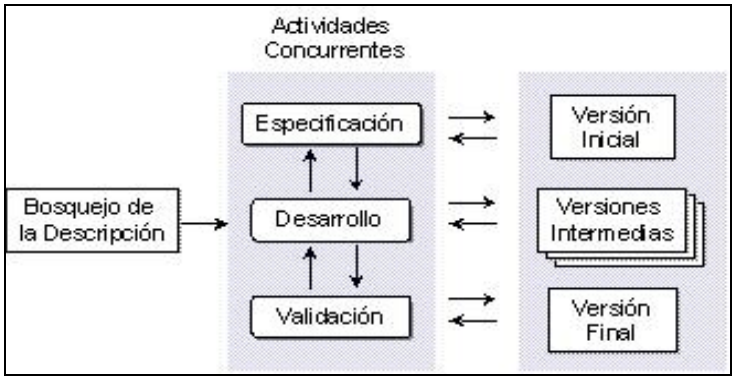
incremental en modelos incremental o evolutivo.

El Prototipado ha sido usado frecuentemente en los 90, porque la especificación de requerimientos para sistemas complejos tiende a ser relativamente dificultoso de cursar. Muchos usuarios y clientes encuentran que es mucho más fácil proveer retroalimentación convenientemente basados en la manipulación, desde un prototipo, en vez de leer una especificación de requerimientos potencialmente ambigua y extensa.

Diferente del modelo evolutivo donde los requerimientos mejor entendidos están incorporados, un prototipo generalmente se construye con los requerimientos entendidos más pobremente.

Una ventaja de este modelo es que se obtiene una rápida realimentación del usuario, ya que las actividades de especificación, desarrollo y pruebas se ejecutan en cada iteración.

Figura 4: Modelo de desarrollo evolutivo



Entre los puntos favorables de este modelo están:

- ✓ La especificación puede desarrollarse de forma creciente.

- ✓ Los usuarios y desarrolladores logran un mejor entendimiento del sistema. Esto se refleja en una mejora de la calidad del software.
- ✓ Es más efectivo que el modelo de cascada, ya que cumple con las necesidades inmediatas del cliente.

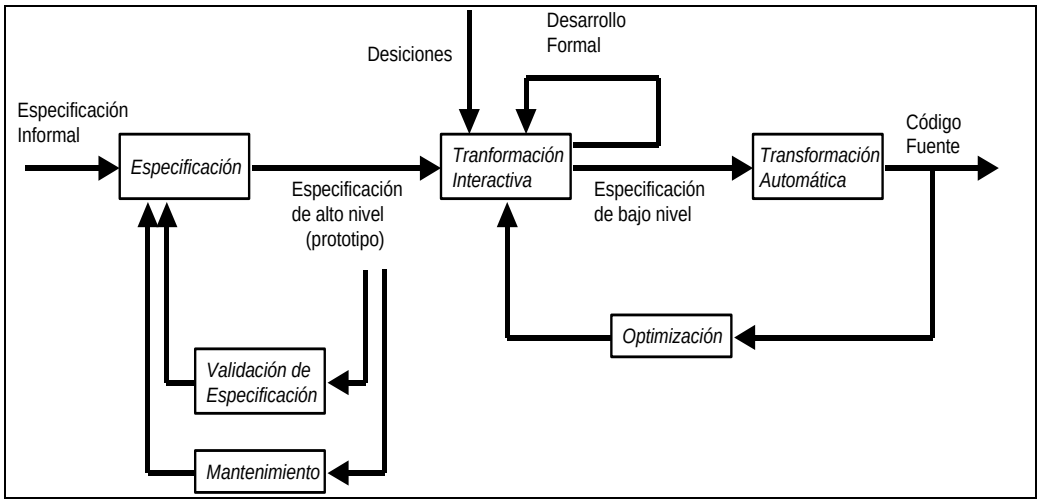
Desde una perspectiva de ingeniería y administración se identifican los siguientes problemas:

- **Proceso no visible:** Los administradores necesitan entregas para medir el progreso. Si el sistema se necesita desarrollar rápido, no es efectivo producir documentos que reflejen cada versión del sistema.
- **Sistemas pobremente estructurados:** Los cambios continuos pueden ser perjudiciales para la estructura del software haciendo costoso el mantenimiento.
- **Se requieren técnicas y herramientas:** Para el rápido desarrollo se necesitan herramientas que pueden ser incompatibles con otras o que poca gente sabe utilizar.

3.3.4 Desarrollo formal de sistemas

Este modelo se basa en transformaciones formales de los requisitos hasta llegar a un programa ejecutable.

Figura 5: Paradigma de programación automática



La Figura 5 ilustra un paradigma ideal de programación automática. Se distinguen dos fases globales: especificación (incluyendo validación) y transformación. Las características principales de este paradigma son: la especificación es formal y ejecutable, constituye el primer prototipo del sistema), la especificación es validada mediante prototipos. Posteriormente, a través de transformaciones formales la especificación se convierte en la implementación del sistema, en el último paso de transformación se obtiene una implementación en un lenguaje de programación determinado. , el mantenimiento se realiza sobre la especificación (no sobre el código fuente), la documentación es generada automáticamente y el mantenimiento es realizado por repetición del proceso (no mediante parches sobre la implementación).

Observaciones sobre el desarrollo formal de sistemas:

- Permite demostrar la corrección del sistema durante el proceso de transformación. Así, las pruebas que verifican la correspondencia con la especificación no son necesarias.
- Es atractivo sobre todo para sistemas donde hay requisitos de seguridad y confiabilidad importantes.

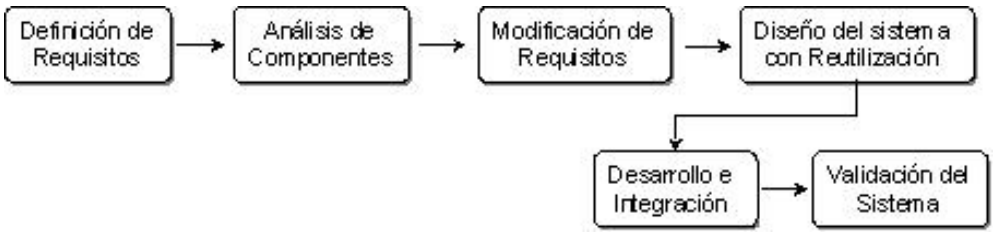
- Requiere desarrolladores especializados y experimentados en este proceso para llevarse a cabo.

3.3.5 Desarrollo basado en reutilización

Como su nombre lo indica, es un modelo fuertemente orientado a la reutilización. Este modelo consta de 4 fases ilustradas en la Figura 6. A continuación se describe cada fase:

1. **Análisis de componentes:** Se determina qué componentes pueden ser utilizados para el sistema en cuestión. Casi siempre hay que hacer ajustes para adecuarlos.
2. **Modificación de requisitos:** Se adaptan (en lo posible) los requisitos para concordar con los componentes de la etapa anterior. Si no se puede realizar modificaciones en los requisitos, hay que seguir buscando componentes más adecuados (fase 1).
3. **Diseño del sistema con reutilización:** Se diseña o reutiliza el marco de trabajo para el sistema. Se debe tener en cuenta los componentes localizados en la fase 2 para diseñar o determinar este marco.
4. **Desarrollo e integración:** El software que no puede comprarse, se desarrolla. Se integran los componentes y subsistemas. La integración es parte del desarrollo en lugar de una actividad separada.

Figura 6: Desarrollo basado en reutilización de componentes



Las ventajas de este modelo son:

- ✓ Disminuye el costo y esfuerzo de desarrollo.
- ✓ Reduce el tiempo de entrega.
- ✓ Disminuye los riesgos durante el desarrollo.

Desventajas de este modelo:

- Los “compromisos” en los requisitos son inevitables, por lo cual puede que el software no cumpla las expectativas del cliente.
- Las actualizaciones de los componentes adquiridos no están en manos de los desarrolladores del sistema.

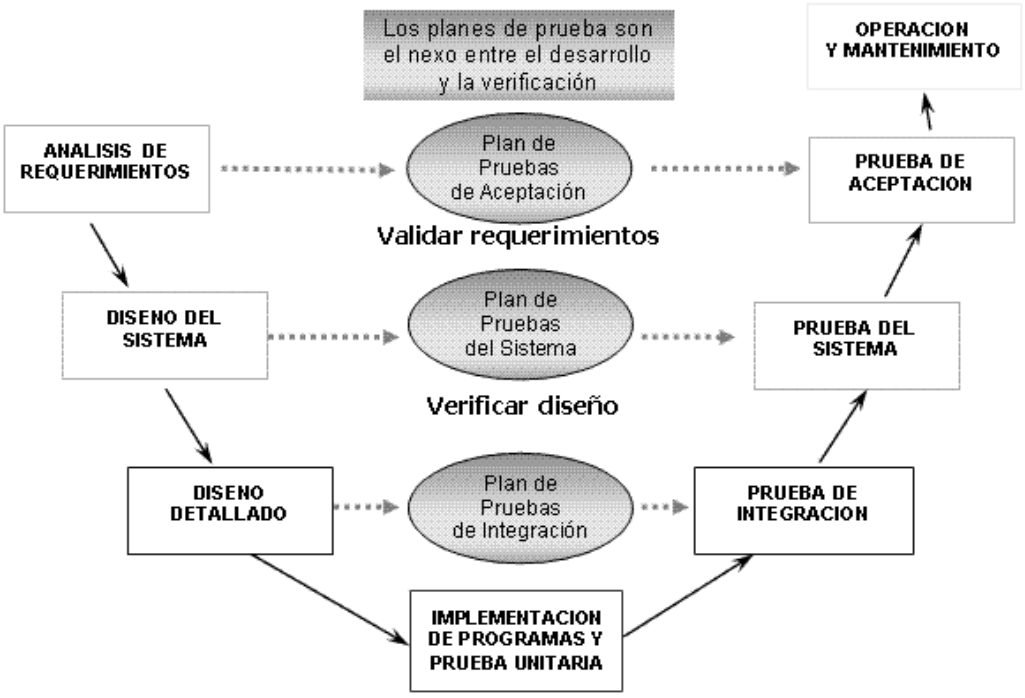
3.3.6 Desarrollo incremental

Mills sugirió el enfoque incremental de desarrollo como una forma de reducir la repetición del trabajo en el proceso de desarrollo y dar oportunidad de retrasar la toma de decisiones en los requisitos hasta adquirir experiencia con el sistema (ver Figura 12). Es una combinación del Modelo de Cascada y Modelo Evolutivo.

Reduce el rehacer trabajo durante el proceso de desarrollo y da oportunidad para retrasar las decisiones hasta tener experiencia en el sistema.

Durante el desarrollo de cada incremento se puede utilizar el modelo de cascada o evolutivo, dependiendo del conocimiento que se tenga sobre los requisitos a implementar. Si se tiene un buen conocimiento, se puede optar por cascada, si es dudoso, evolutivo

Figura 7: Modelo de desarrollo iterativo incremental



Entre las ventajas del modelo incremental se encuentran:

- ✓ Los clientes no esperan hasta el fin del desarrollo para utilizar el sistema. Pueden empezar a usarlo desde el primer incremento.
- ✓ Los clientes pueden aclarar los requisitos que no tengan claros conforme ven las entregas del sistema.
- ✓ Se disminuye el riesgo de fracaso de todo el proyecto, ya que se puede distribuir en cada incremento
- ✓ Las partes más importantes del sistema son entregadas primero, por lo cual

se realizan más pruebas en estos módulos y se disminuye el riesgo de fallos.

Algunas de las desventajas identificadas para este modelo son:

- Cada incremento debe ser pequeño para limitar el riesgo (menos de 20.000 líneas).
- Cada incremento debe aumentar la funcionalidad.
- Es difícil establecer las correspondencias de los requisitos contra los incrementos.
- Es difícil detectar las unidades o servicios genéricos para todo el sistema.

3.3.7 Desarrollo en espiral

El modelo de desarrollo en espiral (ver Figura 13) es actualmente uno de los más conocidos y fue propuesto por Boehm. El ciclo de desarrollo se representa como una espiral, en lugar de una serie de actividades sucesivas con retrospectiva de una actividad a otra.

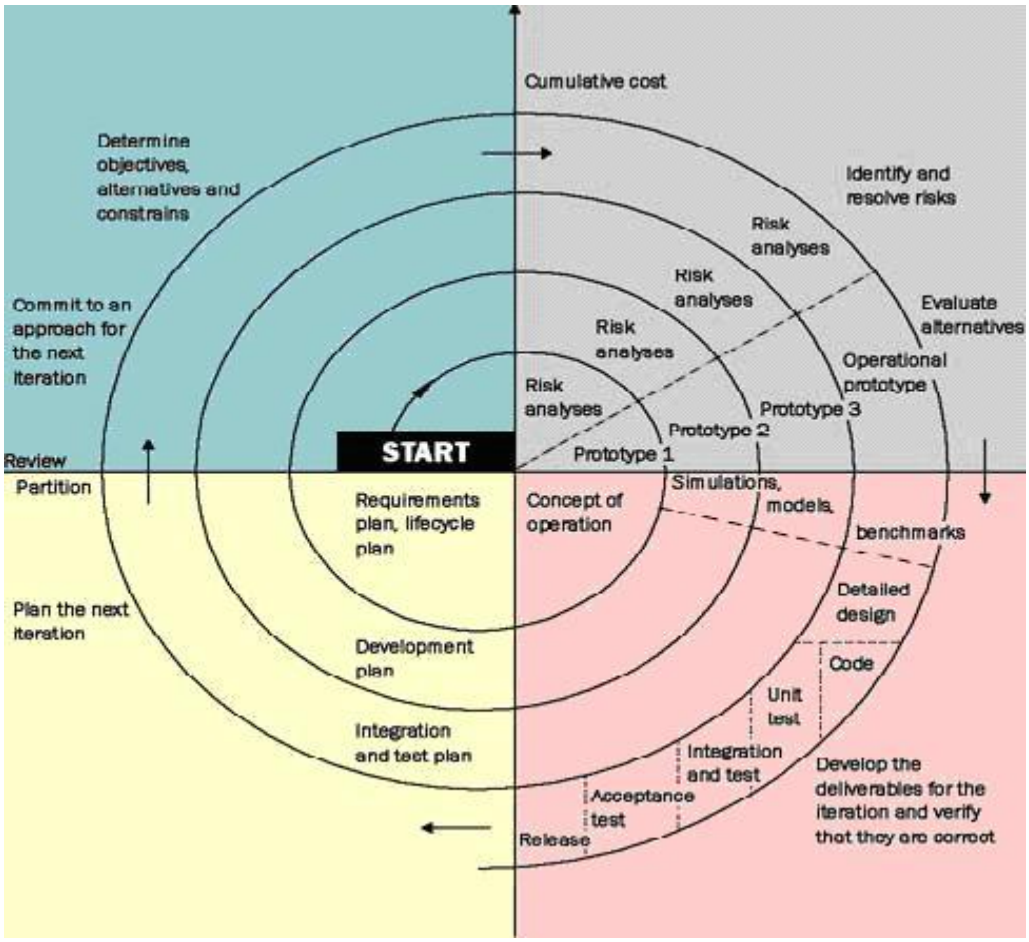
Cada ciclo de desarrollo se divide en cuatro fases:

1. **Definición de objetivos:** Se definen los objetivos. Se definen las restricciones del proceso y del producto. Se realiza un diseño detallado del plan administrativo. Se identifican los riesgos y se elaboran estrategias alternativas dependiendo de estos.
2. **Evaluación y reducción de riesgos:** Se realiza un análisis detallado de cada riesgo identificado. Pueden desarrollarse prototipos para disminuir el riesgo de requisitos dudosos. Se llevan a cabo los pasos para reducir los riesgos.

3. **Desarrollo y validación:** Se escoge el modelo de desarrollo después de la evaluación del riesgo. El modelo que se utilizará (cascada, sistemas formales, evolutivo, etc.) depende del riesgo identificado para esa fase.

4. **Planificación:** Se determina si continuar con otro ciclo. Se planea la siguiente fase del proyecto.

Figura 8: Modelo de desarrollo en Espiral



3.4 ¿CUÁL ES EL MODELO MÁS ADECUADO?

Cada proyecto de software requiere de una forma de particular de abordar el problema. Las propuestas comerciales y académicas actuales promueven procesos iterativos, donde en cada iteración puede utilizarse uno u otro modelo de proceso, considerando un conjunto de criterios (Por ejemplo: grado de definición de requisitos, tamaño del proyecto, riesgos identificados, entre otros).

3.5 METODOLOGÍAS PARA EL DESARROLLO DE SOFTWARE

Un proceso de software detallado y completo suele denominarse “Metodología”. Las metodologías se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo, incremental, etc.). Adicionalmente una metodología debería definir con precisión los artefactos, roles y actividades involucrados, junto con prácticas y técnicas recomendadas, guías de adaptación de la metodología al proyecto, guías para uso de herramientas de apoyo, etc. Habitualmente se utiliza el término “método” para referirse a técnicas, notaciones y guías asociadas, que son aplicables a una (o algunas) actividades del proceso de desarrollo, por ejemplo, suele hablarse de métodos de análisis y/o diseño.

La comparación y/o clasificación de metodologías no es una tarea sencilla debido a la diversidad de propuestas y diferencias en el grado de detalle, información disponible y alcance de cada una de ellas. A grandes rasgos, si se toma como criterio las notaciones utilizadas para especificar artefactos producidos en actividades de análisis y diseño, se pueden clasificar las metodologías en dos grupos: Metodologías Estructuradas y Metodologías Orientadas a Objetos. Por otra parte, considerando su filosofía de desarrollo, aquellas metodologías con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales (o peyorativamente denominada Metodologías Pesadas, o Peso Pesado). Otras

metodologías, denominadas Metodologías Ágiles, están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso. A continuación se revisan brevemente cada una de estas categorías de metodologías.

3.5.1 Metodologías estructuradas

Los métodos estructurados comenzaron a desarrollarse a fines de los 60's con la Programación Estructurada, luego a mediados de los 70's aparecieron técnicas para el Diseño (por ejemplo: el diagrama de Estructura) primero y posteriormente para el Análisis (por ejemplo: Diagramas de Flujo de Datos). Estas metodologías son particularmente apropiadas en proyectos que utilizan para la implementación lenguajes de 3ra y 4ta generación.

3.5.2 Metodologías orientadas a objetos

Su historia va unida a la evolución de los lenguajes de programación orientados a objetos, los más representativos: a fines de los 60's SIMULA, a fines de los 70's Smalltalk-80, la primera versión de C++ por Bjarne Stroustrup en 1981 y actualmente Java o C# de Microsoft. A fines de los 80's comenzaron a consolidarse algunos métodos Orientadas a Objeto.

En 1995 Booch y Rumbaugh proponen el Método Unificado con la ambiciosa idea de conseguir una unificación de sus métodos y notaciones, que posteriormente se reorienta a un objetivo más modesto, para dar lugar al Unified Modeling Language

(UML), la notación OO más popular en la actualidad.

3.5.3 Metodologías tradicionales (no ágiles)

Las metodologías no ágiles son aquellas que están guiadas por una fuerte planificación durante todo el proceso de desarrollo; llamadas también metodologías tradicionales o clásicas, donde se realiza una intensa etapa de análisis y diseño antes de la construcción del sistema.

Todas las propuestas metodológicas antes indicadas pueden considerarse como metodologías tradicionales. Aunque en el caso particular de RUP, por el especial énfasis que presenta en cuanto a su adaptación a las condiciones del proyecto (mediante su configuración previa a aplicarse), realizando una configuración adecuada, podría considerarse Ágil.

3.5.4 Metodologías ágiles

Un proceso es ágil cuando el desarrollo de software es incremental (entregas pequeñas de software, con ciclos rápidos), cooperativo (cliente y desarrolladores trabajan juntos constantemente con una cercana comunicación), sencillo (el método en sí mismo es fácil de aprender y modificar, bien documentado), y adaptable (permite realizar cambios de último momento).

Entre las metodologías ágiles se pueden destacar:

- Extreme Programming.
- Scrum.
- Familia de Metodologías Crystal.
- Feature Driven Development.
- Proceso Unificado Rational
- Dynamic Systems Development Method
- Adaptive Software Development

4 METODOLOGÍA DE DESARROLLO

En éste capítulo se muestra el progreso del proyecto durante cada una de las fases especificadas en la metodología, especificación de requisitos, análisis, diseño e implementación.

El desarrollo del sistema de información para una Galería de Arte, tendrá un impacto positivo para todas aquellas personas que de una u otra forma estarán beneficiadas con este sistema, facilitando así el actual proceso de observación, control y cotización durante el proceso de desarrollo y comercialización de las obras.

4.1 ANÁLISIS

4.1.1 Requisitos específicos

1. El sistema debe ser una aplicación en entorno Web, con el propósito de estar disponible en cualquier computador con acceso a internet.

2. El sistema deberá ser de fácil manipulación tanto para el administrador como para el usuario o cliente.

3. Para entrar a la sección de administración del sistema, el usuario administrador contara con un Login y un Password, los cuales obtendrá con su debido registro en el sistema.

4. El usuario o cliente contara con un Login y un Password, los cuales le permitirán acceder a muchos de los servicios del sistema, estos los adquirirá con el correspondiente registro en el sistema.

5. Realizar validaciones de todas las entradas de datos.

6. Permite al usuario administrador cambiar su Login y contraseña cuando y cuantas veces lo desee.

7. Permitir al usuario o cliente cambiar su Login y contraseña, cuando y cuantas veces lo desee.

8. Hacer uso de iconos y gráficos para ayudar al usuario a llegar más rápido al modulo exacto, del cual pretende beneficiarse.

9. Al tratar de entrar al sistema, si el usuario escribe mal sus datos, deberá mostrarle un mensaje que indique que la entrada ha fallado.

10. El usuario tendrá la posibilidad de visualizar las obras que forman parte de la galería, observar sus características generales, cotizarlas, adquirirlas, saber más de alguna obra en particular, además de la posibilidad de proponer sus propias obras.

11. Si el usuario o cliente desea saber más de alguna obra en particular, lo realizara por medio del link “**Te interesa saber mas de alguna obra**”.

12. En el formulario para saber más de alguna obra en particular, la selección de la obra, se realizara de entre las que actualmente estén cargadas en el sistema, con sus correspondientes características.

13. Para la opción de cotización de obra es necesario que el usuario este debidamente registrado.

14. La opción de cotización de obra contará con la posibilidad de adjuntar una imagen guía, además de realizar una breve descripción de los intereses particulares de la pintura.

15. Si el usuario o cliente decide adquirir alguna obra, es preciso que este registrado.

16. Para el proceso de adquisición de la obra el pago se realizara por medio de consignación y no por medios electrónicos, por ello el sistema de información carecerá de este servicio de pago.

17. En la opción de adquisición o compra de la obra, se desplegará una primera sección con un formulario con la información para efectuar el pago (banco, titular, número de cuenta…), al final de esto aparecerá el link de la segunda sección.

18. La segunda sección del formulario de compra, se visualizara cliqueando el link "**ya efectúo el pago**", de esta forma permitirá la visualización de los datos de pago.

19. Implementar un modulo, que le permita al usuario administrador cargar las imágenes de sus respectivas obras con sus características.

20. El modulo de administración de obras, contará con la posibilidad de modificar la técnica y estilo con la que se desarrollo la obra.

21. La información necesaria en el formulario para cargar la imagen de la obra, contara con las siguientes especificaciones:
    - Cargar Imagen
    - Nombre
    - Técnica
    - Estilo
    - Dimensión
    - Precio
    - Tiempo de elaboración
    - Descripción

22. Las opciones de Técnica y Estilo en el formulario para cargar la imagen de la obra, corresponderán a las almacenadas en la base de datos por el artista.

23. Establecer un modulo de noticias, en donde el usuario administrador tenga la posibilidad de reportar las novedades de su negocio, así como la facultad de editar y borrar sus reportes de acuerdo al orden de novedades.

24. Efectuar en los formularios implementados, mecanismos de seguridad y validación.

25. Módulo que permita al administrador del sistema, tener un control de compra de insumos y materia prima.

26. La información necesaria en el formulario del modulo para administrar la compra de materiales, contara con las siguientes especificaciones:
    - Agregar Item
    - Consultar Items (la consulta se puede hacer por fecha, proveedor o material)

27. El modulo de compra de insumos y materiales, tendrá la posibilidad de informar al usuario administrador, el total facturado en cada proceso de inserción de información al sistema.

28. El modulo de compra de insumos contara con la posibilidad de consulta por fecha, articulo y proveedor, arrojando resultados de acuerdo a las características de consulta.

29. La información necesaria del proveedor contara con las siguientes especificaciones:
    - Nombre
    - Apellidos
    - Razón Social
    - Nit
    - Dirección
    - Celular
    - Teléfono
    - Email

30. Modulo para el control de consumo de materia prima e insumos por cada obra.

31. El modulo para el control de consumo de materia prima contara con tres opciones
    - Consulta Existencia
    - Registro Consumo
    - Información Consumo por Obra

32. La opción de consulta de existencia, mostrará la meteria prima e insumos que se encuentran actualmente en inventario.
33. La opción de registro de consumo tendrá la posibilidad de registrar el gasto de materia prima e insumos por obra.

34. La información necesaria de la opción de registro de consumo por obra, contara con las siguientes especificaciones:
    - Seleccionar Obra
    - Fecha de Registro
    - Material
    - Cantidad

35. La selección de obra en la opción de registro de consumo, dependerá de la información cargada en la base de datos por el usuario administrador.

36. La información por obra, que se podrá visualizar en la opción de Contabilidad, contara con las siguientes especificaciones.
    - Obra
    - Material gastado
    - Cantidad de material gastado

- Tiempo de elaboración

37. Permitir recordar la contraseña a los usuarios en caso de olvidarla.

4.1.2 Diagramas de casos de uso

Los diagramas de casos de uso del sistema han sido elaborados de acuerdo a los usuarios del sistema, mostrando en cada diagrama las acciones que pueden realizar cada uno de ellos.

Figura 9: Casos de uso del visitante / Cliente

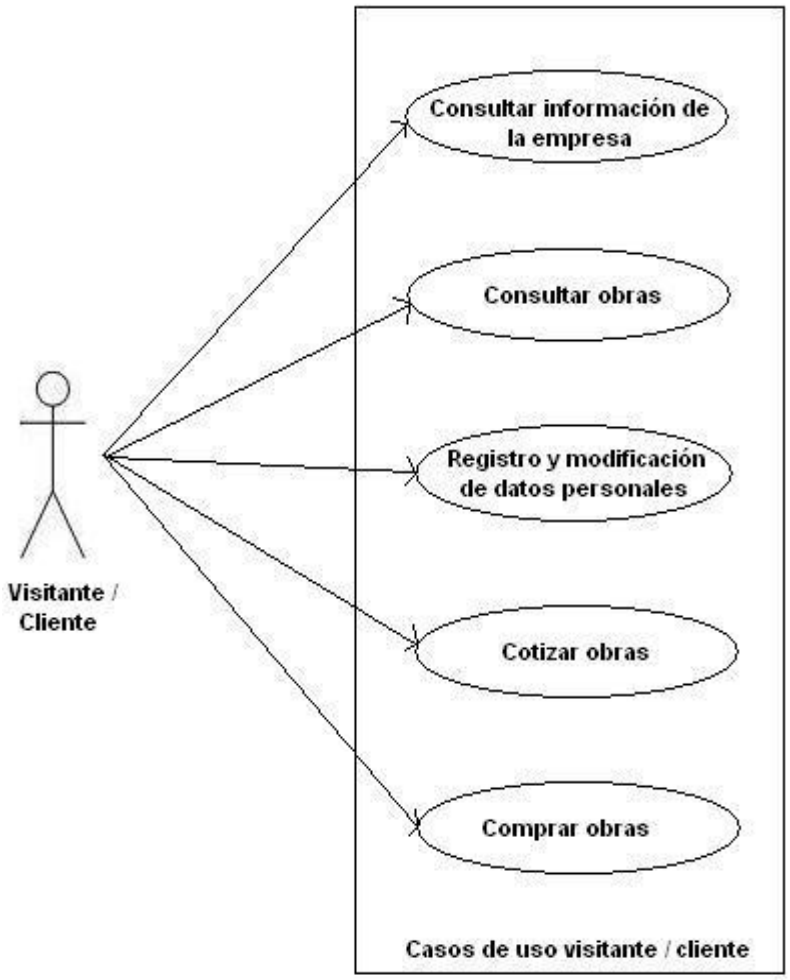
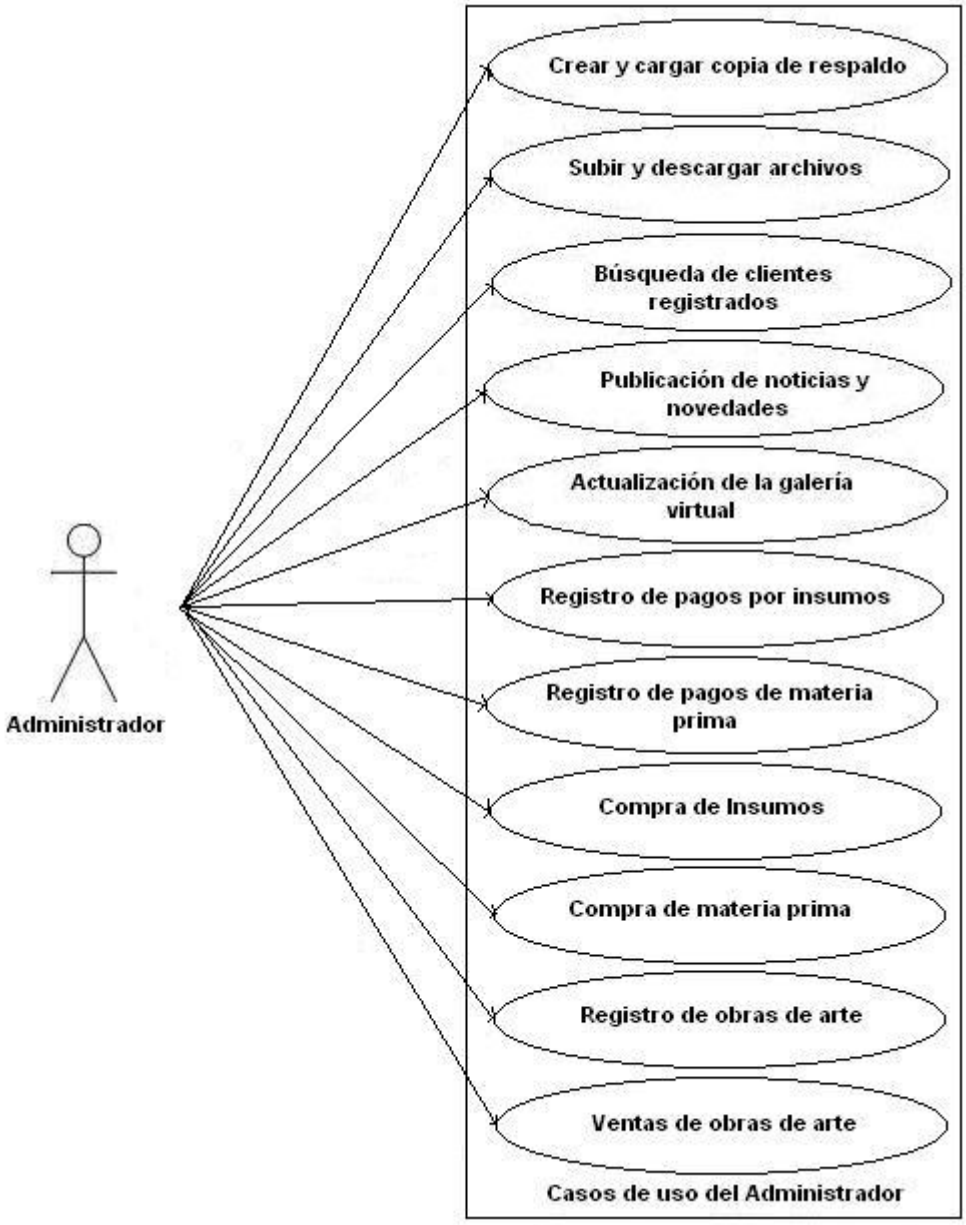


Figura 10: Casos de uso del administrador



4.1.3 Actores del sistema

Los usuarios o actores que tendrán participación en el sistema son, en primer lugar los visitantes o posibles clientes interesados en las obras de arte, seguido por el administrador del sistema.

Tabla 1: Actores del sistema

USUARIO	DESCRIPCIÓN	RESPONSABILIDADES	NECESIDADES
Visitante / cliente	Representa a todos aquellos interesados en las obras de arte ofrecidas por la galería.	Mantener actualizados sus datos, en caso de que se registre, además de responder por las acciones realizadas bajo su login	Observar, cotizar y comprar las obras de su agrado, ofrecidas por la Galería de Arte, además de la actualización de sus datos personales
Administrador del	Es la persona con los privilegios de administración del sistema	Responder por las acciones realizadas bajo su login, Administrar los usuarios, noticias y novedades publicadas, contenidos, módulos	

sistema		de control, contraseñas y archivos	

4.1.4 Relación de los actores del sistema

Tabla 2: Actores del sistema Vs. Casos de uso

ACTORES DEL SISTEMA	CASOS DE USO
Visitante / Cliente	Consultar información de la empresa Consultar obras Registro y modificación de datos personales Cotizar obras Comprar obras
Administrador del	Crear y cargar copia de respaldo Subir y descargar archivos Búsqueda de clientes registrados Envío de correos a clientes registrados Publicación de noticias y novedades

sistema	Actualización de la galería virtual Control de inventario de insumos, materia prima, y obras de arte

4.1.5 Descripción de los casos de uso

Tabla 3: Descripción de los casos de uso

CASOS DE USO	DESCRIPCIÓN
USUARIO CLIENTE	
Consultar información de la empresa	Acción que implica ir a la página principal de la Galería de Arte y dar clic en los link correspondientes.
	Esto acción se realiza en la Galería virtual, pues es allí donde se muestran las diversas obras

Consultar obras	realizadas.
Registro y Modificación de datos personales	Proceso mediante el cual un visitante o cliente, puede registrarse, o en caso de ya haber realizado el registro, actualizar los datos personales que tiene almacenados en la base de datos. Hay que recomendar que la contraseña debe ser cambiada cada determinado periodo, con motivos de seguridad.
Cotizar obras	Este es el proceso por el cual, el cliente propone su propia obra, de acuerdo a sus especificaciones particulares.
Comprar obras	Proceso mediante el cual el cliente realiza la adquisición de la obra elegida.
Administrador del sistema	
Crear y cargar la copia de respaldo	Algo importante: crear y cargar la copia de respaldo. Esto se deberá hacer diariamente, semanalmente y semestralmente, se

	recomienda que la copia de respaldo no quede en el mismo lugar donde se encuentra el sistema.
Subir y descargar archivos	En esta acción el administrador deberá ponerle título y una breve descripción a los archivos que se subirán en el sistema de información.
Búsqueda de clientes registrados	Esto lo hará el administrador con el fin de saber el número de clientes ya registrados en la galería.
Envío de correos a clientes registrados	Esta acción se llevará a cabo con la finalidad de enviar publicidad y mensajería de interés para los clientes.
Publicación de noticias y	Las noticias y novedades son las que se estarán desplazando en el panel de la izquierda, y es de suprema importancia que el

novedades	administrador esté pendiente de colgar noticias, novedades y promociones de interés para los visitantes y clientes.
Actualización de la Galería virtual	Este proceso consistirá en cargar en el sistema de información, las diferentes obras realizadas, y actualizar las diversas novedades de las mismas.
Control de gastos de materia prima e insumos	Acción que implica el registro del gasto regular de materia prima e insumos en el modulo correspondiente, para así llevar un adecuado control.
Control de inventario de materia prima, insumos, y obras de arte	Proceso por el cual se lleva el inventario de la materia prima, insumos, y obras de arte, gracias al registro correspondiente de estos datos, en los módulos establecidos del sistema de información.

Control de compra de materia prima e insumos	Acción que implica, el registro de los datos pertinentes a la materia prima, e insumos adquiridos para la elaboración de las obras de arte.
Control de costos de insumos, materia prima y obras de arte	Algo importante: Llevar un control objetivo de los diversos costos implicados en el proceso de elaboración de las respectivas obras.

4.1.6 Casos de uso detallados del Cliente o Visitante

Tabla 4: Casos de uso detallados del Cliente o Visitante

CASOS DE	PRECONDICIONES	PROCESO	POSCONDICIONES

USO			
Consultar información general de la Galería de Arte	Acceder a la plataforma web del Sistema de Información.	Dar clic en los vínculos de información general Empresa, misión visión; etc. en la parte superior.	Obtener información general acerca de la Galería de Arte.
Consultar obras	Acceder a la plataforma web del Sistema de Información.	En la página de inicio del usuario visitante, se encuentra en el menú derecho Obras, en donde están las pinturas realizadas por el artista.	Contar con la información actualizada de las diferentes obras.
Registro y modificación de datos personales	Acceder a la plataforma web del Sistema de Información.	El cliente busca el vínculo Registrarse en la página de incio del sistema de información, en donde podrá registrarse, y dado el	La información personal actualizada, quedara registrada en la base de datos.

		caso modificar los datos personales que accesó en el debido proceso.	
Cotizar obras	Acceder a la plataforma web del Sistema de Información.	En la plataforma web, del Sistema de Información, el cliente accede a la galería virtual, que se encuentra en el menú izquierdo Obras, allí de acuerdo a sus exigencias, el cliente realizara el proceso de cotización de la obra, accediendo al link Cotiza tu Obra.	El artista recibirá la cotización de la obra particular, propuesta por el cliente interesado.
Comprar obras	Acceder a la plataforma web del Sistema de Información.	En la plataforma web, del Sistema de Información, el cliente accede a la galería virtual, que se encuentra en el	Realizado el proceso de pago, el cliente registra los datos necesarios, el artista recibirá los datos de este

		menú izquierdo Obras, allí puede realizar el proceso de adquisición gracias al link Compre ahora.	proceso, para su correspondiente revisión.

4.1.7 Casos de uso detallados del administrador

Tabla 5: Casos de uso detallados del administrador

CASOS DE USO	PRECONDICIONES	PROCESO	POSCONDICIONES
Crear y cargar copia de respaldo	El administrador debe estar registrado en la base de datos.	El Administrador digita su código y contraseña en la página de login, luego selección el botón de administrador y da clic en entrar, si alguno de los dos campos no coinciden, no se permite el acceso	El Administrador inicia sección en el sistema

		al sistema	
Subir y descargar archivos	El Administrador debe estar en su sección de usuario	El Administrador busca el vínculo modificar datos personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da clic en el botón Actualizar, para guardar los cambios	La información personal actualizada queda guardada en la base de datos

Búsqueda de clientes registrados.	El administrador debe estar en su sección de usuario	El Administrador buscar el link de cambiar las noticias, luego podrá ingresar una nueva noticia si así lo desea, o podrá editar las noticias que ya existen, es decir, podrá cambiar el título de la notica, el enlace y la descripción de la misma	Se registra una nueva noticia en la base de datos, o se actualiza un registro modificado por parte del administrador
Envío de correos a clientes registrados	El Administrador debe estar en su sección de usuario; y para eliminar, al menos debe existir un archivo	El Administrador ingresa una palabra o frase del archivo, a continuación da clic en buscar y en el resultado de la búsqueda podrá seleccionar el archivo que desea descargar, pero si por el contrario desea eliminar el	El administrador ha subido un nuevo archivo al sitio web, o ha borrado un archivo existente

		archivo, sólo tendrá que dar clic en la X que se encuentra en la parte superior derecha, al lado del nombre y la descripción del archivo	
Publicación de noticias y novedades	El Administrador debe estar en su sección de usuario	En el menú del administrador, se encuentra la opción de ingresar un nuevo usuario al sistema, él escoge que tipo de usuario desea ingresar y de clic en el vínculo correspondiente al que escogió, luego se despliega un formulario en el cual el administrador	Se registra en la base de datos el nuevo usuario que el administrador creó, o se actualiza la información de éste que haya sido modificada.

		ingresará los datos del usuario y le podrá el estado de activo o inactivo, la clave del usuario es generada randómicamente y posteriormente es enviada al correo del usuario; luego de esto el usuario procederá a revisar su bandeja de entrada para consultar la contraseña, y luego de que ingrese con ésta contraseña, podrá cambiar la contraseña asignada por una que él desee	
	El Administrador debe estar en su	El Administrador busca el vínculo modificar datos	La información personal actualizada queda

Actualización de la galería virtual	sección de usuario	personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da clic en el botón Actualizar, para guardar los cambios	guardada en la base de datos
Control de gastos de insumos y materia prima.	El Administrador debe estar en su sección de usuario	El Administrador busca el vínculo modificar datos personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da clic en el botón Actualizar, para guardar los cambios	La información personal actualizada queda guardada en la base de datos

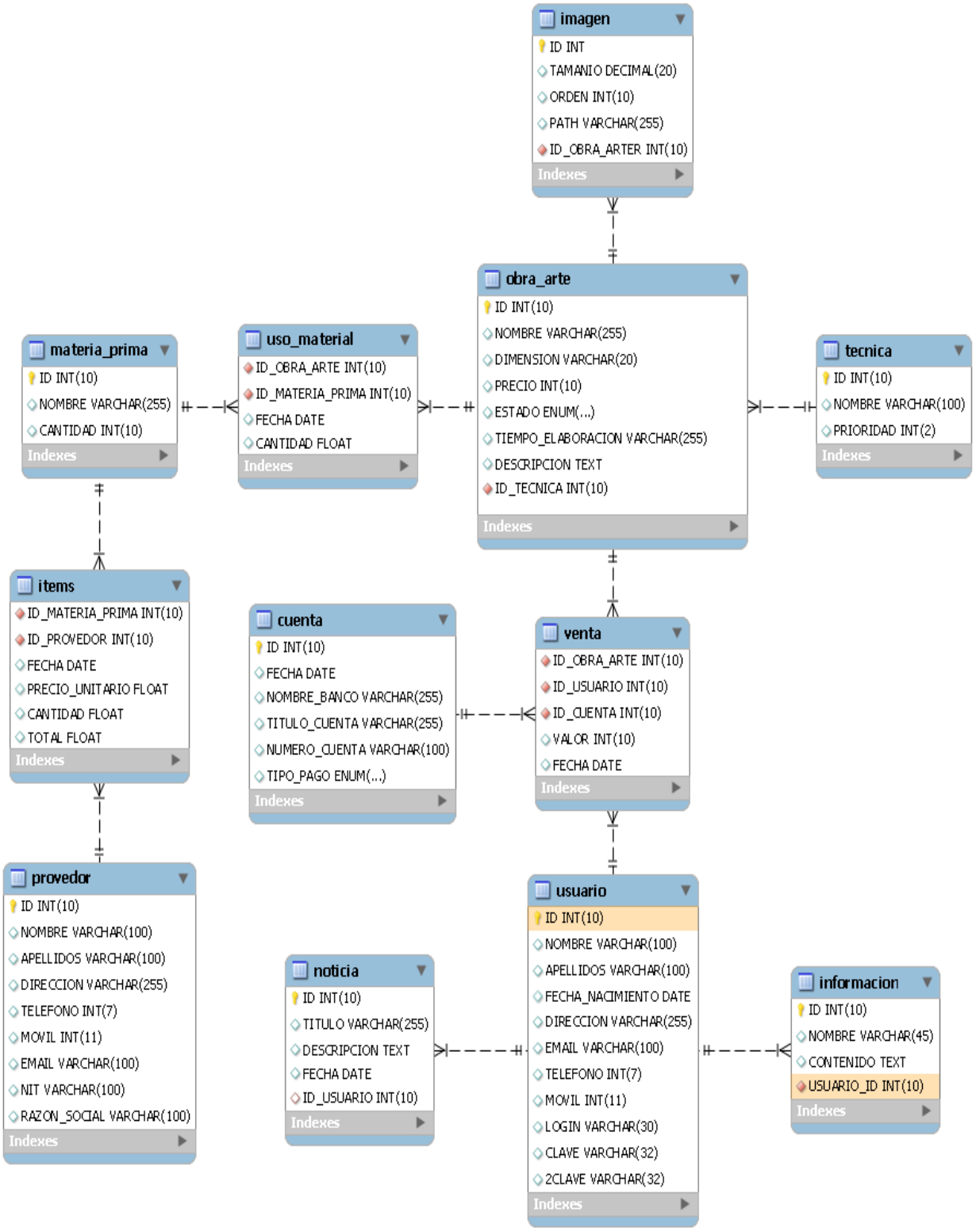
Control de inventario de insumos, materia prima, y obras de arte	El Administrador debe estar en su sección de usuario	El Administrador busca el vínculo modificar datos personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da clic en el botón Actualizar, para guardar los cambios	La información personal actualizada queda guardada en la base de datos
Control de compra de insumos y materia prima	El Administrador debe estar en su sección de usuario	El Administrador busca el vínculo modificar datos personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da	La información personal actualizada queda guardada en la base de datos

		clic en el botón Actualizar, para guardar los cambios	
Control de costos de insumos, materia prima, y obras de arte	El Administrador debe estar en su sección de usuario	El Administrador busca el vínculo modificar datos personales y actualiza en el formulario la información que haya cambiado, o ingresa aquella que no estaba registrada y da clic en el botón Actualizar, para guardar los cambios	La información personal actualizada queda guardada en la base de datos

4.2 DISEÑO

4.2.1 Diagrama entidad relación

Figura 1: Diagrama Entidad-Relación



4.2.2 Definición de tablas del Modelo de datos

A continuación se detallan las entidades que conforman la base del sistema de información

Tabla de obra_arte: Ésta tabla almacena la información pertinente a la Obra de arte realizada; nombre, dimensión, precio, incluyendo datos como el estado de elaboración, y su tiempo de elaboración.

Figura 22: Tabla de obra_de_arte

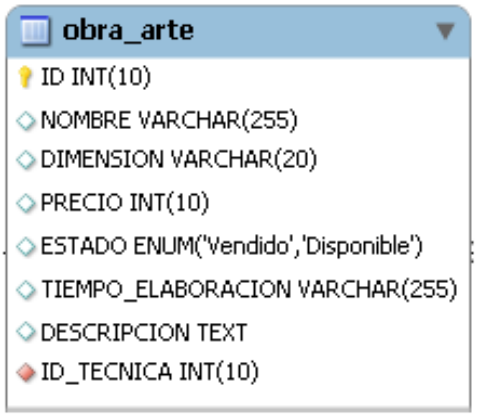


Tabla tecnica: Ésta tabla almacena las diferentes técnicas usadas por el artista en el desarrollo de las obras, ya que existe gran variedad de estas en su desarrollo, de esta forma el artista tendrá la posibilidad de almacenarlas en la base de datos, de acuerdo su uso en las pinturas elaboradas.

Figura 3: Tabla de Técnica

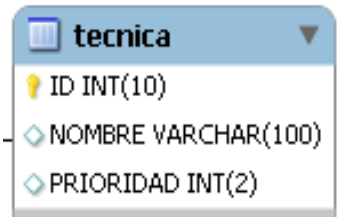


Tabla usuario: Esta tabla es la encargada del almacenamiento y control de los datos de los usuarios registrados, esto incluye, a todos los visitantes o clientes, que se registran en el sistema para poder realizar efectivamente la cotización de alguna obra, o comprar; a su ves almacena los datos del administrador, encargado del control del sistema de información.

Figura 14: Tabla de usuario

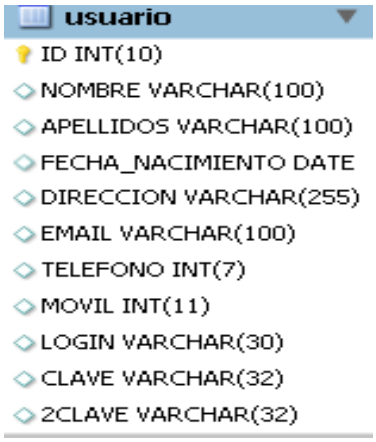


Tabla Venta: Esta tabla se genero con la relación establecida ente obra_arte y usuario, la cual permitirá tener un control objetivo de las ventas realizadas en la galería, para cuyo caso se tendrá información tanto de la obra comercializada, como del cliente que la adquirió, de hay que este ultimo debe de estar registrado en el sistema de información.

Figura 15: Tabla de venta

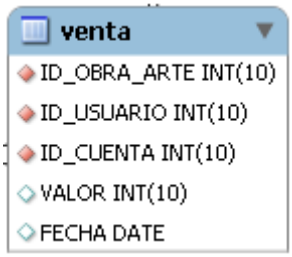


Tabla noticia: Esta tabla consta de tres campos; TITULO, DESCRIPCION, FECHA, permitiendo al administrador, la posibilidad de cargar sus noticias, ofertas y novedades en fechas especificas, editarlas y borrarlas, de esta forma podrá dar un impulso mercadotécnico mas efectivo a su empresa y sus productos.

Figura 16: Tabla de noticias

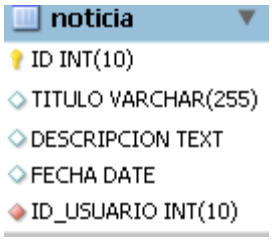


Tabla materia_prima: En esta tabla se almacena información pertinente a la materia prima adquirida, cuya finalidad primordial es la de llevar un control de inventario, de la mercancía disponible para la elaboración de productos, de esta forma se tendrá un conocimiento de existencias.

Figura 17: Tabla de materia_prima

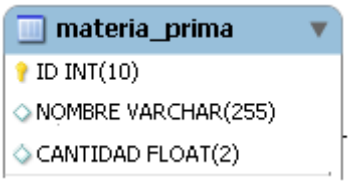


Tabla uso_material: La tabla se genero de la relación entre materia_prima y obra_arte; consta de los campos FECHA, CANTIDAD, permite realizar un control objetivo, de los gastos de materia prima e insumos por obra, de esta manera se tendrá información verídica de los consumos incurridos en la elaboración de los productos.

Figura 18: Tabla de uso_material

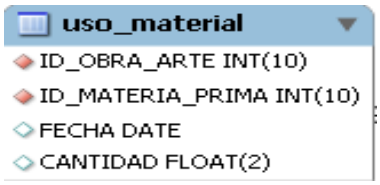


Tabla proveedor: Ésta tabla almacena la información pertinente a los diferentes proveedores, los cuales suministran la materia prima e insumos necesarios para la elaboración de las obras, su objetivo primordial es brindarle al administrador, de acuerdo a la información almacenada, la posibilidad de tomar la mejor decisión al adquirir los suministros necesarios.

Figura 19: Tabla de proveedor

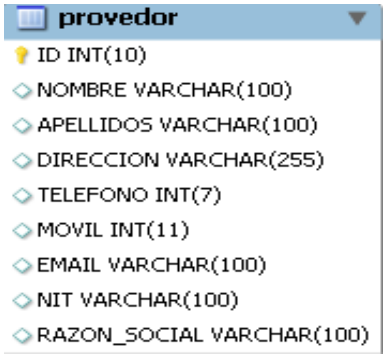
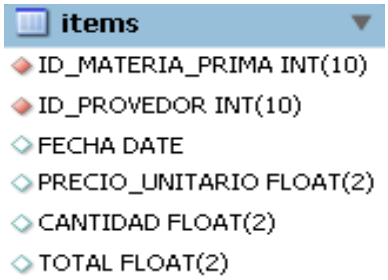


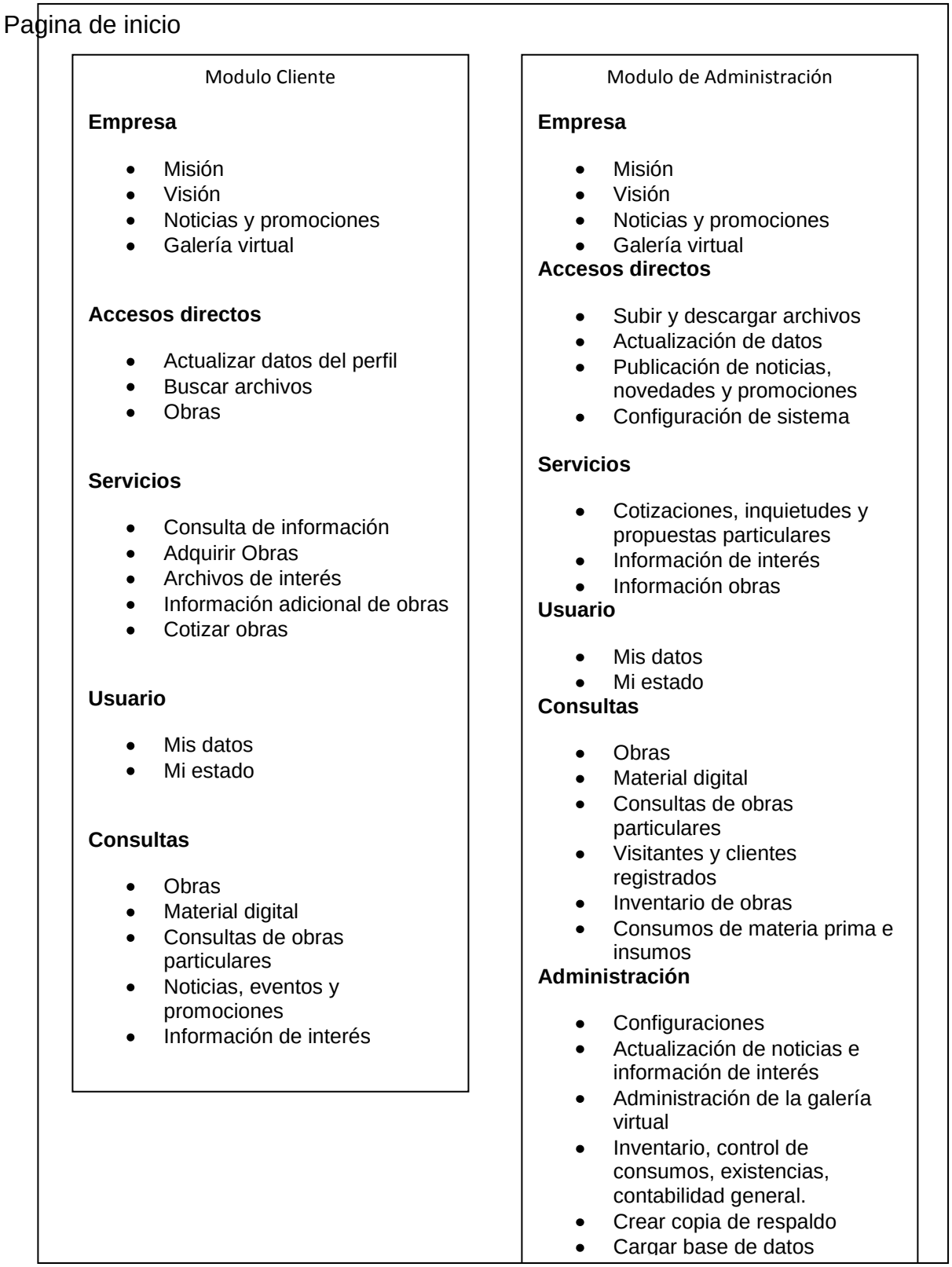
Tabla items: Esta tabla se genero gracias a la relación entre materia_prima y proveedor, es de suma importancia, ya que en ella se almacena la información pertinente a la materia prima e insumos adquiridos; el nombre del producto comprado, la cantidad, el precio, el proveedor que lo suministra, y la fecha del reporte; de esta manera se almacena información de importancia para el correspondiente inventario de materiales.

Figura 20: Tabla ítems



4.2.3 Diagrama de módulos del sistema

Figura 21: Diagrama de módulos del sistema



4.2.4 Diagrama de actividades usuario

Figura 22: Diagrama de actividades para el registro de un usuario

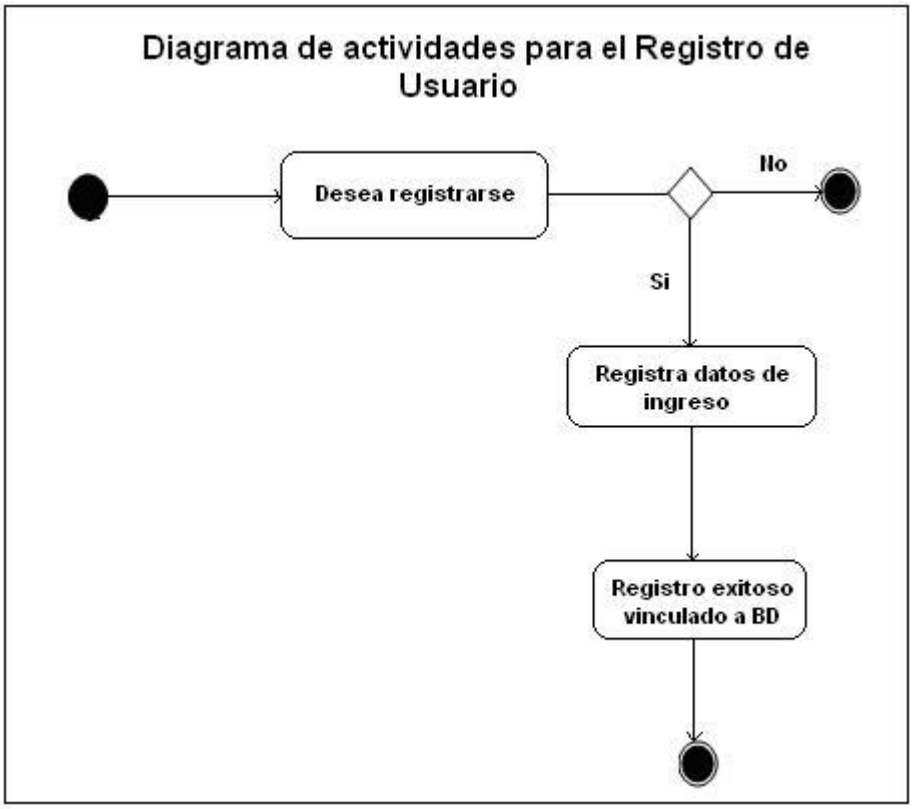


Figura 23: Diagrama de actividades para el ingreso de un usuario

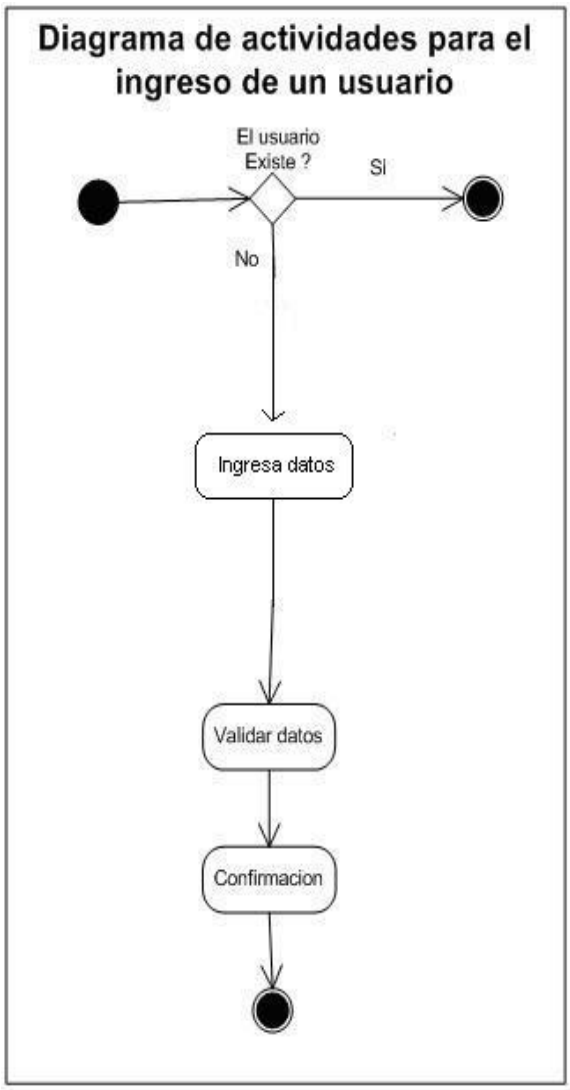


Figura 24: Diagrama de actividades para la validación de un usuario

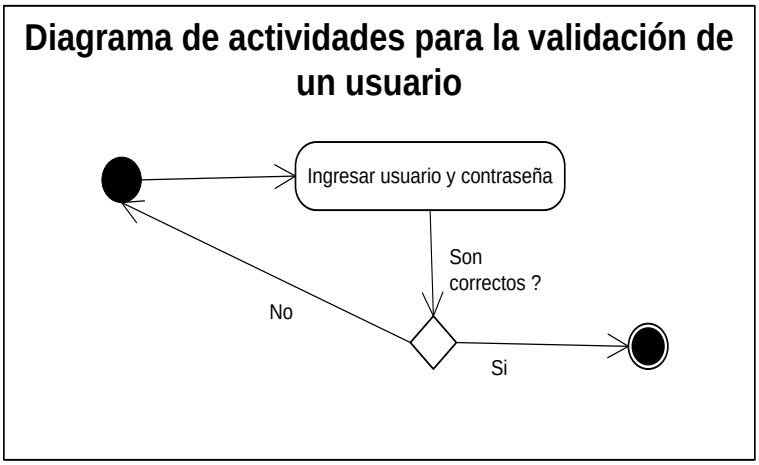


Figura 25: Diagrama de actividades para la adquisición de obras

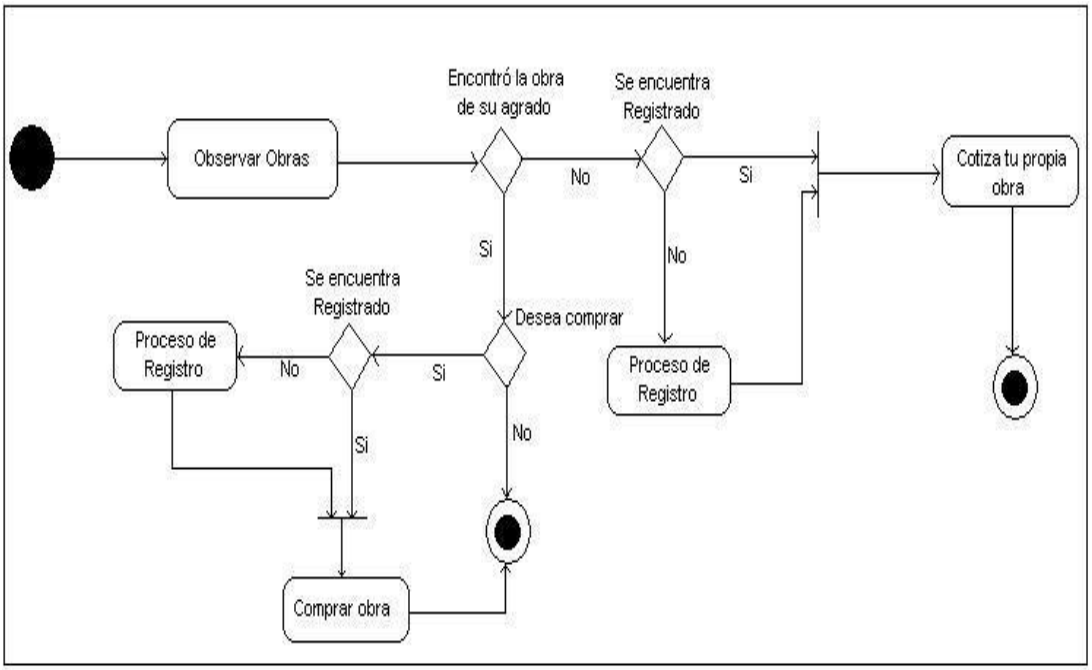
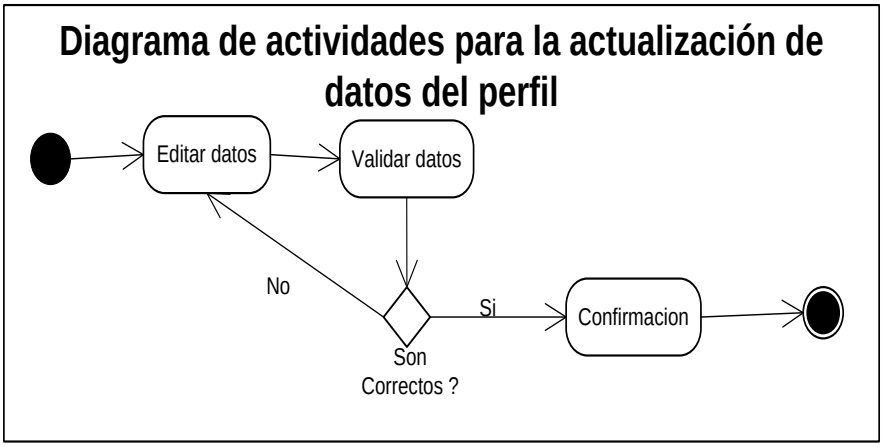


Figura 26: Diagrama de actividades para la actualización de los datos del perfil



4.2.5 Diagrama de actividades Administrador

Figura 27: Diagrama de actividades para el registro de administrador

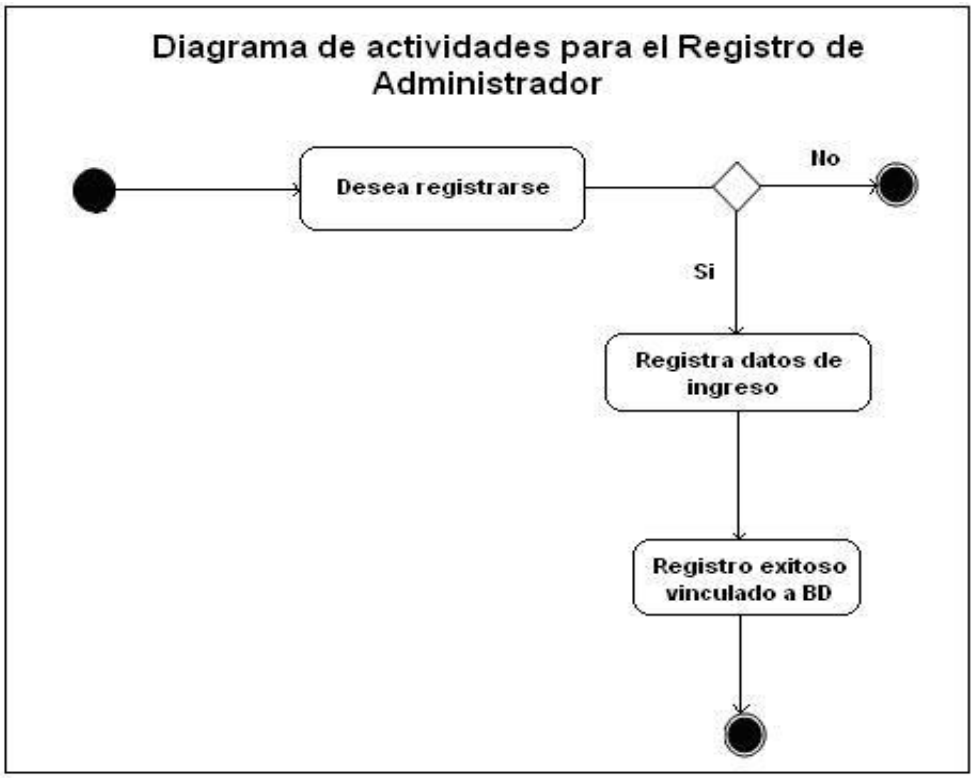


Figura 28: Diagrama de actividades para el ingreso al módulo de administración

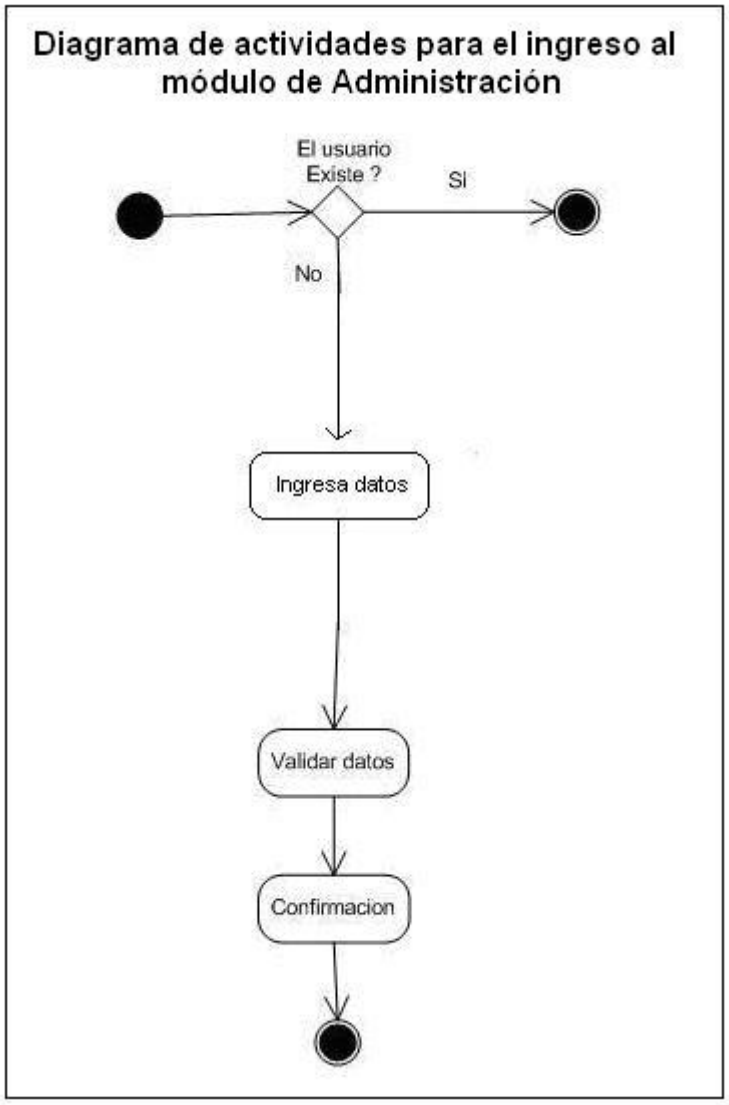


Figura 29: Diagrama de actividades para la validación de información de acceso al módulo de administración

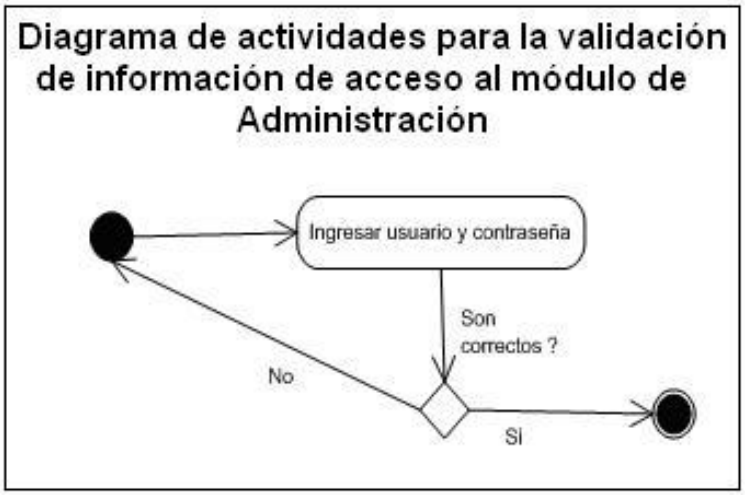


Figura 30: Diagrama de actividades para la actualización y control de archivos e información

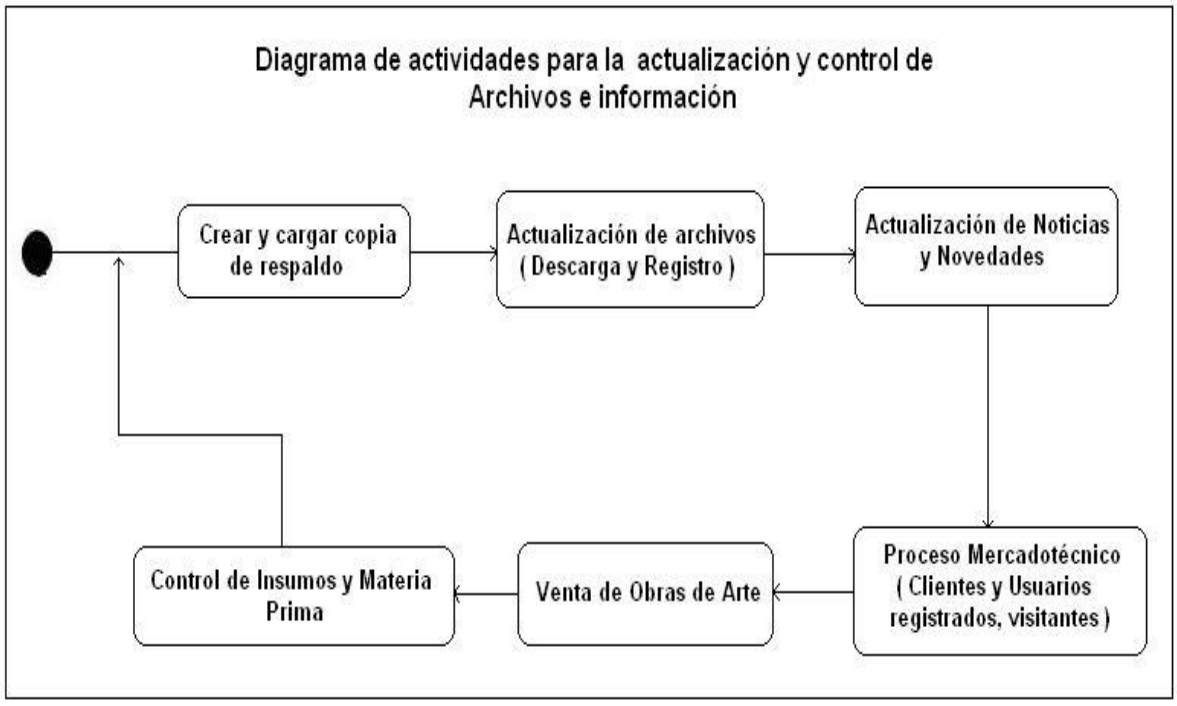


Figura 31: Diagrama de actividades para el control de insumos y materia prima

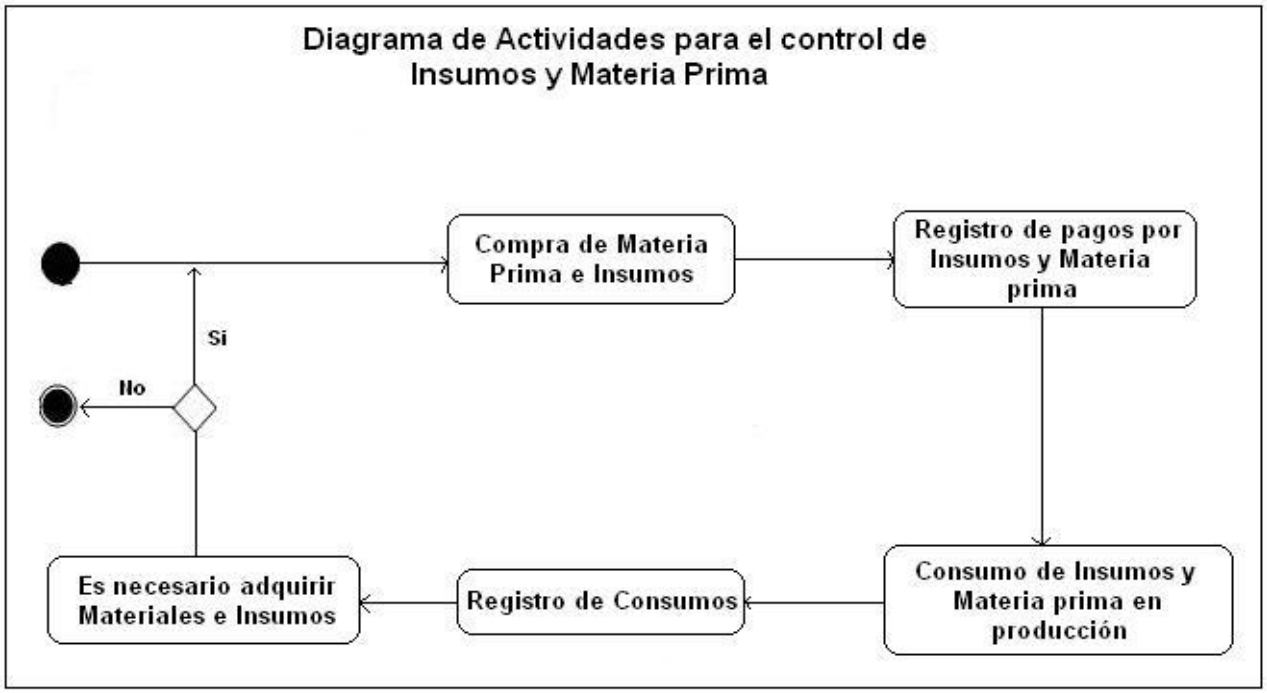


Figura 32: Diagrama de actividades para edición de datos del administrador

4.3 DESARROLLO

FASES DEL PROYECTO:

Fase De análisis: esta fase comienza con el reconocimiento y diagnostico del proceso actual; se estudio en detalle la forma como se lleva a cabo el proceso que se sigue desde la vinculación de tema de proyecto, las diferentes actividades que se generan en el proceso de gestión y administración de los mismos.

Fase de diseño: En esta fase se elaboran actividades como el diseño de base de datos, diagrama entidad relación (Tablas, Procedimientos, Consultas), al igual que un bosquejo general de los diagramas de casos de uso y clases del sistema, como también un modelo de las interfaces de usuario.

Fase de Desarrollo: En esta fase se da paso a la configuración de herramientas y tecnologías a utilizar para el desarrollo del sistema, se da paso a la elaboración de la base de datos, construcción de las paginas Web para la captura, registro, actualización, mantenimiento, informes, validación, prueba, y construcción de los mecanismos de seguridad que controlan los accesos a la base de datos, elaboración de los manuales de usuario del sistema y texto correspondiente al presente proyecto.

Fase de prueba: En esta instancia se procede a la depuración del código, ejecución y puesta en marcha del sistema por parte de los interventores del sistema. Se alimento el sistema inicialmente con datos prueba para la ejecución de los diferentes procesos.

Para llevar a cabo la implementación del sistema, la información y los archivos se guardaron directamente en un servidor web, dicho servidor trabaja bajo el sistema operativo Linux.

4.3.1 Pruebas

Las pruebas son parte fundamental de un producto software antes de su entrega definitiva, con base en este concepto se llevaron a cabo tres tipos de pruebas:

4.3.2 De tiempo de desarrollo

Este tipo de pruebas son aquellas que se realizan informal y periódicamente por parte de los desarrolladores, durante la etapa del desarrollo del software. Por ende, no tiene un orden definido. Estas pruebas permiten verificar la funcionalidad de cada módulo y fueron realizadas durante toda la etapa de programación y empalme de los módulos que conforman el software

4.3.3 Pruebas de aceptación

Son aquellas donde el potencial usuario comprueba la funcionalidad del sistema, y determina si acepta el software como está o precisa aplicar nuevas optimizaciones y soluciones de fallas. Para ésta parte se contó con la colaboración del futuro administrador del sistema y potenciales clientes, dando una valoración positiva sobre el manejo de la herramienta y su funcionalidad.

4.3.4 De validación

En ésta prueba, el software totalmente ensamblado se prueba como un todo, para comprobar si cumple los requisitos funcionales de rendimiento, facilidad de mantenimiento, recuperación de errores, etc. Al inicio de éstas pruebas se usaron los módulos por separado, para corroborar su correcto funcionamiento, por ejemplo: que los campos numéricos sólo aceptaran números, que las direcciones

de correo electrónico tuvieran el formato correcto y la verificación de los campos obligatorios, entre otros.

Es importante anotar que el desarrollo a través de módulos presenta ventajas, en especial en un sistema como el desarrollado en el presente proyecto, debido a que se pueden signar responsabilidades a los desarrolladores sobre determinadas partes del código que, posteriormente harán parte de todo el sistema

Además de las pruebas anteriormente mencionadas, se llevaron a cabo las siguientes:

Pruebas de navegación en los navegadores Internet Explorer 8.0, Mozilla Firefox 3.5, Safari 3.1.1 y Google Chrome; dando como respuesta, resultados favorables para la visualización del sitio web

El sistema de información estuvo instalado durante su proceso de desarrollo en un servidor de pruebas local (equipo de escritorio) donde fue desarrollado y donde se fueron haciendo las pruebas de funcionalidad respectivas. En este equipo se creo y testeo la conexión a base de datos, se manipulo la información por medio de los navegadores entre otras funcionalidades.

La instalación definitiva se llevo a cabo en un servidor con sistema operativo Linux. **Servidor de pruebas: fastdomain.com, S.O. Linux kernel 2.6**

Además de las pruebas anteriormente mencionadas, durante el desarrollo de cada submódulo se llevaron a cabo dos tipos de pruebas: **pruebas en tiempo de desarrollo** y **pruebas después de la programación**, con el objetivo de detectar posibles fallas además de interactuar en un proceso real con el sistema.

En el desarrollo de cada submódulo, las tablas que hacían parte de este, eran llenadas con registros falsos con el objetivo de comprobar la integridad de los datos y verificar la funcionalidad de los procesos

Las pruebas en tiempo de desarrollo fueron realizadas durante toda la etapa de programación de las funcionalidades que conformaban cada submódulo

Las pruebas después de la programación se efectuaron cuando se considero que el sistema de información estaba terminado en su totalidad. El objetivo principal de estas pruebas finales de tipo funcional consistió en buscar fallas o errores específicos como

- Fallas en la interfaz del usuario.
- Funciones incorrectas o ausentes
- Errores en estructura de datos
- Errores de rendimiento
- Errores de inicialización y terminación

Algunos detalles que se tuvieron en cuenta al realizar estas pruebas fueron:

- Enlaces entre las paginas correspondientes al registro e inicio de la sesión
- Conexión a la base de datos
- Validación de los datos a ingresar en la base de datos
- Ingresos de registros
- Ejecución de sentencias SQL
- Inicio de sesión y manejo de variables de sesión
- Envío de formularios a paginas receptoras
- Modificación y eliminación de registros
- Interfaz grafica
- Correcta ejecución de las consultas SQL resultados de búsqueda

Herramientas de desarrollo

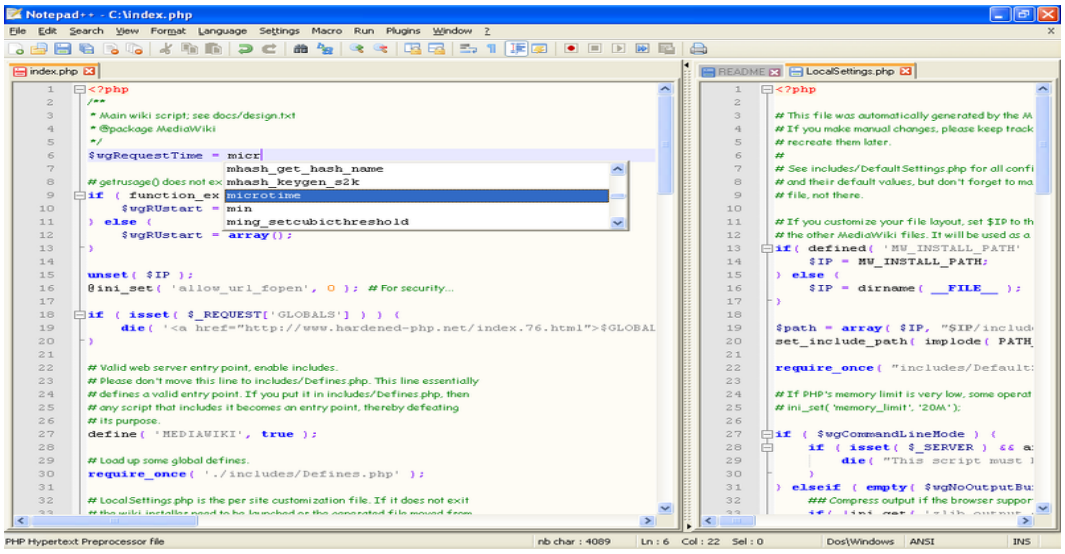
Notepad ++ : Es un editor de texto y de código fuente libre con soporte para varios lenguajes de programación para Microsoft Windows.

Gracias a su velocidad, puede convertirse en una alternativa al bloc de notas. Con la implementación de navegación por pestañas, moverse entre los archivos de texto abiertos es más cómodo.

Por defecto incluye la extenxión TextFX que proporciona muchas opciones de transformación de texto.

Aunque Scintilla no permite la búsqueda y reemplazo de expresiones regulares múltiples, Notepad++ permite el uso de complementos que ayudan a mitigar este hecho.

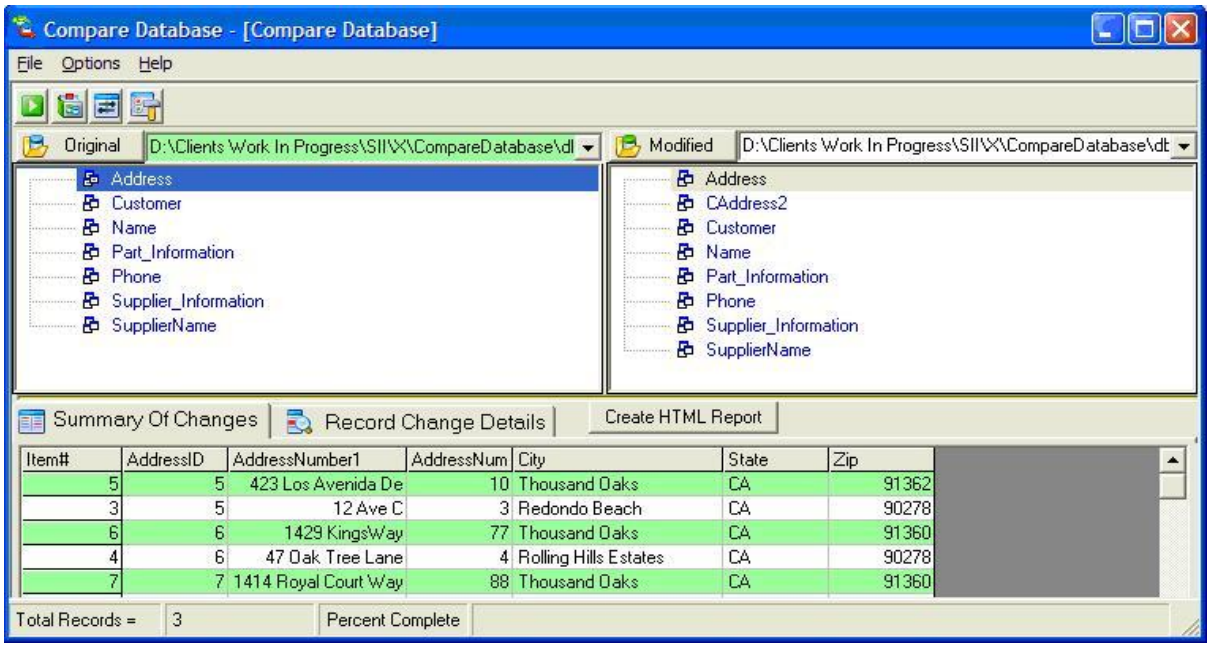
Figura 33: Pantallazo de Notepad ++ en ejecución



Compare tool : Permite realizar **comparaciones entre bases de datos**, entre comparando de forma sencilla bases de datos SQL Server.

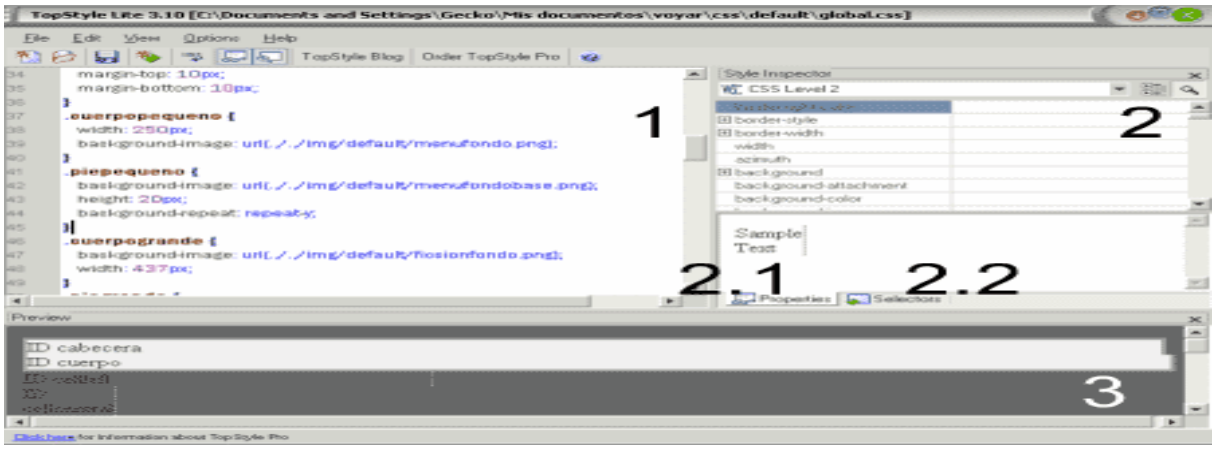
La aplicación principal consiste en cuatro ventanas. Las dos de la parte superior muestran los objetos de la base de datos que están en la base de datos original y en la modificada. Las dos de la parte inferior se usan tanto para mostrar el informe de diferencias de datos como para el informe estructural, dependiendo de lo que estés ejecutando.

Figura 34: Pantallazo de Compare Tool en ejecución



TopStyle Lite

Figura 35: Pantallazo de TopStyle Lite en ejecución



La pantalla de trabajo de TS Lite se divide en tres paneles principales. Siguiendo las indicaciones de la imagen superior, son los siguientes:

1) Editor. Se trata de la zona principal donde podemos editar a mano los elementos, identificadores y clases. Aunque el trabajo es manual, cuenta con la ventaja de que a medida que escribimos, se abre una lista desplegable que incluye todas las opciones posibles para cada atributo. Por ejemplo, si escribimos margin, se abrirá la lista que incluye: margin-top, margin-left, ..., margin-color, margin-width, etc, así como una lista de valores: colores (contiene una paleta y un selector al vuelo con el que es posible reconocer cualquier color del monitor, incluyendo las ventanas de otros programas distintos a TopStyle), tamaños porcentuales y fijos con sus correspondientes unidades demedida, etc.

2) El inspector de estilos muestra el listado de todos los atributos posibles, e incluye un pequeño panel de previsualización de texto de ejemplo con el estilo seleccionado en cada momento. El inspector cuenta a su vez con dos pestañas inferiores:

2.1) Properties, que aparece por defecto y muestra el mencionado listado de atributos y valores y la previsualización.

2.2) Selectors. Esta pestaña da acceso a un cómodo esquema que contiene: elementos, clases e identificadores. Es especialmente útil para obtener una visión global y sencilla de todos los estilos utilizados. Además, si seleccionamos uno, éste quedará también seleccionado en el panel de edición.

3) Vista Previa: Establece una vista preliminar de todos los estilos empleados en el documento, uno después de otro.

SQLyog Es un administrador de bases de datos MySQL para Windows que, recientemente, ha liberado su versión Community como código abierto, gratuita para uso no comercial.

Figura 36: Pantallazo de SQL yog en ejecución

Editor de codigo PHP: netbeans 6.5.7

Este editor trabaja del lado servidor embebido en HTML y usado para crear páginas Web dinámicas). Pero en la versión 6.5 el soporte es completo e integrado a nivel de producto. Ofrece dispositivos como compleción de código, codificación coloreada por semántica e integración a bases de datos.

NetBeans 6.5 soportadesarrollo con JavaFX además es un editor para desarrollo JavaScript con compleción de código CSS/HTML y capacidad de debuggin del código del lado cliente en los browsers Firefox y Explorer. También tiene debugging de tecnologíasJava en modo multithreaded, además de soporte ampliado para Spring, Hibernate, Java Server Pages y la API Java Persistence.

Esta versión viene con el servidor de aplicaciones GlassFish 3.0. GlassFish 3.0 apunta a la capa Web para servicio de aplicaciones Web y tiene un diseño muy modular. Ocupa muy pocos recursos, tanto como unos 100 KB de base, para incorporar nueva funcionalidad a medida que la requiere.

Servidor de base de datos : MySQL 5 El software MySQL proporciona un servidor de base de datos SQL (Structured Query Language) muy rápido, multi-threaded, multi usuario y robusto. El servidor MySQL está diseñado para entornos de producción críticos, con alta carga de trabajo así como para integrarse en software para ser distribuido. MySQL es una marca registrada de MySQL AB.

Sistema desarrollado con PHP 5 en apache2

La versión más reciente de PHP es la 5.3.0 (30 de junio de 2009), que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- ✓ Mejor soporte para la Programación Orientada a Objetos, que en versiones anteriores era extremadamente rudimentario, con PHP Data Objects.
- ✓ Mejoras de rendimiento.
- ✓ Mejor soporte para MySQL con extensión completamente reescrita.
- ✓ Mejor soporte a XML (XPath, DOM, etc.).
- ✓ Soporte nativo para SQLite.
- ✓ Soporte integrado para SOAP.
- ✓ Iteradores de datos.
- ✓ Manejo de excepciones.
- ✓ Mejoras con la implenetacion con oracle.

Editor de contenidos: FCKEditor.

CKEditor es un editor de texto HTML/ WYSIWYG de código abierto que provee a la web del poder de las aplicaciones de escritorio al estilo de editores como Microsoft Word, sin la necesidad de instalar ningún componente en la computadora del cliente.

Framework para ajax: XAJAX

Xajax es una biblioteca de código abierto para PHP que permite crear de manera fácil y simple aplicaciones Web basadas en AJAX usando además HTML, CSS, y Javascript. Las aplicaciones desarrolladas con Xajax pueden comunicarse asíncronamente con funciones que se encuentran del lado del servidor y así actualizar el contenido de una página sin tener que recargarla nuevamente, su

última versión es la 0.5 Final que cambia ligeramente comparado con las versiones anteriores 2.5.x y anteriores.

En un principio se crea una instancia de objeto Xajax (**xajax object**). Este objeto manejará todo el procesamiento a través de Xajax. En segundo lugar debemos registrar todas las funciones que hemos definido previamente en el objeto Xajax, esto se puede hacer usando el método **xajax->register()**. Finalmente todas las respuestas serán procesadas utilizando el método **xajax->processRequest()**.

Librería de estilos visuales: jQuery

jQuery es una biblioteca o framework de Javascript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el arbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas web. Fue presentada en enero de 2006 en el BarCamp NYC.

jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en Javascript que de otra manera requerirían de mucho más código. Es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

5 CONCLUSIONES

- ✓ El sistema de información para la Galería, entra a formar parte activa de la comunidad de clientes, visitantes y en general de todos aquellos quienes aprecian el arte, convirtiéndose en una herramienta que ofrece una manera sencilla para manejar el inventario y el control de los diferentes recursos necesarios en el correcto proceso de desarrollo que se lleva a cabo en la Galería.

- ✓ El entorno dinámico y auto administrativo del sistema de información, facilita el impulso publicitario y gerencial del proceso de desarrollo y finalmente del producto terminado.

- ✓ El sistema de información se hizo totalmente con software libre lo cual hizo que disminuyeran los costos del desarrollo de éste proyecto. Es importante resaltar que estas herramientas como son PHP y MySQL, son usados ampliamente en aplicaciones web en todo el mundo, lo que facilita participar interactivamente con numerosas comunidades conocedoras de los lenguajes usados, además de otras herramientas como foros y boletines que brindan apoyo cuando se presenta alguna falencia en cuanto a conocimiento, o cuando surge algún inconveniente.

- ✓ El diseño de la interfaz gráfica implementada, cumple a cabalidad con las diversas exigencias, manteniendo un equilibrio uniforme para la presentación de las obras, y el debido proceso publicitario, así como una interacción fácil y amigable con la aplicación.

- ✓ El lenguaje de Unificado de Modelado UML, permite condensar en diagramas todos los aspectos del proyecto, teniendo en cuenta los distintos puntos de vista de cada una de las personas que serán usuarios del sistema

- ✓ La implementación del sistema de información ingresa a formar parte activa de la reducción de costos en el desarrollo de las obras de arte, proveyendo a su administrador de información verás para el control objetivo de insumos y recursos, factores primordiales en el precio final del producto terminado.

- ✓ Se dio cumplimiento a todos los objetivos planteados al inicio del proyecto, lo cual se puede demostrar ingresando al portal de la Galería de Arte y haciendo uso de los servicios que presenta el sistema.

- ✓ El sistema de información, ingresa a mejorar y a fortalecer de manera definitiva y eficaz el impulso publicitario de la galería y las obras que allí se desarrollan, rompiendo los esquemas de la publicidad local y nacional, para convertirse en un lanzamiento en la que el mundo entero tendrá la posibilidad de familiarizarse con los hermosos productos desarrollados en la galería.

- ✓ Se diseñó una base de datos relacional que garantiza la integridad, confiabilidad y no redundancia de datos; basados en un desarrollo que tuvo en cuenta la correcta validación de los datos antes de ser ingresados.

6 RECOMENDACIONES

- Implementar otras formas de pago, como son los pagos con tarjetas de crédito y/o débito.

- Hacer uso de los diversos recursos de actualización de la información y los diversos contenidos, para ofrecerle a los visitantes contenidos frescos que visitar.

- Es de suma importancia que el administrador del sistema este atento a las nuevas amenazas de seguridad que se presenten, y ante ello, buscar las medidas necesarias para evitar problemas de seguridad.

- Robustecer el sistema de información mediante el desarrollo de nuevos módulos, y mejorar los que han sido desarrollados con este proyecto, generando paulatinamente un sistema integral ante las diversas necesidades.

- Implementar en la plataforma web, foros de discusión, que alienten a los diversos visitantes y potenciales clientes a la creación de debates y espacios de discusión, alentándolos diariamente a que visiten el sitio.

BIBLIOGRAFÍA

LIBROS

PRESSMAN, Roger S ; ingeniería del software. Un enfoque práctico. McGraw Hill 5ta edición, Madrid España 2002

JACOBSON, Ivar, BOOCH, Grady, RUMBAUGH, James; El proceso unificado de desarrollo de software . 1 Ed Español. Addison Wesley – España

PROYECTOS DE GRADO

BOTELLO LUGO, Carlos Eduardo, OROZCO VALLEJO, Rafael Hernando, ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DE UNA HERRAMIENTA SOFTWARE QUE PERMITE LA GENERACIÓN DE PORTALES WEB CONFIGURABLES PARA LA ADMINISTRACIÓN DE LOS PROGRAMAS DE POSGRADO DE LA UNIVERSIDAD INDUSTRIAL DE SANTANDER. UIS Bucaramanga 2008

FLOREZ, Jan, RUEDA NÚÑEZ, Jorge Mario; SISTEMA DE INFORMACIÓN DE AMBIENTE WEB DE LA ESCUELA DE INGENIERÍA DE PETÓLEOS DE LA UNIVERSIDAD INDUSTRIAL DE SANTANDER. UIS Bucaramanga 2008

SERRANO ZAMBRANO, Walter, DISEÑO E IMPLEMENTACIÓN DE UNA APLLICACIÓN WEB QUE PERMITA RESOLVER PROBLEMAS DE FORMA CONJUNTA EN LAS ORGANIZACIONES, UIS Bucaramanga 2008

SITIOS WEB:

Desarrollo de software, portal donde se encuentran documentos con el material utilizado en la Universidad Politécnica de Valencia
https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Forms/AllItems.aspx

http://www.gamarod.com.ar/recursos/tutoriales/php/

http://tutorialphp.net/

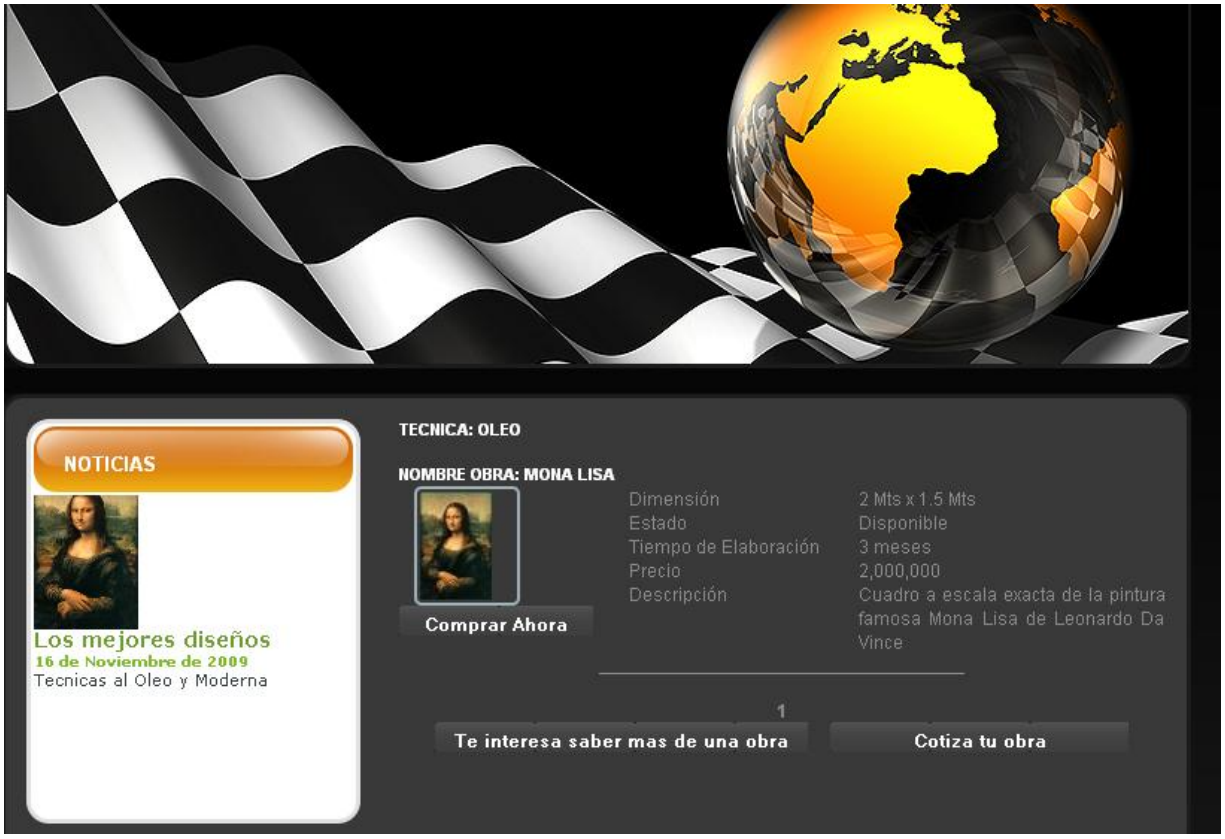
http://www.php.net/tut.php

http://www.phpya.com.ar/

http://www.desarrolloweb.com/php/

http://www.webexperto.com/tutoriales/17/php/

ANEXO A: MANUAL DEL USUARIO

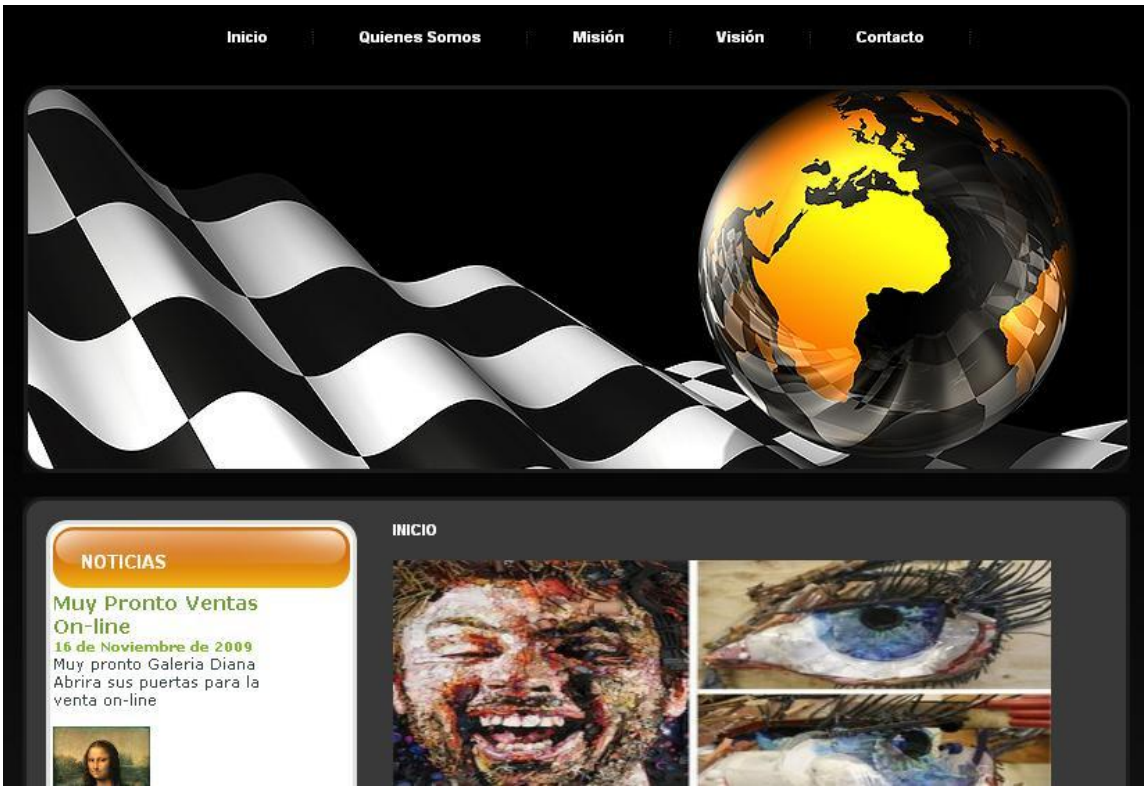


Gráfica A.1 pantalla de bienvenida del usuario

En la imagen anterior se muestra la visualización general del sistema al usuario, en donde se podrá observar los links de acceso generales, la sección de noticias y novedades, la sección de registro, y la galería virtual de obras.

A.1 Menú Principal

Esta parte de la plataforma web, le provee al usuario de toda la información pertinente a la galería de arte, además de poder observar las diversas promociones y novedades que ofrece la empresa en su modulo de noticias.



Gráfica A.2 menú principal

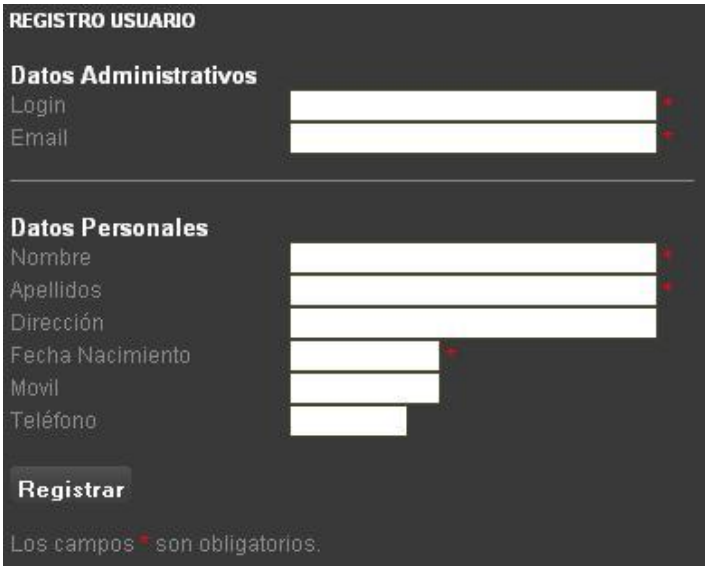
A.2 Registro

Este menú se usará para que los visitantes y clientes potenciales de la galería, se registren ingresando sus datos generales, esto con la finalidad de acceder a

servicios como cotizar obras particulares, o realizar el proceso de confirmación de pago de obra.

Gráfica A.3 Datos del usuario

En esta parte del menú se podrán editar datos, de acuerdo a las respectivas novedades del usuario, al indicar la opción registrarse el formulario será el siguiente, en donde el usuario procederá a realizar el debido proceso de registro, es un formulario totalmente validado.

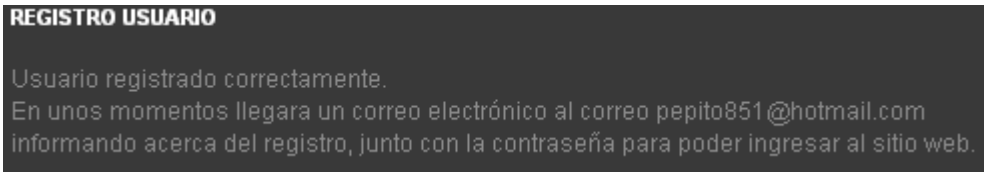


Gráfica A.4 Formulario de Registro

El usuario procede a llenar los datos, posteriormente le da registrar.

Gráfica A.5 Formulario de Registro con datos

El usuario obtiene una respuesta de registro, indicándole que a su correo le ha llegado información del registro.



Gráfica A.6 Mensaje después de haber terminado de Registrarse

En su correo llegará un mensaje así.

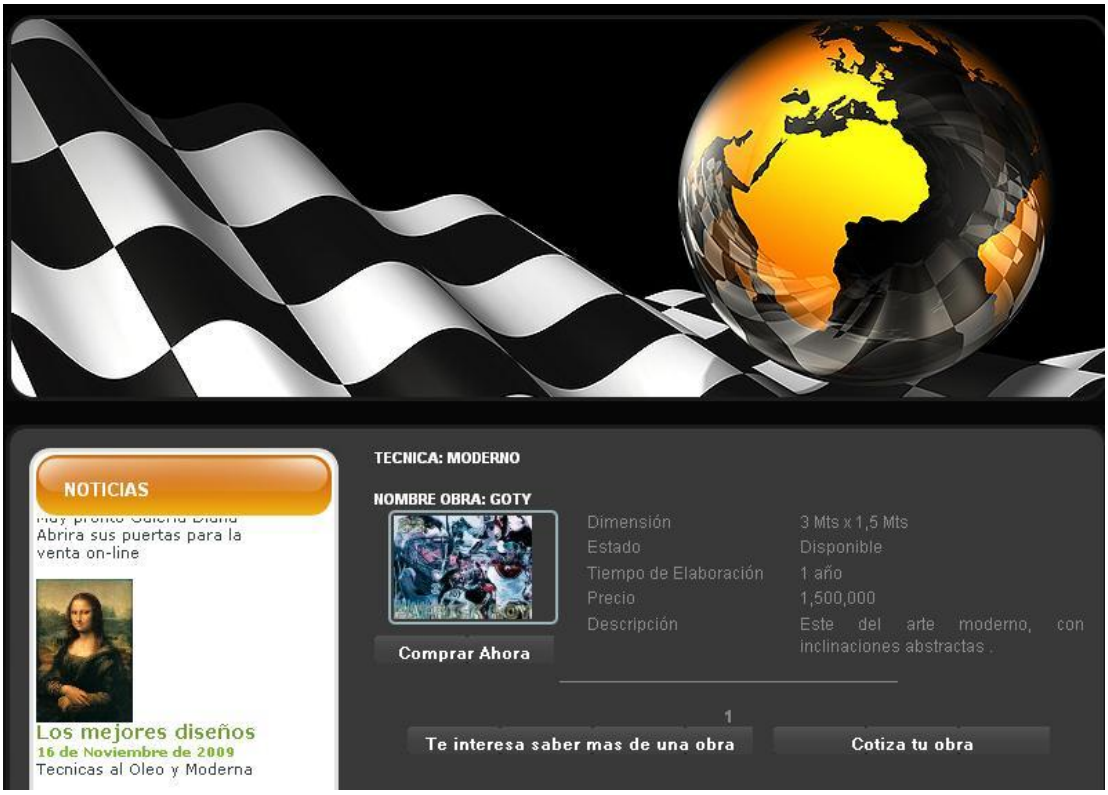
Gráfica A.7 Cuerpo del correo enviado después del registro

Datos que el sistema genera por defecto pero que posteriormente podrá editar.

Gráfica A.8 Opciones del módulo perfil

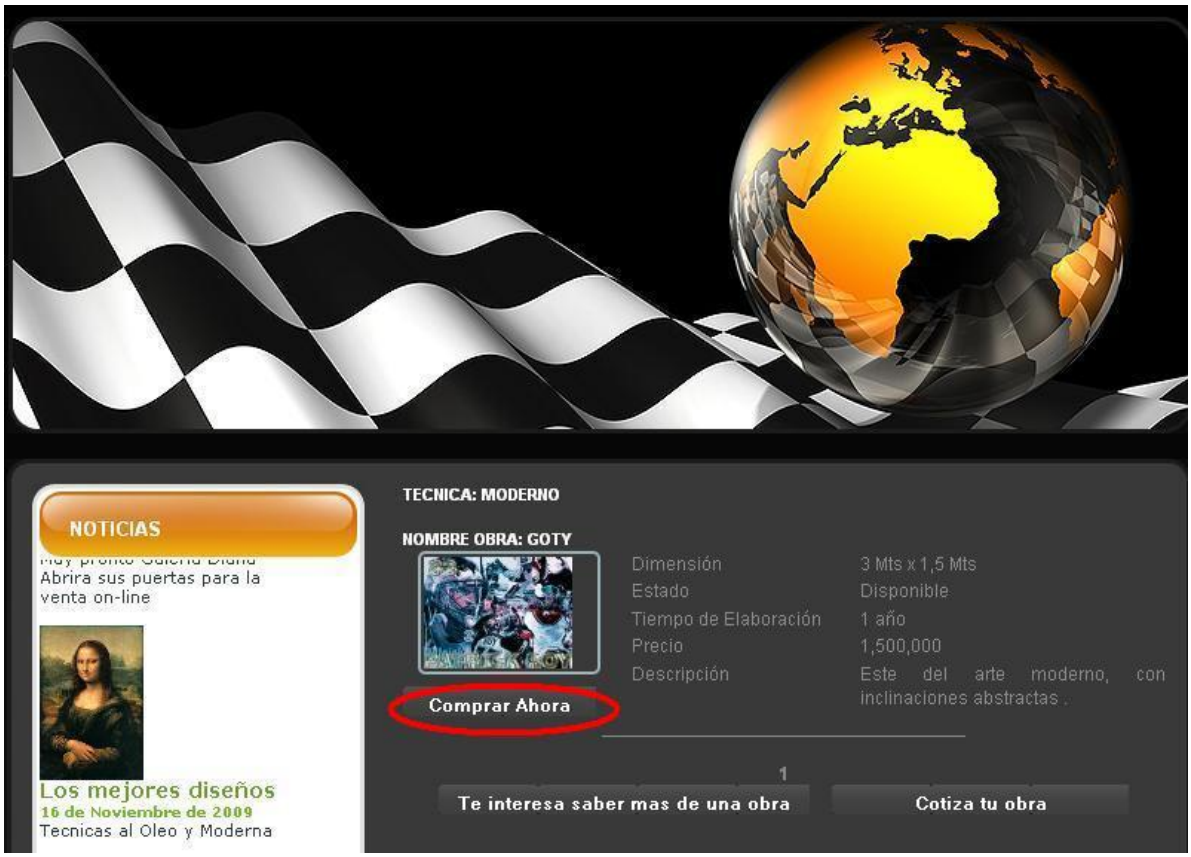
A.4 Obras

En esta sección se podrán visualizar todas las obras que la galería tiene a la venta, cada una de ellas cuenta con sus datos generales, clasificadas según la técnica de elaboración.



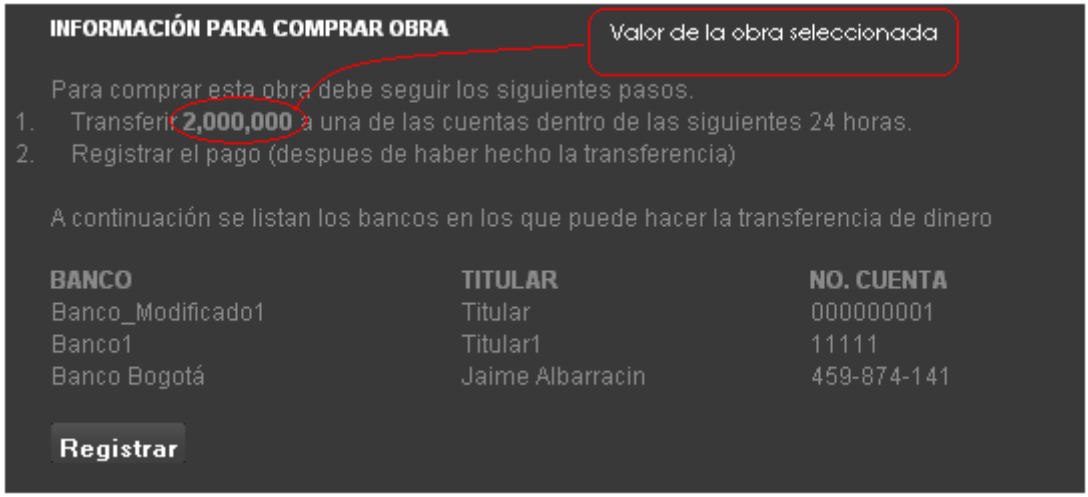
Gráfica A.9 Galería virtual

Si el usuario esta interesado en algunas de las obras allí exhibidas, tiene la posibilidad de realizar el debido proceso de adquisición, a través del link **COMPRE AHORA**.



Gráfica A.10 Al cliquear el botón señalado muestra la información.

Al oprimir el link de comprar, se despliega un formulario con la información general de la obra a adquirir, e información pertinente al pago, cabe resaltar que para poder realizar tal operación es necesario que el usuario se encuentre debidamente registrado, de lo contrario deberá registrarse, y así tener la posibilidad de realizar el correspondiente proceso de adquisición.



Gráfica A.11 Información requerida para comprar la obra.

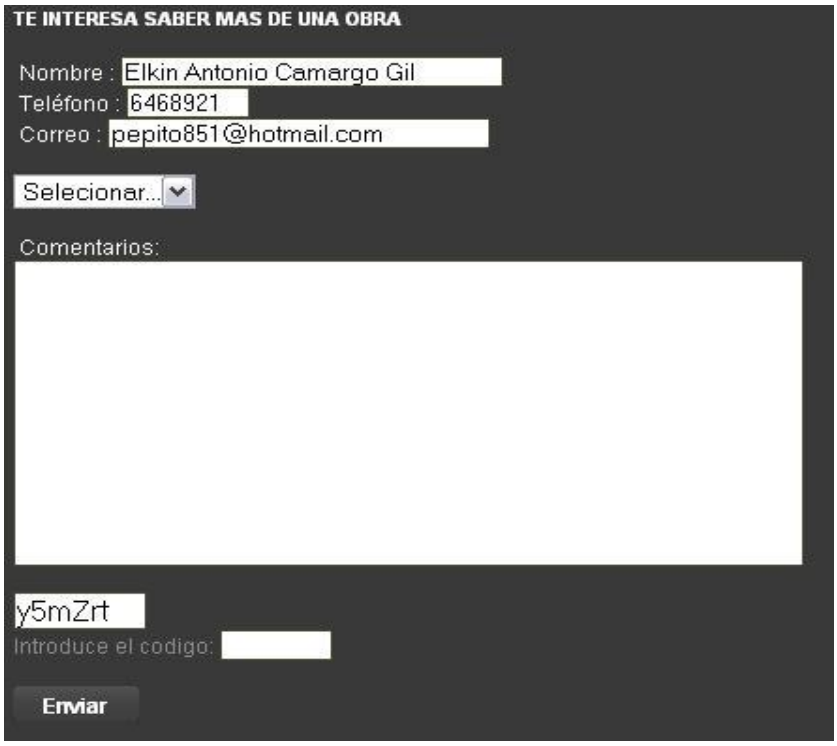
Después de que el cliente haya hecho la consignación puede seguir con el proceso de compra llenando el siguiente formulario.

Gráfica A.12 Formulario para registrar pago de obra.

Al darle clic al botón de registrar, el sistema le envía un correo al administrador informándole que un usuario ha comprado una obra, por lo que se debe comprobar la transferencia y luego proseguir con el envío de la obra.

Si el cliente considera que la información suministrada en la galería virtual no es suficiente, puede solicitar información adicional a través del link **TE INTERESA**

SABER MAS DE ALGUNA OBRA, aquí el interesado tendrá la posibilidad de escribir sus dudas o comentarios al respecto de la obra de interés, la cual previamente seleccionara en la opción seleccionar, en donde se encontraran registradas todas las obras del artista, las dudas, o sugerencias llegarán directamente al correo del artista y así solucione todas las dudas del caso.



Gráfica A.13 Formulario para saber más sobre una obra.

Gráfica A.14 Formulario para saber más sobre una obra (habiendo seleccionado la obra).

Si en definitiva las obras presentadas en la galería virtual no cubren las expectativas del visitante, este tiene la posibilidad de proponer su propia obra, a través del link **COTIZA TU OBRA;** para tal fin es necesario el registro por parte del visitante.

Al oprimir el link de cotización se despliega un formulario en donde el usuario tendrá la posibilidad de escribir sus expectativas correspondientes, además de tener la posibilidad de subir una imagen que le permita dar una idea más específica al artista de sus gustos y requerimientos.

Gráfica A.15 Formulario para cotizar una obra nueva.

ANEXO B: MANUAL TÉCNICO

La plataforma de administración es generada pensando en una interfaz totalmente grafica y amigable para cualquier tipo de usuario, al ingresar en el sitio como administrador, basta sólo con ingresar el nombre de usuario y la contraseña..

Gráfica B.1 Formulario para inicio de sesión.

Ingresados los datos pulsamos el botón Acceder para tener acceso a la sección de administración del sistema, la cual será manipulada solo por el administrador del sistema, el cual tiene todos los permisos.

Gráfica B.2 Menú para el administrador.

El menú principal nos brinda todas las opciones de administración necesarias del sistema de información, módulos que observaremos en detalle mas adelante.

Perfil

Gráfica B.3 Opciones del perfil.

La opción de perfil le permite al administrador cambiar, modificar y actualizar los datos que considere necesarios.

Gráfica B.4 Información de usuario

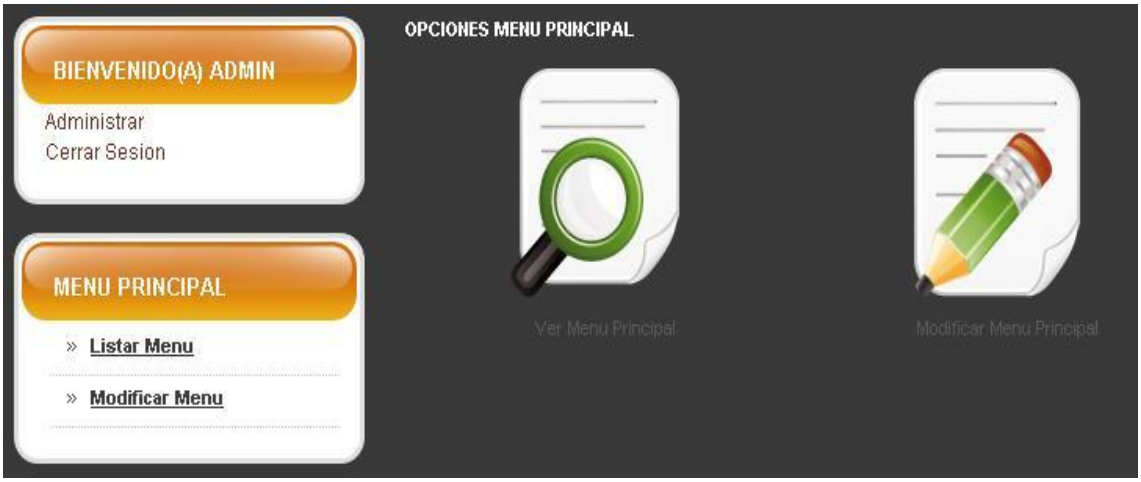
Observamos a continuación la opción de modificar la contraseña, una ventaja que se hace presente en todas las secciones del sistema, proveyendo al administrador la opción de administrar dinámicamente todo el software.

Gráfica B.5 Formulario para cambiar contraseña

A su vez en la opción de **Perfil,** se encuentra la opción de observar todos los usuarios que se encuentran registrados en el sistema, de esta manera se tendrá la posibilidad de realizar diariamente actividades publicitarias y mercadotécnicas.

Gráfica B.6 Información de todos los usuarios registrados

Menú Principal



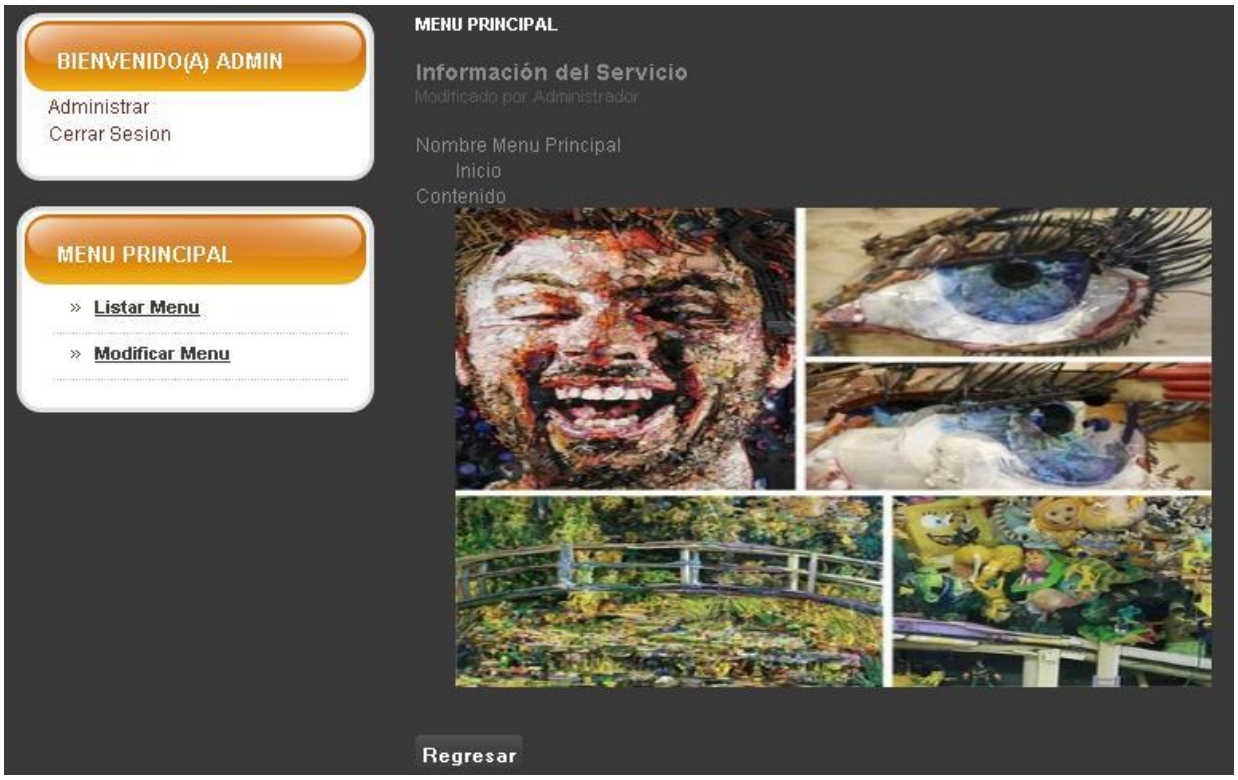
Gráfica B.7 Opciones del menú principal

En la opción del menú principal se le permite al administrador observar y modificar la totalidad del menú, podemos observar que las opciones las encontramos tanto en el recuadro izquierdo, como en la parte central, esto con el fin de afianzar aun mas la accesibilidad y facilidad de administración del sistema de información.

La opción **Listar Menú** Nos permite observar la totalidad del menú observado por los usuarios con la posibilidad de observar los contenidos de cada uno.

Gráfica B.8 Listado del menú principal

Al pinchar cualquiera de los botones azules del listado nos permite ver su contenido, para nuestro caso particular **Inicio.**

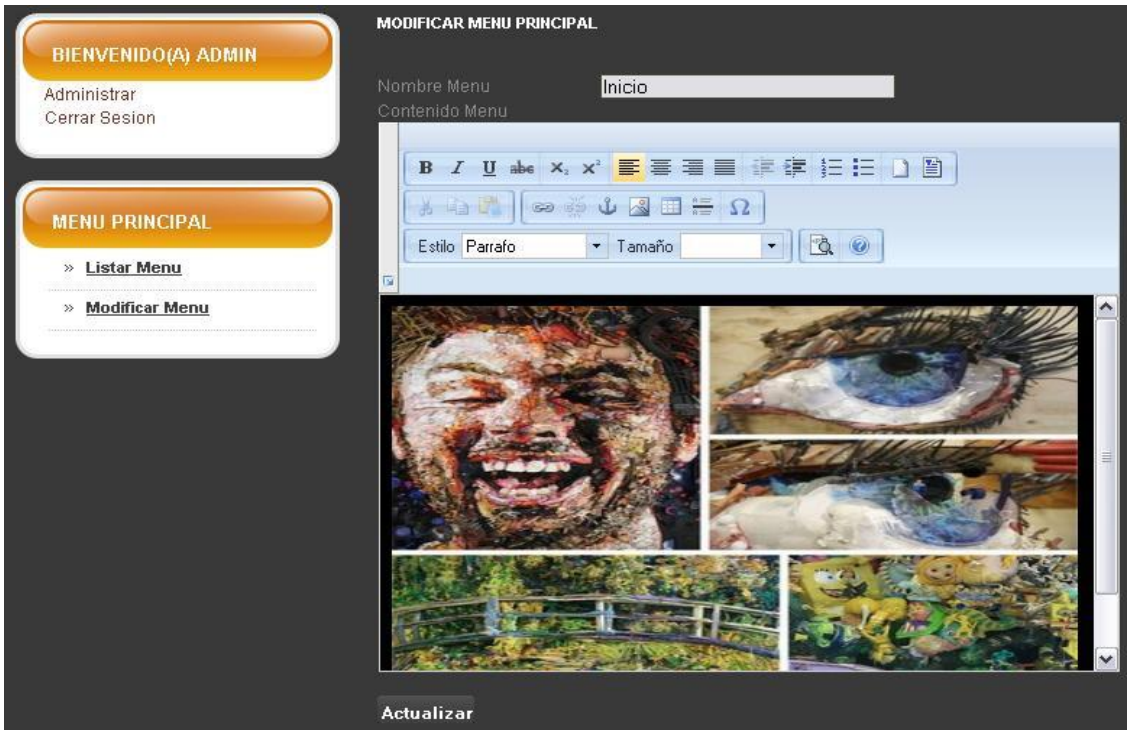


Gráfica B.9 Información del menú inicio

La opción **Modificar Menú** nos permite seleccionar cada uno de los menús que observa el usuario, y así realizarle las correspondientes modificaciones, actualizaciones o cambios, basta solo con seleccionar el botón del menú a modificar y cliqueamos **Modificar**.

Gráfica B.10 Seleccionar menú para modificarlo

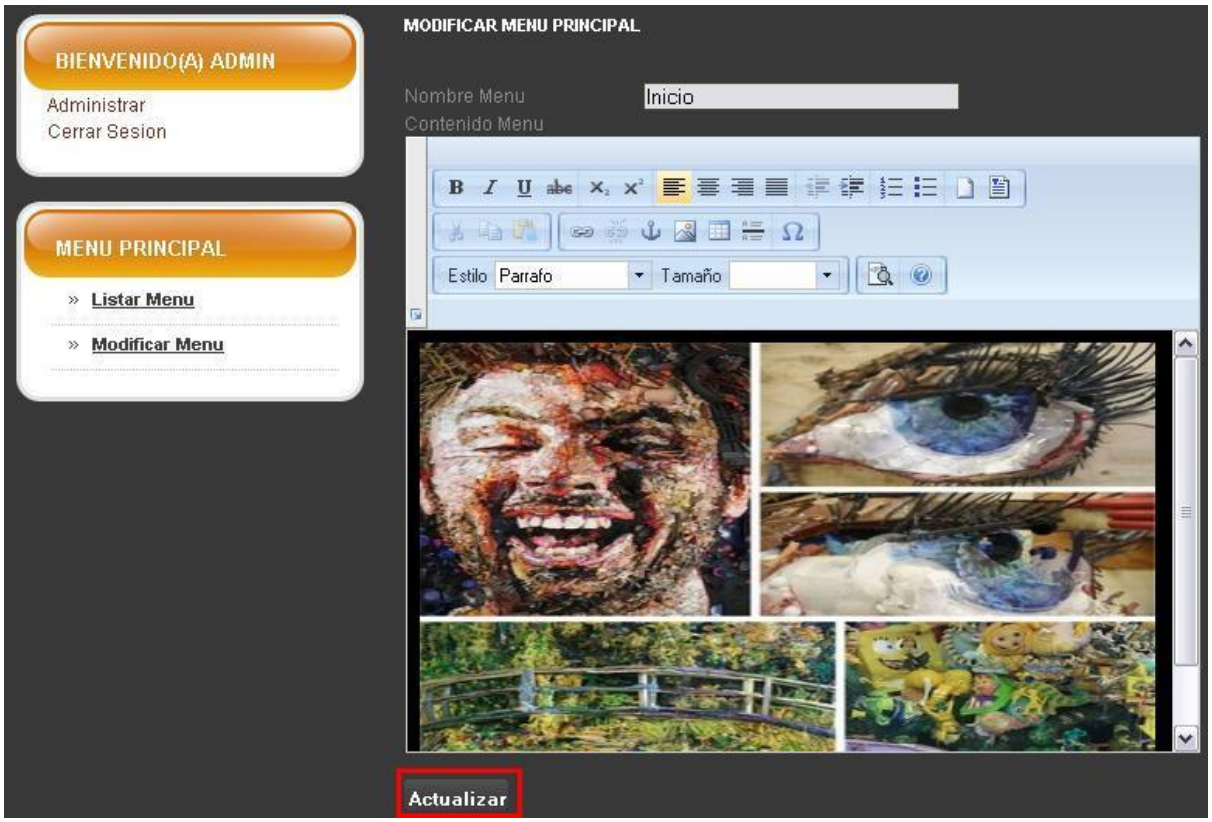
Para nuestro caso particular seleccionamos el botón **Inicio.**



Gráfica B.11 Modificar menú principal

Observamos que se nos carga un editor de contenidos muy similar a la herramienta **Word ofrecida por Windows,** con esta podremos gestionar la actualización, cambio o modificación total de cada uno de los links; de esta manera actualizar el la plataforma web será tan sencillo como escribir un documento en Word o enviar un correo electrónico.

Finalmente gestionamos los cambios o las actualizaciones cliqueando en el botón **Actualizar** para guardar los correspondientes cambios en el menú.



Gráfica B.12 Guardar cambios hecho en el menú principal (Inicio)

Noticias

En esta sección permite administrar las Noticias y novedades que observa el usuario en el recuadro superior izquierdo

Gráfica B.13 Noticia

Gráfica B.14 Opciones de noticia

Listar Noticias Permite observar las noticias montadas hasta el momento.

Gráfica B.15 Listar noticias

Agregar Noticia Permite adicionar una nueva noticia, para tal fin se establecio un gestor de contenidos muy similar a Word, con el que establecer nuevas noticias resulta sencillo de realizar.

Gráfica B.16 Agregar nueva noticia

Modificar Noticia Permite modificar noticias ya establecidas en el sistema, en esta sección seleccionamos entre las noticias que ya se han montado y posteriormente se procede a modificar.

Gráfica B.17 Seleccionar noticia para su modificación

Gráfica B.18 Modificando noticia

Eliminar Noticia Permite eliminar noticias de acuerdo al gusto y necesidades del administrador.

Gráfica B.19 Seleccionar noticia para eliminar

Portafolio

El portafolio provee de todas las herramientas necesarias para administrar la galería virtual.

Gráfica B.20 Opciones del portafolio de obras

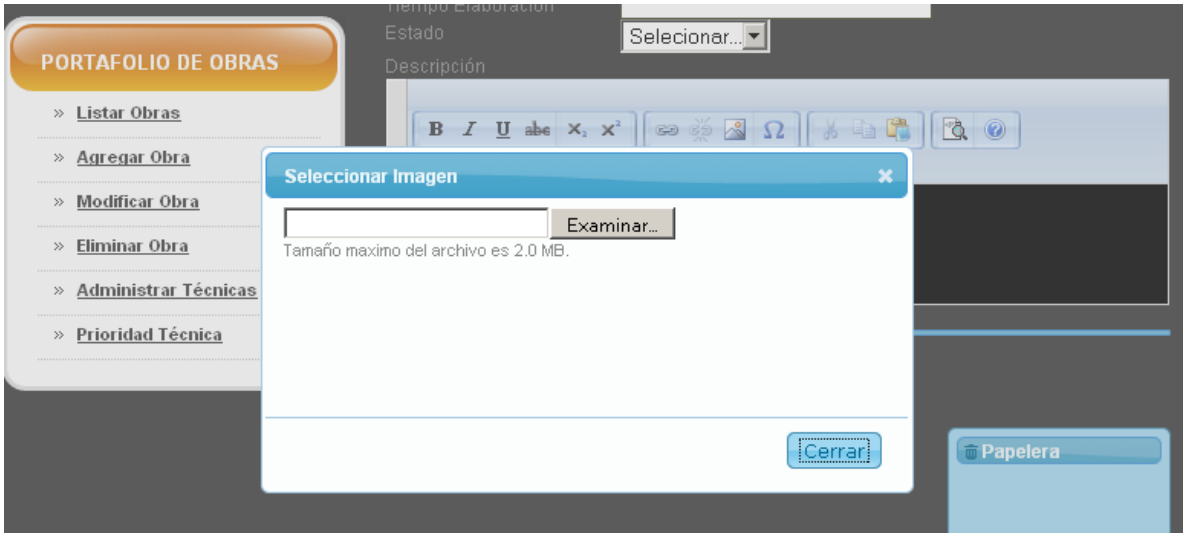
Listar Obras Permite observar las obras establecidas hasta el momento en el sistema.

Gráfica B.21 Listar obras cargadas anteriormente

Agregar Obra Permite agregar nuevas obras, en esta opción previamente ingresamos los datos de la obra.

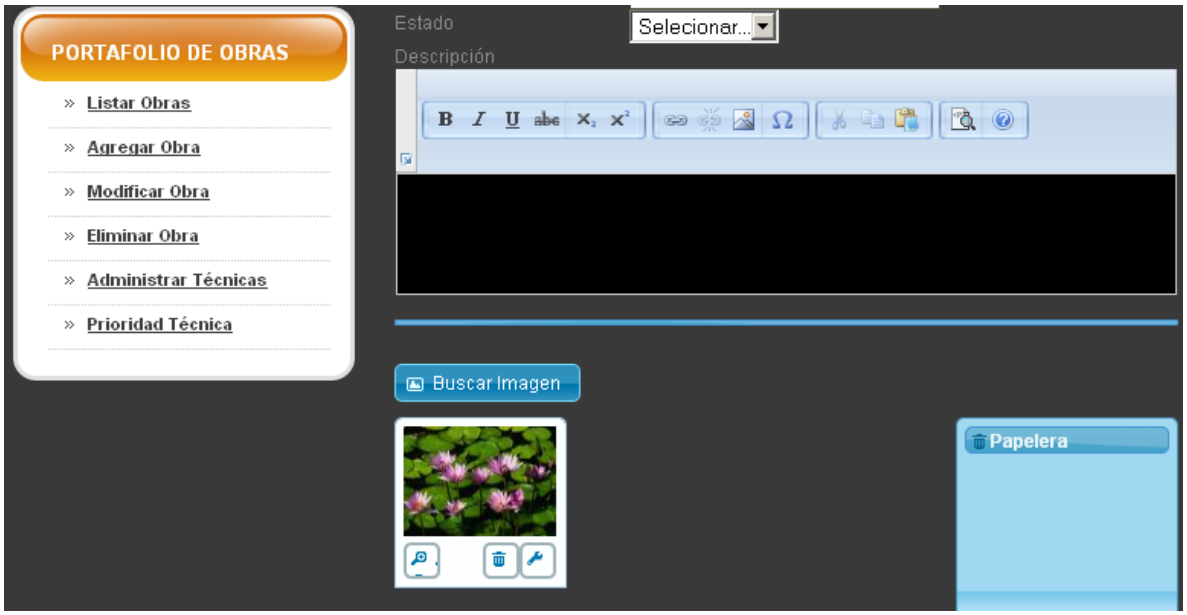
Gráfica B.22 Agregar nueva obra

Aparte de los datos correspondientes a las características de la obra, también está la opción de subir una o varias imágenes para publicar junto con la información; Al cliquear el botón de agregar imagen nos muestra el siguiente cuadro:



Gráfica B.23 Agregar imagen

Una vez se seleccionada la imagen a subir, observamos que esta se carga en la parte inferior del editor.



Gráfica B.24 Después de agregar imagen

Podemos observar que en la parte inferior de la imagen se muestran unas opciones que nos permiten ampliar la imagen o colocarla en la papelera (también se puede colocar en la papelera arrastrando la imagen hasta esta).

Cuando la imagen se ha colocado en la papelera nos muestra la opción de recuperar la imagen.

Modificar Obra Permite editar información e imágenes pertenecientes a una obra.

Gráfica B.25 Seleccionar obra para modificar

Eliminar Obra Permite eliminar obras que se habían subido anteriormente en el sistema.

Gráfica B.26 Seleccionar obra a eliminar

Administrar Técnicas Permite agregar, modificar, o eliminar las técnicas manejadas por el artista.

Gráfica B.27 Opciones para administrar las técnicas

Prioridad Técnica Permite establecer la prioridad de la técnica de acuerdo a las exigencias y prioridades del artista, solo basta con arrastrar (hacia arriba o abajo) la técnica a la posición que se desee.

Gráfica B.28 Establecer orden de técnica

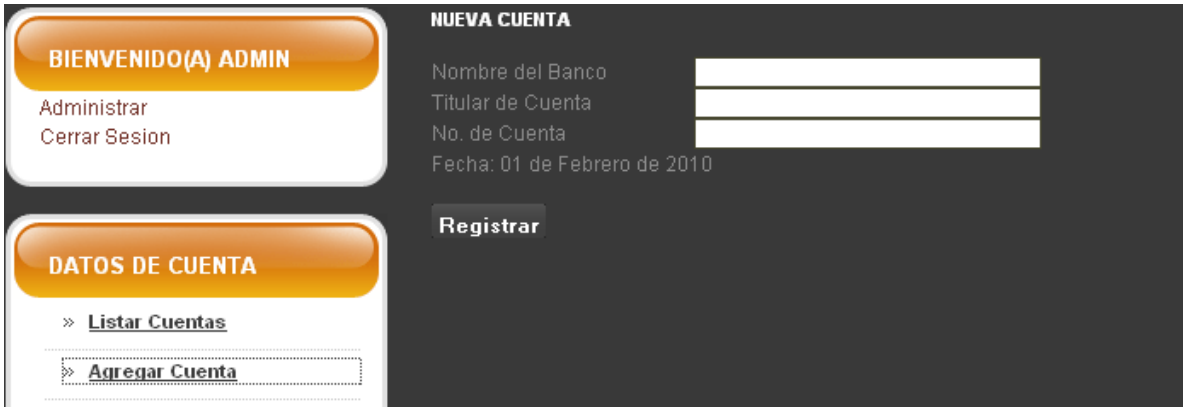
Cuentas Bancarias

Esta opción nos permite administrar las cuentas bancarias que se mostrarán al cliente al momento de comprar una obra.

Listar Cuentas permite mostrar una lista de las cuentas bancarias que han sido registradas por el administrador.

Gráfica B.29 Listar cuentas bancarias

Agregar cuenta permite al administrador agregar la información de una nueva cuenta bancaria.



Gráfica B.30 Formulario para agregar nueva cuenta

Modificar cuenta lista las cuentas bancarias actuales para seleccionar alguna y proceder a modificar su información.

Gráfica B.31 Modificar cuenta

Eliminar cuenta de igual manera nos lista las cuentas existentes pero con la opción de eliminar la cuenta que sea seleccionada.

Gráfica B.32 Eliminar cuenta

Items (orden de compra)

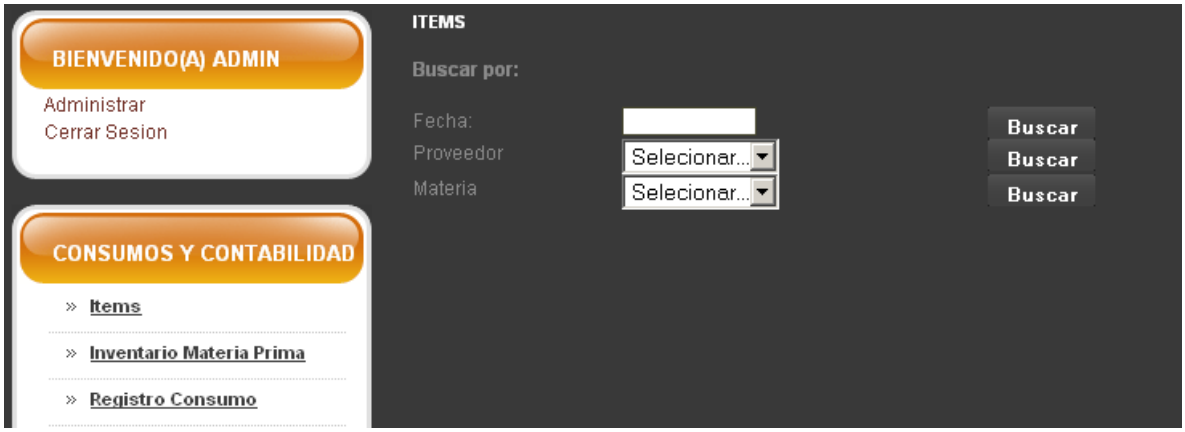
Este menú permite agregar información de materia prima o insumos comprados para tener un control de las entradas y del inventario.

Los ítems se pueden agregar o consultar por fecha, proveedor o material.

Agregar Items nos muestra el formulario para agregar un nuevo ítems y también tiene el enlace para los formularios de proveedor o material los cuales se utilizarían en caso de no existir en la base de datos un proveedor o material a ingresar en la orden de compra.

Gráfica B.33 Agregar nuevo ítems

Consultar Items nos permite buscar un ítem ya sea por fecha, proveedor o material.



Gráfica B.34 Consultar ítems

Al seleccionar una de las opciones y cliquear el botón **Buscar** se mostrará en pantalla todos los registros que concuerden con la opción.

Gráfica B.35 Resultado de **c**onsultar ítems por proveedor

Registro de Consumo

En esta opción del menú se ingresan los datos pertinentes que indiquen la cantidad y el material utilizado por una obra específica.

Gráfica B.36 Formulario de registro de consumo por obra

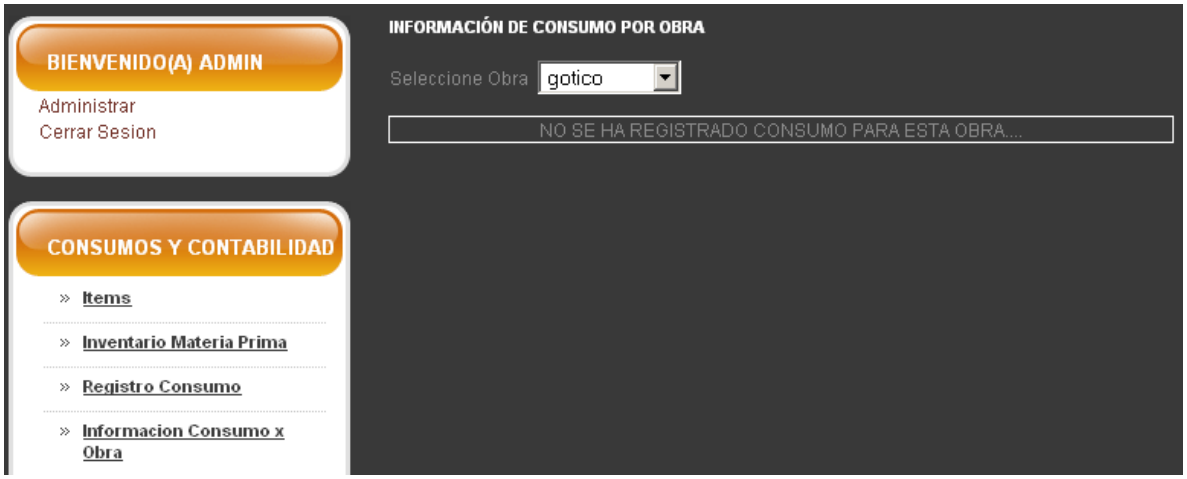
Información de consumo por obra nos permite seleccionar una obra para observar los datos de los materiales utilizados esta.

Primeramente se tiene que seleccionar una obra para que se cargue toda la información pertinente al consumo de material durante su elaboración.

Gráfica B.37 Seleccionar obra para cargar información de consumo

Una vez se ha seleccionado una obra se muestra en pantalla la información, como se puede ver en la Gráfica B38.

Gráfica B.38 Información de consumo por obra



Gráfica B.39 Mensaje mostrado en caso de no existir registro de consumo

ANEXO C: CÓDIGO FUENTE

Función generación de menús aleatorios para la parte administrativa e index

```
/**
 * Función encargada de cargar los menús de la página administrativa
  * @param String $sObjetivo
 * @return xajaxResponse
 */
 function fnModificarMenuModulo($sObjetivo){
    // Dependiendo de la selección en el menú principal escogemos el menú de opciones.
    // Hce referencia a los modulos
    switch($sObjetivo){
        case "centroeducativo":
            // Abrimos el archivo del cual vamos a leer el texto(o HTML).
            $archivo = @fopen("modules/GestionCentrosEducativos/menus/menus_CentroEducativo.php", "r");
            //$contenido_mainbody = @fopen("formularios/texto_inicio_inicio.html", "r");
            break;
        default:
            $archivo = false;
    }
    // Con esta variable se imprimirá en el div.
    $Formulario = "";
    if($archivo) {
    while (!feof($archivo)) {
```

```
        // si no le paso el tamaño de línea el fgets, me lee hasta el EOL
        $Formulario .= fgets($archivo);
    }
    // cerrramos el archivo.
    fclose($archivo);
    }
    // declaramos un objeto tipo respuesta (para las request que vendran.)
    $xajaxRespuesta = new xajaxResponse();
    // le decimos que debe cambiar el div 'left'; en el html interno escribir
    // el contenido de la variable formulario.
    $xajaxRespuesta->assign("breadcrumb","innerHTML", $Formulario);
    //$xajaxRespuesta->jquery->fadeOut("#breadcrumb","fast");
    //$xajaxRespuesta->jquery->fadeIn("#breadcrumb","fast");
    $xajaxRespuesta->jquery->addClass("#".$sObjetivo,"active");
  // Y también el div 'mainbody' por el contenido adecuado para cada modulo.
    //$xajaxRespuesta->assign("main-content","innerHTML",$Contenido);
    // la asignación anterior es la que se va a devolver para aplicar los cambios.
    return $xajaxRespuesta;
}
$xajax->register(XAJAX_FUNCTION,"fnModificarMenuModulo");

 /**
 * Función encargada de cargar los menús de la página administrativa
* @param String $sObjetivo
 * @return xajaxResponse
```

```
 */
 function fnModificarSubMenuModulo($sObjetivo){
    // Variable que me va almacenar le nombre del menu
    $sNombreSubmenu = "";
    // Abrimos el archivo del cual vamos a leer el texto(o HTML).
    $archivo = @fopen("modules/menu/submenu_administracion.php", "r");
    $contenido = @fopen("modules/menu/contenido_administracion.php", "r");
    // dependiendo de la selección en el menú principal escogemos el menú de opciones.
    switch($sObjetivo){
        // Caso del modulo de la gestión de centros educativos
        case "perfil":
            $sNombreSubmenu = "<h3>Perfil</h3>";
            break;
        case "menuprincipal":
            $sNombreSubmenu = "<h3>Menu Principal</h3>";
            break;
        case "galeria":
            $sNombreSubmenu = "<h3>Portafolio de Obras</h3>";
            break;
        case "noticia":
            $sNombreSubmenu = "<h3>Noticias</h3>";
            break;
        case "consumo":
            $sNombreSubmenu = "<h3>Consumos y Contabilidad</h3>";
            break;
```

```
    case "pago":
        $sNombreSubmenu = "<h3>Datos de Cuenta</h3>";
        break;
    default:
        $archivo = false;
        $contenido = false;
}
// Con esta variable se imprimirá en el div.
$Formulario = "";
// Obtenemos la información del administrador
$objUsuario = new Usuario();
$objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
if($archivo) {
    $bPaso=false;
    while($sLine = fgets($archivo)){
        if(eregi("<!--".$sObjetivo."-->", $sLine)) $bPaso = true;
        elseif(eregi("<!--fin".$sObjetivo."-->", $sLine)) $bPaso = false;
        if($bPaso){
        if(eregi("<!--admin-->", $sLine)){
            if($objUsuario->getId()==1) $bPaso = true;
            else $bPaso = false;
        }}
        elseif(eregi("<!--finadmin-->", $sLine)) $bPaso = false;
        if($bPaso) $Formulario .= $sLine;
    }
```

```
    // cerrramos el archivo.
    fclose($archivo);
}
// Con esta variable se imprimirá en el div.
$ContenidoModulo = "";
if($contenido) {
      $bPaso=false;
      while($sLine = fgets($contenido)){
        if(eregi("<!--".$sObjetivo."-->", $sLine)) $bPaso = true;
        elseif(eregi("<!--fin".$sObjetivo."-->", $sLine)) $bPaso = false;
        if($bPaso) $ContenidoModulo .= $sLine;
    }
    // cerramos el archivo.
    fclose($contenido);
}
// Declaramos un objeto tipo respuesta (para las request que vendrán.)
$objResponse = new xajaxResponse();
$objResponse->assign("nombresubmenu", "innerHTML", $sNombreSubmenu);
// Le decimos que debe cambiar el div 'left'; en el html interno escribir
// El contenido de la variable formulario.
$objResponse->assign("submenu","innerHTML", $Formulario);
$objResponse->assign("contenido","innerHTML", $ContenidoModulo);
// Y tambien el div 'mainbody' por el contenid adecuado para cada modulo.
//$xajaxRespuesta->assign("main-content","innerHTML",$Contenido);
// la asignacion anterior es la que se va a devolver para aplicar los cambios.
```

```
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnModificarSubMenuModulo");
```

Funciones para la modificación de los contenidos de los diversos links

```
/**
 * Función encargada de Mostrar todos los servicios registrados en la Base de Datos
 * @return HTTPReponse
 */
function fnMostrarEmpresa($iId=null){
        require_once("modules/empresa/_empresa.php");
        if(count($aRegistros)>0){
         if(null == $iId){
                $salidaHTML .= "<table class='contentpaneopen'>";
                $salidaHTML .= "<tr><th colspan='2'><b>Menu Principal</b></th></tr>";
                foreach($aRegistros as $aRegistro){
                        $salidaHTML .= "<tr>";
                        $salidaHTML .= "<td width='4%'><a href='javascript:void(0)'
onclick='xajax_fnMostrarEmpresa($aRegistro[0])'><img
src=\"images/con_info.png\"></a>";
                        $salidaHTML .= "<td>$aRegistro[1]</td>";
                        $salidaHTML .= "</tr>";
                }
                $salidaHTML .= "</table>";
         }
         else{
```

```
                $objInformacion->__construct($iId);

                $salidaHTML .= "<h2>Informaci&oacute;n del Servicio</h2>";

                $salidaHTML .= "<span class='small'>Modificado por ".$objInformacion->getUsuario()->getNombre()."</span><br /><br />";

                $salidaHTML .= "<table class='contentpaneopen'>";

                $salidaHTML .= "<tr><td colspan='2'>Nombre Menu Principal</td></tr>";

                $salidaHTML .= "<tr><td width='5%'></td><td>".$objInformacion->getNombre()."</td></tr>";

                $salidaHTML .= "<tr><td colspan='2'>Contenido</td></tr>";

                $salidaHTML .= "<tr><td width='5%'></td><td>".$objInformacion->getContenido()."</td></tr>";

                $salidaHTML .= "</table>";

                $salidaHTML .= "<br /><br /><input type='button' class='button' value='Regresar' onclick=\"xajax_fnMostrarEmpresa()\";'/>";

            }


        }

        $salidaHTML .= "</div>";

        // Declaramos un objeto tipo respuesta (para las requeste que vendrán.)

    $objResponse = new xajaxResponse();

    // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    // la asignación anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnMostrarEmpresa");
```

```
/**
 * Función encargada de seleccionar un menú principal a modificar o eliminar en la Base de Datos
 * @return HTTPReponse
 * @param int $iTipoAccion - Accion a ejecutar 2=Modificar; 3=Eliminar
 */
function fnSeleccionarEmpresa($iTipoAccion){
    require_once("modules/empresa/_empresa.php");
    if(count($aRegistros)>0){
        $salidaHTML  = "<div class='contentheading'>Seleccionar Menu</div>";
        $salidaHTML .= "<div> </div>";
        $salidaHTML .= "<form id='form' onsubmit='return false;'>";
        $salidaHTML .= "<table class='contentpaneopen'>";
        $salidaHTML .= "<tr><th width='15%' style='text-align:center;'><b>Selecionar</b></td><th><b>Nombre Menu</b></td></tr>";

        $iContador = 0;
        foreach ($aRegistros as $aRegistro) {
            $salidaHTML .= "<tr>";
            $salidaHTML .= "<td><div align='center'><input type='radio' name='id' id='id$iContador' value='$aRegistro[0]'></div></td>";
            $salidaHTML .= "<td><label for='id$iContador' style='font-size:14px;cursor:pointer'>$aRegistro[1]</label></td>";
            $salidaHTML .= "</tr>";
            $iContador++;
        }
```

```
// Seleccionamos que acción se va a realizar cuando se de click en enviar
// 2 = Modificar Director
// 3 = Eliminar Director
if ($iTipoAccion == 2) {
    $tipo_funcion = "fnModificarEmpresa";
    $value_boton  = "Modificar";
}

else {
    $tipo_funcion = "fnEliminarEmpresa";
    $value_boton  = "Eliminar";
}

$salidaHTML .= "</table>";
    $salidaHTML .= "<input type='hidden' name='tipo_accion' value='$iTipoAccion' />";
    $salidaHTML .= "<input type='radio' disabled='true' name='id' value='0' checked
style='visibility:hidden'>";
    $salidaHTML .= "<br><div id='validacion_selector'></div><br>";
    $salidaHTML .= "<input type='button' class='button' value='$value_boton'
onclick='xajax_$tipo_funcion(xajax.getFormValues(\"form\")); return false;'>";
    $salidaHTML .= "</form>";
  }
  $salidaHTML .= "</div>";
  // declaramos un objeto tipo respuesta (para las requeste que vendran.)
  $objResponse = new xajaxResponse();
```

```
        // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML
        $objResponse->assign("contenido", "innerHTML", $salidaHTML);
        // la asignacion anterior es la que se va a devolver para aplicar los cambios.
        return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnSeleccionarEmpresa");

/**
 * Función encargada de cargar el formulario de modificar el contenido del menú principal en la Base de Datos
 * @return HTTPReponse
 * @param array $aFormulario - Parametros enviados por el formulario
 */
function fnModificarEmpresa($aFormulario){
        // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
        if($aFormulario['id']==0){
         $salidaHTML = "<font color='red'>Debes selecionar un menu.</font>";
         $objResponse->assign("validacion_selector", "innerHTML", $salidaHTML);
        }
        else{
         // Declaramos los objetos respectivos
         $objGenerico = new Generico();
         $objInformacion = new Informacion($aFormulario['id']);
         $objUsuario = new Usuario();
```

```
        // Eliminamos la cache que probablemente se haya generado

        $objResponse->jquery->remove("iframe");

            //$objResponse->jquery->remove("menu_frame");


        $salidaHTML  = "<div class='contentheading'>Modificar Menu Principal<br
/></div><br />";

        $salidaHTML .= "<div> </div>";

        $salidaHTML .= "<div id='subcontenido'>";

        $salidaHTML .= "<form id='form' onsubmit='return false;'>";

        $salidaHTML .= "<div class='module-form'>";

        $salidaHTML .= "<table class='contentpaneopen'>";

        $salidaHTML .= "<tr><td width='30%'>Nombre Menu</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='nombre' name='nombre'
value='".$objInformacion->getNombre()."' size='30' disabled='disabled' /></td><td><div
id='validar_nombre'></div></td></tr>";

        $salidaHTML .= "<tr><td colspan='4'>Contenido Menu<span
id='validar_menu'></span>";

        $salidaHTML .= "<div id='menu_frame'>".$objGenerico-
>fnCargarEditor($objInformacion->getContenido(),3,"menu")."</div></td></tr>";

        $salidaHTML .= "</table>";

        $salidaHTML .= "</div>";

        $salidaHTML .= "<p> </p>";

        //$salidaHTML .= "<input type='button' class='button' value='Registrar'
onclick='xajax_fnAccionBDProyectoPedagogico(xajax.getFormValues(\"form\"),1);'/>";

        $salidaHTML .= "<input type='button' class='button' value='Actualizar' onclick=\"var
aFCKeditorXML=new Array();";

                $salidaHTML .=
"aFCKeditorXML['menu']=FCKeditorAPI.GetInstance('menu').GetXHTML();";
```

```
        $salidaHTML .= "xajax_fnAccionBDEmpresa(xajax.getFormValues('form'),2,aFCKeditorXML); return false;\">";

            $salidaHTML .= "<input type='hidden' id='id' name='id' value='".$aFormulario['id']."' />";

            $salidaHTML .= "</form>";

            $salidaHTML .= "</div>";

            // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML

           $objResponse->assign("contenido", "innerHTML", $salidaHTML);

           }

    // la asignación anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnModificarEmpresa");


/**

 * Función encargada de ejecutar las acciones de Agregar, Modificar o Eliminar el contenido del menú principal en la Base de Datos

 * @return HTTPReponse

 * @param array $aFormulario - Parametros enviados por el formulario

 * @param int $iTipoAccion - Accion a ejecutar en la base de datos 1=Agregar; 2=Modificar; 3=Eliminar

 */

function fnAccionBDEmpresa($aFormulario, $iTipoAccion,$aFCKEditor = null){

           // Declaramos los objetos respectivos

           $objGenerico = new Generico();

           $objInformacion = new Informacion();
```

```
        $objUsuario  = new Usuario();
        // Declaramos un objeto tipo respuesta (para las requeste que vendrán.
    $objResponse = new xajaxResponse();
        // Variable de control
        $bValidacion = array();
        $bValidacion[] = true;
        if(($iTipoAccion == 1)or($iTipoAccion == 2)){
         // Rescatamos las variables enviadas por el formulario
         //$sNombre              = $objGenerico->Caracter(trim($aFormulario['nombre']));
         $sContenido           = $aFCKEditor['menu'];

         // Validaremos cada una de las respectivas variables
         // Validar el nombre
         //$bValidacion[] = $objResponse->validation-
>text($sNombre,"nombre","validar_nombre");
         $bValidacion[] = $objResponse->validation-
>empt($sContenido,"menu_frame","validar_menu");
        }
        if(!in_array(false,$bValidacion)){
         $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
         // Ejecutamos las respectivas acciones con la base de datos
         switch($iTipoAccion){
                case 1:
            // Asignamos los valores al objeto Servicio
                        $objInformacion->setNombre($sNombre);
                        $objInformacion->setContenido($sDescripcion);
```

```
                $objInformacion->setUsuario($objUsuario->getId());
                // Registramos el Servicio
                $bRegistro = $objInformacion->registrarInformacion();

                $salidaHTMLtemp = "<p>Menu Principal registrado
correctamente.</p>";
                break;
            case 2:
                // Incializamos el objeto Proyecto Pedagogico
                $objInformacion->__construct($aFormulario['id']);
                // Asignamos los valores al objeto Proyecto Pedagógico
                //$objEmpresa->setNombre($sNombre);
                $objInformacion->setContenido($sContenido);
                //$objInformacion->setUsuario($objUsuario->getId());
               $objInformacion->setUsuario(1);
                // Actualizamos el Servicio
                $bRegistro = $objInformacion->actualizarInformacion();

                $salidaHTMLtemp = "<p>Menu Principal modificado
correctamente.</p>";
                break;
            case 3:
        // Iniciamos el objeto Director
                $objInformacion->__construct($aFormulario['id']);
                // Eliminamos el director
                $bRegistro = $objInformacion->eliminarInformacion();
```

```
                        $salidaHTMLtemp = "<p>Menu Principal eliminado
correctamente.</p>";

                        break;

            }

            // Si fallo la ejecución del comando imprimimos un error.

        if (!$bRegistro) $salidaHTML .= "Error. Imposible realizar la operación.<br><br>";

            // Si no notificamos que todo fue un éxito

        else $salidaHTML .= $salidaHTMLtemp;


            $salidaHTML .= "<input type='button' class='button' value='Aceptar'
onclick=\"xajax_fnMostrarEmpresa()\";'/>";

            $salidaHTML .= "</div>";

            // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

            $objResponse->assign("subcontenido", "innerHTML", $salidaHTML);

           }

    // la asignación anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnAccionBDEmpresa");
```

Librería con las funciones para cargar el editor de contenido FCKeditor

```
/**

        * @desc Función encargada de cargar el editor FckEditor

        * @param String Cadena con el valor a cargar dentro del editor

        * @param int Tipo de editor a cargar
```

```
        * @param String Nombre del editor
        */
        public function
fnCargarEditor($sContenido,$iTipo=null,$sNombre=null,$sPathArchivos=null){
         include_once("FCKeditor/fckeditor.php");
         if(null !== $sNombre){$oFCKeditor = new FCKeditor($sNombre);}
         else{$oFCKeditor = new FCKeditor('FCKeditor1');}
         $oFCKeditor->BasePath='FCKeditor/';
         if(null !== $iTipo){
                  // Vamos a definir los diferentes tipos
                  switch($iTipo){
                           // Caso para insertar el himno
                           case 1:
                                    $oFCKeditor->Height='100%';
                                    $oFCKeditor->ToolbarSet='Titulo';
                                    break;
                           case 2:
                                    $oFCKeditor->Height='260';
                                    $oFCKeditor->ToolbarSet='Contenido';
                                    break;
                           case 3:
                                    $oFCKeditor->Height='400';
                                    $oFCKeditor->ToolbarSet='Completo';
                                    break;
                         case 4:
                           $oFCKeditor->Height='150';
```

```
                    $oFCKeditor->ToolbarSet='Noticia';
                    break;
                }
        }
        $oFCKeditor->Value=$sContenido;
            if(null !== $sPathArchivos) $_SESSION['ubicacionFCK'] = $sPathArchivos;
            else $_SESSION['ubicacionFCK'] = '/files/';
        return $oFCKeditor->CreateHTML();
       }
```

Funciones de administración de sesión para los usuarios

```
function fnFormularioRegistro(){
    $objResponse = new xajaxResponse();

    $salidaHTML  = "<div class='contentheading'>Registro Usuario</div>";
    $salidaHTML .= "<div> </div>";
    $salidaHTML .= "<div id='subcontenido'>";
    $salidaHTML .= "<form id='form_registro' onsubmit='return false;'>";
    $salidaHTML .= "<div class='module-form'>";
    $salidaHTML .= "<table class='contentpaneopen'>";
    $salidaHTML .= "<tr><td colspan='3'><div style='font-size:110%;font-
weight:bold;color:white'>Datos Administrativos</div></td></tr>";
    $salidaHTML .= "<tr><td width='30%'>Login</td><td width='35%' colspan='2'><input
type='text' class='inputbox' id='login_registro' name='login_registro' value='' size='30'
/> <span style='color:red'>*</span></td><td><div
id='validar_login_registro'></div></td></tr>";
```

```
    $salidaHTML .= "<tr><td width='30%'>Email</td><td width='35%' colspan='2'><input
type='text' class='inputbox' id='email' name='email' value='' size='30' /> <span
style='color:red'>*</span></td><td><div id='validar_email'></div></td></tr>";

    $salidaHTML .= "<tr><td
colspan='3'>_____<br /><br
/></td></tr>";

    $salidaHTML .= "<tr><td colspan='3'><div style='font-size:110%;font-
weight:bold;color:white'>Datos Personales</div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Nombre</td><td width='35%' colspan='2'><input
type='text' class='inputbox' id='nombre' name='nombre' value='' size='30' /> <span
style='color:red'>*</span></td><td><div id='validar_nombre'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Apellidos</td><td width='35%' colspan='2'><input
type='text' class='inputbox' id='apellido' name='apellido' value='' size='30' /> <span
style='color:red'>*</span></td><td><div id='validar_apellido'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Direcci&oacute;n</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='direccion' name='direccion' value=''
size='30' /></td><td><div id='validar_direccion'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Fecha Nacimiento</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='fecha_nacimiento'
name='fecha_nacimiento' value='' size='10' /> <span
style='color:red'>*</span></td><td><div id='validar_fecha_nacimiento'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Movil</td><td width='35%' colspan='2'><input
type='text' class='inputbox' id='movil' name='movil' value='' size='10' maxlength='10'
/></td><td><div id='validar_movil'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Tel&eacute;fono</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='telefono' name='telefono' value=''
size='7' maxlength='7' /></td><td><div id='validar_telefono'></div></td></tr>";

    $salidaHTML .= "</table>";

    $salidaHTML .= "</div>";

    $salidaHTML .= "<p> </p>";

    $salidaHTML .= "<input type='button' class='button' value='Registrar'
onclick=\"xajax_fnAccionDBUsuario(xajax.getFormValues('form_registro'),1); return
false;\">";
```

```
    $salidaHTML .= "</form>";

    $salidaHTML .= "<br />Los campos <font color='red'>*</font> son obligatorios.";

    $salidaHTML .= "</div>";


    // Eliminamos la cache generada por el jQuery

    //$objResponse->jquery->each("body>:not(.tail)","function(){\$(this).remove();}");

    //$objResponse->jquery->each("body>:not(#tail)","function(){\$(this).remove();}");

    // Le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    // Calendario

    // Opciones Adicionales

    // minDate: -20, maxDate: '+1M +10D' ,showButtonPanel: true

    $objResponse->jquery->datepicker("#fecha_nacimiento","{changeMonth:
true,changeYear: true, maxDate:'+0D', showButtonPanel: true, yearRange:
'1915:2000',prevText: 'Anterior',nextText: 'Siguiente'}");

    // la asignación anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION, "fnFormularioRegistro");


function fnComprovarLogin($aFormulario){

    $objUsuario=new Usuario();

        // Declaramos un objeto tipo respuesta (para las request que vendrán.)

    $objResponse = new xajaxResponse();

    // Obtenemos los datos enviados por el formulario

    $sLogin = $aFormulario['login'];
```

```
    $sPasswd = $aFormulario['passwd'];

    // Variable de control

    $bValidacion = array();

    // Validamos los campos enviados

    $bValidacion[] = $objResponse->validation->text($sLogin,"login","validar_login");

    $bValidacion[] = $objResponse->validation-
>text($sPasswd,"passwd","validar_passwd");

    $sPasswd = md5($sPasswd);

    //Iniciaremos la sección

    if(!in_array(false,$bValidacion)){

        $bValidacion = true;

        if(!$objUsuario->getUsuario("LOGIN='".$sLogin."'")){

            $objResponse->alert($objUsuario->getLogin()." - ".$sLogin);

            $outHTML = "<a id='tt_login' title='Nombre de usuario no existe'><img
src='images/exclamation.gif'></a>";

            $objResponse->assign("validar_login", "innerHTML", $outHTML);

            $objResponse->assign("login", "style.border", "1px solid #FF0000");

            $objResponse->jquery->tooltip("#tt_login","{cssClass:'tooltip-blue',opacity:10}");

            $bValidacion = false;

        }

          elseif(!$objUsuario->getUsuario("LOGIN='".$sLogin."' AND
CLAVE='".$sPasswd."'")){

            if(!$objUsuario->getUsuario("LOGIN='".$sLogin."' AND 2CLAVE='".$sPasswd."'")){

                $objResponse->assign("login", "style.border", "1px solid #b0a88f");

                $objResponse->assign("validar_login", "innerHTML", "");

                $outHTML = "<a id='tt_passwd' title='Contrase&ntilde;a incorrecta'><img
src='images/exclamation.gif'></a>";
```

```
            $objResponse->assign("validar_passwd", "innerHTML", $outHTML);
            $objResponse->assign("passwd", "style.border", "1px solid #FF0000");
            $objResponse->jquery->tooltip("#tt_passwd","{cssClass:'tooltip-
blue',opacity:10}");
            $bValidacion = false;
        }
        // Actualizamos la contraseña
        else{
            $objUsuario->setClave($objUsuario->get2Clave());
            $objUsuario->set2Clave("");
            $objUsuario->actualizarUsuario();
        }
    }
    // Iniciamos la sesión y redirigimos a la pagina de administración
    if($bValidacion){
        $_SESSION['usuario'] = $sLogin;
        $objResponse->redirect("administracion.php",0);
    }
  }
  // la asignación anterior es la que se va a devolver para aplicar los cambios.
  return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnComprovarLogin");

function fnCerrarSession(){
  // Eliminamos las variables tipo sección que habíamos creado
```

```
    unset($_SESSION['usuario']);
    // Declarmaos nuestro objeto HTTResponse
    $objResponse = new xajaxResponse();
    // Recarmos la apagina de inicio
    $objResponse->redirect("index.php",0);
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnCerrarSession");

/**
 * Función encargada de cargar el formulario para modificar la contraseña
 * @return HTTPResponse
 */
function fnContrasenaPerdida(){
        $salidaHTML  = "<div class='contentheading'>Recuperar contrase&ntilde;a</div>";
        $salidaHTML .= "<div> </div>";
        $salidaHTML .= "<div class='box-indent'>";
        $salidaHTML .= "<div class='width' id='confirmacion'>";
        $salidaHTML .= "<form id='recuperar_contrasena' onsubmit='return false;'>";
        $salidaHTML .= "<label for='correo'>Correo</label>";
        $salidaHTML .= "<input type='text' name='correo' id='correo' class='inputbox'
value='' /> <span id='validar_correo'></span>";
        //$salidaHTML .= "<div class='clr'>";
       $salidaHTML .= "<p> </p>";
```

```
        $salidaHTML .= "<input class='button' value='Recuperar' type='button' onclick=\"xajax_fnValidarCambioLoginContrasena(xajax.getFormValues('recuperar_contrasena'),2); return false;\">   ";

        $salidaHTML .= "<input class='button' value='Cancelar' type='button' onclick=\"xajax_fnMostrarMenuEmpresa()\">";

        //$salidaHTML .= "</div>";

        $salidaHTML .= "</form>";

        $salidaHTML .= "</div>";

        $salidaHTML .= "</div>";

    // Declaramos un objeto tipo respuesta (para las request que vendrán.)

    $objResponse=new xajaxResponse();

    // le decimos que debe cambiar el div 'mainbody' y le el html interno escribir

    // el contenido de la variable formulario.

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    // la asignación anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnContrasenaPerdida");


/**

 * Función encargada de cargar el formulario para modificar el Login del usuario

 * @return HTTPResponse

 */

function fnLoginPerdido(){

        $salidaHTML  = "<h2>Recuperar Nombre de Usuario</h2>";

        $salidaHTML .= "<div class='box-indent'>";

        $salidaHTML .= "<div class='width' id='confirmacion'>";
```

```
	$salidaHTML .= "<form id='form' onsubmit='return false;'>";

	$salidaHTML .= "<label for='correo'>Correo</label>";

	$salidaHTML .= "<input type='text' name='correo' id='correo' class='inputbox'
value='' /> <span id='validar_correo'></span>";

	$salidaHTML .= "<div class='clr'>";

	$salidaHTML .= "<input class='button' value='Recuperar' type='button'
onclick=\"xajax_fnValidarCambioLoginContrasena(xajax.getFormValues('form'),1); return
false;\">   ";

	$salidaHTML .= "<input class='button' value='Cancelar' type='button'
onclick=\"parent.location='login.php'\">";

	$salidaHTML .= "</div>";

	$salidaHTML .= "</form>";

	$salidaHTML .= "</div>";

	$salidaHTML .= "</div>";

	// Declaramos un objeto tipo respuesta (para las request que vendrán.)

	$objResponse=new xajaxResponse();

	// le decimos que debe cambiar el div 'mainbody' y le el html interno escribir

	// el contenido de la variable formulario.

	$objResponse->assign("contenido", "innerHTML", $salidaHTML);

	// la asignacion anterior es la que se va a devolver para aplicar los cambios.

	return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnLoginPerdido");


/**
```

*** Funcion encargada de realizar las respectivas validaciones para le cambio de la contraseña o el recordatorio**

```
 * del Nombre de Usuario
 * @return HTTPResponse
 * @param array $aFormValues
 * @param int $iAccion
 */
function fnValidarCambioLoginContrasena($aFormValues,$iAccion) {
	$objUsuario = new Usuario();
	$aRegistros = $objUsuario->getUsuarioAll();
	// declaramos un objeto tipo respuesta (para las request que vendran.)
	$objResponse=new xajaxResponse();
	$sCorreo = trim($aFormValues['correo']);
	// Validamos el correo
	$bValidacionCorreo = $objResponse->validation-
>email($sCorreo,"correo","validar_correo");
	if($bValidacionCorreo){
	 $bValidacionCorreo = false;
	 foreach($aRegistros as $aRegistro){
		if($aRegistro[5]==$sCorreo)
			$bValidacionCorreo = true;
	 }
	 if($bValidacionCorreo){
		$objCorreo = new Correo();
		if($iAccion == 1){
			$objUsuario->getUsuario("CORREO=\"".$sCorreo."\"");
			$sTituloMensaje = "Recuperar Nombre de Usuario";
```

```
                $sContenido  = "Coordial Saludo:<br />Ud. ha solicitado un
recordatorio para el nombre de usuario del sitio web";

                $sContenido .= "de la empresa Intelmec Ltda.<br />";

                $sContenido .= "El Nombre de Usuario es: <b>".$objUsuario-
>getLogin()."</b>.<p> </p>";

        }

        elseif($iAccion == 2){

                $sTituloMensaje = "Recuperar Contraseña";

                $sNuevaContrasena = fnGenerarClaveAleatorio(8);

                $sContenido  = "Coordial Saludo:<br />Ud. ha solicitado un cambio
en la contrase&ntilde;a para el sitio web";

                $sContenido .= "de la empresa Intelmec Ltda.<br />";

                $sContenido .= "Su nueva Contrase&ntilde;a es:
<b>".$sNuevaContrasena."</b><br />Por favor ingrese y modifiquela.<p> </p>";

                $sContenido .= "PD: Si ud. no ha solicitado el cambio de
contrase&ntilde;a por favor haga caso omiso de este mensaje.";

              $objResponse->alert($sContenido);

        }

        $sEnvio = $objCorreo->EnviarCorreo("no-
reply@galeria.com",$sCorreo,$sTituloMensaje,$sContenido);

        if($sEnvio!="ok"){

                $salidaHTML  = "Fallo al enviar el correo a las siguiente
direci&oacute;n <b>$sCorreo</b>,";

                $salidaHTML .= "el error generado fue el siguiente: ".$sEnvio.".<br
/>";

        }

        else{

                $salidaHTML  = "Correo enviado satisfactoriamente a la siguiente
direcci&oacute;n <b>$sCorreo</b><br />";
```

```
                    if($iAccion == 1) $salidaHTML .= "Su Nombre de Usuario se ha
enviado adjunto al correo mencionado.";

                    if($iAccion == 2) $salidaHTML .= "Su nueva Contrase&ntilde;a se ha
enviado adjunta al correo mencionado.";

                    $objUsuario->getUsuario("EMAIL='$sCorreo'");

                    $objUsuario->set2Clave(md5($sNuevaContrasena));

                    $objUsuario->actualizarUsuario();

                }

                $salidaHTML .= "<div class='clr'>";

                $salidaHTML .= "<input class='button' value='Regresar' type='button'
onclick=\"xajax_fnMostrarMenuEmpresa()\">";

                $salidaHTML .= "</div>";

                $objResponse->assign("confirmacion", "innerHTML", $salidaHTML);

            }

            else{

                $outHTML = "<a id='tt_correo' title='No se encuentra el correo
registrado'><img src='images/exclamation.gif'></a>";

                $objResponse->assign("validar_correo", "innerHTML", $outHTML);

                $objResponse->assign("correo", "style.border", "1px solid #FF0000");

                $objResponse->jquery->tooltip("#tt_correo","{cssClass:'tooltip-
blue',opacity:10}");

            }

        }

        // la asignacion anterior es la que se va a devolver para aplicar los cambios.

        return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnValidarCambioLoginContrasena");
```

```
function fnMostrarUsuarios(){
        $objUsuario = new Usuario();
        $objFechas = new Fechas();
        $aRegistros = $objUsuario->getUsuarioAll("ID>1");
        // declaramos un objeto tipo respuesta (para las request que vendran.)
        $objResponse=new xajaxResponse();
        $salidaHTML  = "<h2>Usuarios Registrados</h2>";
        $salidaHTML .= "<div class='box-indent'>";
        $salidaHTML .= "<table width='100%'>";
   $salidaHTML .= "<tr><td colspan='2'><div align='center'>----------------------------------------
----------</div></td></tr>";
        foreach($aRegistros as $aRegistro){
         $dFecha = $objFechas->FormatoFecha($aRegistro[3]);
         $salidaHTML .= "<tr>";
         $salidaHTML .= "<td>Nombre:</td><td>".$aRegistro[1]." ".$aRegistro[2]."</td>";
       $salidaHTML .= "</tr>";
       $salidaHTML .= "<tr>";
          $salidaHTML .= "<td>Fecha Nacimiento: </td><td>".$dFecha['d']." de
".$dFecha['M']." de ".$dFecha['Y']."</td>";
       $salidaHTML .= "</tr>";
       $salidaHTML .= "<tr>";
          $salidaHTML .= "<td>Tel&eacute;fono: </td><td>".$aRegistro[6]."</td>";
          $salidaHTML .= "</tr>";
       $salidaHTML .= "<tr>";
          $salidaHTML .= "<td>Celular: </td><td>".$aRegistro[7]."</td>";
```

```
            $salidaHTML .= "</tr>";
        $salidaHTML .= "<tr>";
            $salidaHTML .= "<td>Correo: </td><td>".$aRegistro[5]."</td>";
            $salidaHTML .= "</tr>";
        $salidaHTML .= "<tr><td colspan='2'><div align='center'>----------------------------------------</div></td></tr>";
          }
          $salidaHTML .= "</table>";
          $salidaHTML .= "</div>";
    $objResponse->assign("contenido", "innerHTML", $salidaHTML);
          // la asignacion anterior es la que se va a devolver para aplicar los cambios.
          return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnMostrarUsuarios");

/**

 * Funcion encargada de generar la clave aleatoria
 * @return string $clave
 * @param int $cantLetras
 */
function fnGenerarClaveAleatorio($cantLetras){
          $clave = "";
          for($i=0;$i<$cantLetras;$i++){
           //como el codigo puede tener numeros o letras en mayusculas o minusculas
           //elijo aleatoriamente una de las 3 opciones
```

```
            //si $nol vale 0 entonces es un numero
            //si $nol vale 1 entonces es una letra mayuscula
            //si $nol vale 2 entonces es una letra minuscula
            $nol = rand(0,2);
            switch($nol){
                    case 0: $clave .= chr(rand(48,57));break; //0-9
                    case 1: $clave .= chr(rand(65,90));break; //A-Z
                    case 2: $clave .= chr(rand(97,122));break; //a-z
            }
           }
           return $clave;
}

/**
 * Funcion encargada de mostrar la informacion del perfil
 * @return HTTPResponse
 */
function fnVerPerfil(){
           // Declaramos los objetos respectivos
           $objUsuario = new Usuario();
           $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
           $salidaHTML  = "<div class='contentheading'>Informaci&oacute;n Usuario</div>";
           $salidaHTML .= "<div> </div>";
           $salidaHTML .= "<table class='contentpaneopen'>";
           $salidaHTML .= "<tr><td width='15%'>Nombre</td><td><b>".$objUsuario-
>getNombre()." ".$objUsuario->getApellidos()."</b></td></tr>";
```

```
	$salidaHTML .= "<tr><td width='15%'>Direcci&oacute;n</td><td><b>".$objUsuario->getDireccion()."</b></td></tr>";

	$salidaHTML .= "<tr><td width='15%'>Tel&eacute;no</td><td><b>".$objUsuario->getTelefono()."</b></td></tr>";

	$salidaHTML .= "<tr><td width='15%'>Celular</td><td><b>".$objUsuario->getMovil()."</b></td></tr>";

	  $salidaHTML .= "<tr><td width='15%'>Correo</td><td><b>".$objUsuario->getEmail()."</b></td></tr>";

	  $salidaHTML .= "</table>";

	  // Declarmaos nuestro objeto HTTResponse

	  $objResponse = new xajaxResponse();

	  $objResponse->assign("contenido", "innerHTML", $salidaHTML);

	  // la asignacion anterior es la que se va a devolver para aplicar los cambios.

	  return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnVerPerfil");


/**

 * Funcion encargada de cargar el formulario para modificar el perfil

 * @return HTTPResponse

 */

function fnModificarPerfil(){

	  // Declaramos los objetos respectivos

	  $objUsuario = new Usuario();

	$objFechas   = new Fechas();

	$objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");

	  $salidaHTML  = "<div class='contentheading'>Informaci&oacute;n Usuario</div>";
```

```
$salidaHTML .= "<div> </div>";

$salidaHTML .= "<div id='subcontenido'>";

$salidaHTML .= "<form id='form' onsubmit='return false;'>";

$salidaHTML .= "<div class='module-form'>";

$salidaHTML .= "<table class='contentpaneopen'>";

$salidaHTML .= "<tr><td width='20%'>Nombre(s)</td><td><input type='text'
class='inputbox' id='nombre' name='nombre' value='".$objUsuario->getNombre()."'
size='30' /></td><td><div id='validar_nombre'></div></td></tr>";

$salidaHTML .= "<tr><td width='20%'>Apellido(s)</td><td><input type='text'
class='inputbox' id='apellido' name='apellido' value='".$objUsuario->getApellidos()."'
size='30' /></td><td><div id='validar_apellido'></div></td></tr>";

$sFecha = $objFechas->FormatoFecha($objUsuario->getFechaNacimiento());

$sFecha = $sFecha['Y']."-".$sFecha['m']."-".$sFecha['d'];

$salidaHTML .= "<tr><td width='20%'>Fecha Nacimiento</td><td><input type='text'
class='inputbox' id='fecha_nacimiento' name='fecha_nacimiento' value='".$sFecha."'
size='10' /></td><td><div id='validar_fecha_nacimiento'></div></td></tr>";

$salidaHTML .= "<tr><td width='20%'>Direcci&oacute;n</td><td><input type='text'
class='inputbox' id='direccion' name='direccion' value='".$objUsuario->getDireccion()."'
size='40' /></td><td><div id='validar_direccion'></div></td></tr>";

$salidaHTML .= "<tr><td width='20%'>Tel&eacute;fono</td><td><input type='text'
class='inputbox' id='telefono' name='telefono' value='".$objUsuario->getTelefono()."'
size='7' /></td><td><div id='validar_telefono'></div></td></tr>";

$salidaHTML .= "<tr><td width='20%'>Celular</td><td><input type='text'
class='inputbox' id='movil' name='movil' value='".$objUsuario->getMovil()."' size='11'
/></td><td><div id='validar_movil'></div></td></tr>";

$salidaHTML .= "<tr><td width='20%'>Correo</td><td><input type='text'
class='inputbox' id='email' name='email' value='".$objUsuario->getEmail()."' size='30'
/></td><td><div id='validar_email'></div></td></tr>";

$salidaHTML .= "</table>";

$salidaHTML .= "</div>";

$salidaHTML .= "<p> </p>";
```

```
        $salidaHTML .= "<input type='button' class='button' value='Actualizar'
onclick=\"xajax_fnAccionDBUsuario(xajax.getFormValues('form'),3); return false;\">";

        $salidaHTML .= "</form>";

        $salidaHTML .= "</div>";

        // Declarmaos nuestro objeto HTTResponse

        $objResponse = new xajaxResponse();

        $objResponse->assign("contenido", "innerHTML", $salidaHTML);

       // Calendario

        // Opciones Adicionales

        // minDate: -20, maxDate: '+1M +10D' ,showButtonPanel: true

        $objResponse->jquery->datepicker("#fecha_nacimiento","{changeMonth:
true,changeYear: true, maxDate:'+0D', showButtonPanel: true}");

        // la asignacion anterior es la que se va a devolver para aplicar los cambios.

        return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnModificarPerfil");


function fnModificarContrasena(){

        // Declaramos los objetos respectivos

        $objUsuario = new Usuario();

        $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");

        $salidaHTML  = "<div class='contentheading'>Informaci&oacute;n Usuario</div>";

        $salidaHTML .= "<div> </div>";

        $salidaHTML .= "<div id='subcontenido'>";

        $salidaHTML .= "<form id='form' onsubmit='return false;'>";

        $salidaHTML .= "<div class='module-form'>";
```

```
	$salidaHTML .= "<table class='contentpaneopen'>";

	$salidaHTML .= "<tr><td width='25%'>Contrase&ntilde;a actual</td><td
width='28%'><input type='password' class='inputbox' id='contrasena' name='contrasena'
value='' size='20' /></td><td><div id='validar_contrasena'></div></td></tr>";

	$salidaHTML .= "<tr><td width='25%'>Nuevo Contrase&ntilde;a</td><td
width='28%'><input type='password' class='inputbox' id='contrasena_nueva'
name='contrasena_nueva' value='' size='20' /></td><td><div
id='validar_contrasena_nueva'></div></td></tr>";

	$salidaHTML .= "<tr><td width='25%'>Repetir Contrase&ntilde;a</td><td
width='28%'><input type='password' class='inputbox' id='contrasena_repetir'
name='contrasena_repetir' value='' size='20' /></td><td><div
id='validar_contrasena_repetir'></div></td></tr>";

	$salidaHTML .= "</table>";

	$salidaHTML .= "</div>";

	$salidaHTML .= "<p> </p>";

	$salidaHTML .= "<input type='button' class='button' value='Actualizar'
onclick=\"xajax_fnAccionDBUsuario(xajax.getFormValues('form'),2); return false;\">";

	$salidaHTML .= "</form>";

	$salidaHTML .= "</div>";

	// Declarmaos nuestro objeto HTTResponse

	$objResponse = new xajaxResponse();

	$objResponse->assign("contenido", "innerHTML", $salidaHTML);

	// la asignacion anterior es la que se va a devolver para aplicar los cambios.

	return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnModificarContrasena");



/**
```

```
 * Funcion encargada de cominicarse con la tabla Usuario para actualizar el perfil
 * @return HTTPResponse
 * @param array $aFormValues
 */
function fnAccionDBUsuario($aFormValues,$iTipoAccion){
    // Declaramos los objetos respectivos
    $objGenerico = new Generico();
    $objUsuario  = new Usuario();
    $objCorreo = new Correo();
    $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
    // Declaramos un objeto tipo respuesta (para las requeste que vendran.
    $objResponse = new xajaxResponse();
    // Variable de control
    $bValidacion = array();
    $bValidacion[] = true;
    if($iTipoAccion == 1){
        // Rescatamos las variables enviadas por el formulario
        $sNombre = $objGenerico->Caracter(trim($aFormValues['nombre']));
        $sApellido = $objGenerico->Caracter(trim($aFormValues['apellido']));
        $sDireccion = $objGenerico->Caracter(trim($aFormValues['direccion']));
        $sFechaNacimiento = trim($aFormValues['fecha_nacimiento']);
        $sMovil = trim($aFormValues['movil']);
        $sTelefono = trim($aFormValues['telefono']);

        $sEmail = trim($aFormValues['email']);
```

```
        $sLogin = trim($aFormValues['login_registro']);


        // Validaremos cada una de las respectivas variables

        // Validar el nombre

        $bValidacion[] = $objResponse->validation-
>text($sNombre,"nombre","validar_nombre");

        $bValidacion[] = $objResponse->validation-
>text($sApellido,"apellido","validar_apellido");

        if($sDireccion != "")

            $bValidacion[] = $objResponse->validation-
>text($sDireccion,"direccion","validar_direccion");

        if($sMovil != "")

            $bValidacion[] = $objResponse->validation-
>number($sMovil,"movil","validar_movil");

        if($sTelefono != "")

            $bValidacion[] = $objResponse->validation-
>number($sTelefono,"telefono","validar_telefono");

        $bValidacion[] = $objResponse->validation-
>dat($sFechaNacimiento,"fecha_nacimiento","validar_fecha_nacimiento");

        //$bValidacion[] = $objResponse->validation->email($sEmail,"email","validar_email");

        //$bValidacion[] = $objResponse->validation->text($sLogin,"login","validar_login");


        $objUsuarioRegistrado =  new Usuario();

        $aRegistros = $objUsuarioRegistrado->getUsuarioAll();

        // Validamos que el correo electronico no se encuentre registrado

        if($objResponse->validation->email($sEmail,"email","validar_email")){

            foreach ($aRegistros as $aRegistro){

                if($aRegistro[5]==$sEmail){
```

```
                $outHTML = "<a id='tt_email' title='El correo electr&oacute;nico ya se encuentra registrado.'><img src='images/exclamation.gif'></a>";

                $objResponse->assign("validar_email", "innerHTML", $outHTML);

                $objResponse->assign("email", "style.border", "1px solid #FF0000");

                $objResponse->jquery->tooltip("#tt_email","{cssClass:'tooltip-blue',opacity:10}");

                $bValidation[] = false;

            }

            else{

                $objResponse->assign("email", "style.border", "1px solid #b0a88f");

                $objResponse->assign("validar_email", "innerHTML", "");

            }

        }

    }

    // Validamos que el login no se encuentre registrado

    if($objResponse->validation->text($sLogin,"login_registro","validar_login_registro")){

        foreach ($aRegistros as $aRegistro){

            if($aRegistro[8]==$sLogin){

                $outHTML = "<a id='tt_login_registro' title='El login ingresado ya se encuentra en uso.'><img src='images/exclamation.gif'></a>";

                $objResponse->assign("validar_login_registro", "innerHTML", $outHTML);

                $objResponse->assign("login_registro", "style.border", "1px solid #FF0000");

                $objResponse->jquery->tooltip("#tt_login_registro","{cssClass:'tooltip-blue',opacity:10}");

                $bValidation[] = false;

            }

            else{
```

```
                $objResponse->assign("login_registro", "style.border", "1px solid #b0a88f");
                $objResponse->assign("validar_login_registro", "innerHTML", "");
            }
        }
    }
}
if($iTipoAccion == 3){
    // Rescatamos las variables enviadas por el formulario
    $sNombre = $objGenerico->Caracter(trim($aFormValues['nombre']));
    $sApellido = $objGenerico->Caracter(trim($aFormValues['apellido']));
    $sDireccion = $objGenerico->Caracter(trim($aFormValues['direccion']));
    $sFechaNacimiento = trim($aFormValues['fecha_nacimiento']);
    $sMovil = trim($aFormValues['movil']);
    $sTelefono = trim($aFormValues['telefono']);

    $sEmail = trim($aFormValues['email']);

    // Validaremos cada una de las respectivas variables
    // Validar el nombre
    $bValidacion[] = $objResponse->validation-
>text($sNombre,"nombre","validar_nombre");
    $bValidacion[] = $objResponse->validation-
>text($sApellido,"apellido","validar_apellido");
    if($sDireccion != "")
        $bValidacion[] = $objResponse->validation-
>text($sDireccion,"direccion","validar_direccion");
    if($sMovil != "")
```

```
        $bValidacion[] = $objResponse->validation-
>number($sMovil,"movil","validar_movil");

    if($sTelefono != "")

        $bValidacion[] = $objResponse->validation-
>number($sTelefono,"telefono","validar_telefono");

    $bValidacion[] = $objResponse->validation-
>dat($sFechaNacimiento,"fecha_nacimiento","validar_fecha_nacimiento");

    //$bValidacion[] = $objResponse->validation->email($sEmail,"email","validar_email");

    //$bValidacion[] = $objResponse->validation->text($sLogin,"login","validar_login");


    $objUsuarioRegistrado =  new Usuario();

    $aRegistros = $objUsuarioRegistrado->getUsuarioAll("EMAIL!='".$sEmail."'");

    // Validamos que el correo electronico no se encuentre registrado

    if($objResponse->validation->email($sEmail,"email","validar_email")){

        foreach ($aRegistros as $aRegistro){

            if($aRegistro[5]==$sEmail){

                $outHTML = "<a id='tt_email' title='El correo electr&oacute;nico ya se
encuentra registrado.'><img src='images/exclamation.gif'></a>";

                $objResponse->assign("validar_email", "innerHTML", $outHTML);

                $objResponse->assign("email", "style.border", "1px solid #FF0000");

                $objResponse->jquery->tooltip("#tt_email","{cssClass:'tooltip-
blue',opacity:10}");

                $bValidation[] = false;

            }

            else{

                $objResponse->assign("email", "style.border", "1px solid #b0a88f");

                $objResponse->assign("validar_email", "innerHTML", "");
```

```
            }
          }
        }
    }
    if($iTipoAccion == 2){
        // Rescatamos las variables enviadas por el formulario
        $sContrasenaActual  = $objGenerico->Caracter(trim($aFormValues['contrasena']));
        $sContrasenaNueva   = $objGenerico-
>Caracter(trim($aFormValues['contrasena_nueva']));
        $sContrasenaRepetir = $objGenerico-
>Caracter(trim($aFormValues['contrasena_repetir']));

        // Validaremos cada una de las respectivas variables
        // Validar el nombre
        $bValidacion[] = $objResponse->validation-
>text($sContrasenaActual,"contrasena","validar_contrasena");
        $bValidacion[] = $objResponse->validation-
>text($sContrasenaNueva,"contrasena_nueva","validar_contrasena_nueva");
        $bValidacion[] = $objResponse->validation-
>text($sContrasenaRepetir,"contrasena_repetir","validar_contrasena_repetir");
    }
    if(($iTipoAccion == 2)and(!in_array(false,$bValidacion))){
        if($objUsuario->getClave()!=md5($sContrasenaActual)){
            $outHTML = "<a id='tt_contrasena' title='La contrase&ntilde;a que acaba de digitar
es incorrecta.'><img src='images/exclamation.gif'></a>";
            $objResponse->assign("validar_contrasena", "innerHTML", $outHTML);
            $objResponse->assign("contrasena", "style.border", "1px solid #FF0000");
```

```
            $objResponse->jquery->tooltip("#tt_contrasena","{cssClass:'tooltip-
blue',opacity:10}");

            $bValidacion[] = false;

        }

        if($sContrasenaNueva!=$sContrasenaRepetir){

            $outHTML = "<a id='tt_contrasena_nueva' title='La contrase&ntilde;a digitada no
es igual a la del campo repetir contrase&ntilde;a.'><img
src='images/exclamation.gif'></a>";

            $objResponse->assign("validar_contrasena_nueva", "innerHTML", $outHTML);

            $objResponse->assign("contrasena_nueva", "style.border", "1px solid #FF0000");

            $objResponse->assign("contrasena_repetir", "style.border", "1px solid #FF0000");

            $objResponse->jquery->tooltip("#tt_contrasena_nueva","{cssClass:'tooltip-
blue',opacity:10}");

            $bValidacion[] = false;

        }

    }

    if(!in_array(false,$bValidacion)){

        // Ejecutamos las respectivas acciones con la base de datos

        switch($iTipoAccion){

            case 1:

                // Asignamos los valores al objeto Usuario

                $objUsuario->setNombre($sNombre);

                $objUsuario->setApellidos($sApellido);

                $objUsuario->setDireccion("");

                if($sDireccion !== "") $objUsuario->setDireccion($sDireccion);

                $objUsuario->setFechaNacimiento($sFechaNacimiento);

                $objUsuario->setMovil("");
```

```
if($sMovil !== "") $objUsuario->setMovil($sMovil);

$objUsuario->setTelefono("");

if($sTelefono !== "") $objUsuario->setTelefono($sTelefono);

$objUsuario->setEmail($sEmail);

$objUsuario->setLogin($sLogin);

$sClaveAleatoria = fnGenerarClaveAleatorio(8);

$objUsuario->setClave(md5($sClaveAleatoria));


// Registramos el usuario

$bRegistro = $objUsuario->registrarUsuario();


$salidaHTMLtemp = "<p>Usuario registrado correctamente.<br />En unos momentos llegara un correo electr&oacute;nico al correo $sEmail informando acerca del registro, junto con la contrase&ntilde;a para poder ingresar al sitio web.</p>";


// Asunto del correo de registro

$sAsunto = "Registro Sitio Web Galeria de Arte";

$sContenido  = "$sNombre $sApellido, ud ha sido registrado en el sitio web de la galeria de arte.<br />";

$sContenido .= "Su Nombre de usuario es : <b>$sLogin</b><br />";

$sContenido .= "Su Contrase&ntilde;a es : <b>$sClaveAleatoria</b><br /><br />";

$sContenido .= "Para acceder a su cuenta ir a la siguiente direcci&oacute;n e iniciar sesi&oacute;n <a href='http://galeria.gennessis.com' target='_blank'>Galeria de Arte</a>";


// Enviamos un correo de registro al correo del usuario
```

```
            $sEnviarCorreo = $objCorreo->EnviarCorreo("Administrador galeria arte",
$sEmail, $sAsunto, $sContenido);

            //if($sEnviarCorreo != "ok");

            //    $salidaHTMLtemp .= "<p>Error al enviar el correo.<br />Error generado es
$sEnviarCorreo</p>";


            break;

        case 2:

            $objUsuario->setClave(md5($sContrasenaNueva));

            // Actualizamos el usuario

            $bRegistro = $objUsuario->actualizarUsuario();


            $salidaHTMLtemp = "<p>Contrase&ntilde;a modificada correctamente.</p>";

            break;

        case 3:

            // Asignamos los valores al objeto Usuario

            $objUsuario->setNombre($sNombre);

            $objUsuario->setApellidos($sApellido);

            $objUsuario->setDireccion("");

            if($sDireccion !== "") $objUsuario->setDireccion($sDireccion);

            $objUsuario->setFechaNacimiento($sFechaNacimiento);

            $objUsuario->setMovil("");

            if($sMovil !== "") $objUsuario->setMovil($sMovil);

            $objUsuario->setTelefono("");

            if($sTelefono !== "") $objUsuario->setTelefono($sTelefono);

            $objUsuario->setEmail($sEmail);
```

```
            // Registramos el usuario
            $bRegistro = $objUsuario->actualizarUsuario();

            $salidaHTMLtemp = "<p>Datos del usuario actualizados correctamente.</p>";
            break;
        }
        // Si fallo la ejecucion del comando imprimimos un error.
        if (!$bRegistro) $salidaHTML .= "Error. Imposible realizar la operacion.<br><br>";
        // Si no notificamos que todo fue un exito
        else $salidaHTML .= $salidaHTMLtemp;

        if($iTipoAccion!=1)
           $salidaHTML .= "<input type='button' class='button' value='Aceptar'
onclick=\"xajax_fnVerPerfil()\";'/>";
        $salidaHTML .= "</div>";
        // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML
        $objResponse->assign("subcontenido", "innerHTML", $salidaHTML);
    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnAccionDBUsuario");
```

Funciones para la administración de las noticias

```
/**
* Funcion encargada de cargar el listado de todas las noticias disponibles
* en el sitio
*
* @return xajaxResponse
*/
function fnMostrarNoticia($iId=null){
    require_once("modules/noticia/_noticia.php");
    if(count($aRegistros)>0){
        if(null == $iId){
            $salidaHTML .= "<table class='contentpaneopen'>";
            $salidaHTML .= "<tr><th colspan='2'><b>Nombre Noticia</b></th></tr>";
            foreach($aRegistros as $aRegistro){
                $salidaHTML .= "<tr>";
                $salidaHTML .= "<td width='4%'><a href='javascript:void(0)'
onclick='xajax_fnMostrarNoticia($aRegistro[0])'><img src=\"images/con_info.png\"></a>";
                $salidaHTML .= "<td>$aRegistro[1]</td>";
                $salidaHTML .= "</tr>";
            }
            $salidaHTML .= "</table>";
        }
        else{
            $objNoticia->__construct($iId);
            $objUsuario = new Usuario();
            $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
            $salidaHTML .= "<h2>Informaci&oacute;n de la Noticia</h2>";
```

```
        $salidaHTML .= "<span class='small'>Modificado por ".$objUsuario-
>getNombre()."</span><br /><br />";

        $salidaHTML .= "<table class='contentpaneopen'>";

        $salidaHTML .= "<tr><td colspan='2'>Nombre Producto</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objNoticia-
>getTitulo()."</td></tr>";

        $salidaHTML .= "<tr><td colspan='2'>Descripci&oacute;n</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objNoticia-
>getDescripcion()."</td></tr>";

        $salidaHTML .= "</table>";

        $salidaHTML .= "<br /><br /><input type='button' class='button' value='Regresar'
onclick=\"xajax_fnMostrarNoticia()\";'/>";

      }


   }

   $salidaHTML .= "</div>";

   // declaramos un objeto tipo respuesta (para las requeste que vendran.)

   $objResponse = new xajaxResponse();

   // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

   $objResponse->assign("contenido", "innerHTML", $salidaHTML);

   // la asignacion anterior es la que se va a devolver para aplicar los cambios.

   return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnMostrarNoticia");


/**
```

*** Funcion encargada de cargar el formulario para agregar un nueva noticia**

*

* @return xajaxResponse

*/

```
function fnAgregarNoticia(){
    // Declaramos los objetos necesarios
    $objNoticia = new Noticia();
    $objGenerico = new Generico();
    $objFechas = new Fechas();

    // declaramos un objeto tipo respuesta (para las request que vendran.)
    $objResponse=new xajaxResponse();

    $salidaHTML  = "<div class='contentheading'>Nueva Noticia</div>";
    $salidaHTML .= "<div> </div>";
    $salidaHTML .= "<div id='subcontenido'>";
    $salidaHTML .= "<form id='form' onsubmit='return false;'>";
    $salidaHTML .= "<div class='module-form'>";
    $salidaHTML .= "<table class='contentpaneopen'>";
    $salidaHTML .= "<tr><td width='30%'>T&iacute;tulo Noticia</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='titulo' name='titulo' value='' size='30'
/></td><td><div id='validar_titulo'></div></td></tr>";
    $salidaHTML .= "<tr><td colspan='4'>Contenido Noticia<span
id='validar_contenido'></span>";
    $salidaHTML .= "<div id='contenido_frame'>".$objGenerico-
>fnCargarEditor("",4,"contenido")."</div></td></tr>";
    // Obtenemos la fecha a registrar junto con la noticia
```

```
    $aFechas = $objFechas->FormatoFecha(date("Y-m-d"));

    $salidaHTML .= "<tr><td colspan='4'>Fecha: ".$aFechas['d']." de ".$aFechas['M']." de
".$aFechas['Y']."</td></tr>";

    $salidaHTML .= "</table>";

    $salidaHTML .= "</div>";

    $salidaHTML .= "<p> </p>";

    $salidaHTML .= "<input type='hidden' name='fecha' value='".date("Y-m-d H:i:s")."' />";

    $salidaHTML .= "<input type='button' class='button' value='Registrar' onclick=\"var
aFCKeditorXML=new Array();";

    $salidaHTML .=
"aFCKeditorXML['contenido']=FCKeditorAPI.GetInstance('contenido').GetXHTML();";

    $salidaHTML .=
"xajax_fnAccionBDNoticia(xajax.getFormValues('form'),1,aFCKeditorXML); return
false;\">";

    $salidaHTML .= "</form>";

    $salidaHTML .= "</div>";

    // Limpiamos la cache generada

    $objResponse->jquery->each("body>:not(.tail)","function(){\$(this).remove();}");

    $objResponse->jquery->each("body>:not(#tail)","function(){\$(this).remove();}");

    // Imprimimos el formulario de registro de noticia

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    // la asignacion anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnAgregarNoticia");


/**
```

```
* Funcion encargada de Cargar el formulario para modificar o eliminar
* una noticia
*
* @param int $iTipoAccion
* @return xajaxResponse
*/
function fnSeleccionarNoticia($iTipoAccion){
    require_once("modules/noticia/_noticia.php");
    if(count($aRegistros)>0){
        $salidaHTML  = "<div class='contentheading'>Seleccionar Noticia</div>";
        $salidaHTML .= "<div> </div>";
        $salidaHTML .= "<form id='form' onsubmit='return false;'>";
        $salidaHTML .= "<table class='contentpaneopen'>";
        $salidaHTML .= "<tr><th width='15%' style='text-
align:center;'><b>Selecionar</b></td><th><b>T&iacute;tulo Noticia</b></td></tr>";

        $iContador = 0;
        foreach ($aRegistros as $aRegistro) {
            $salidaHTML .= "<tr>";
            $salidaHTML .= "<td><div align='center'><input type='radio' name='id'
id='id$iContador' value='$aRegistro[0]'></div></td>";
            $salidaHTML .= "<td><label for='id$iContador' style='font-
size:14px;cursor:pointer'>$aRegistro[1]</label></td>";
            $salidaHTML .= "</tr>";
            $iContador++;
        }
```

```
// Selecionamos que accion se va a realizar cuando se de click en enviar
// 2 = Modificar Noticia
// 3 = Eliminar Noticia
if ($iTipoAccion == 2) {
    $tipo_funcion = "fnModificarNoticia";
    $value_boton  = "Modificar";
}

else {
    $tipo_funcion = "fnEliminarNoticia";
    $value_boton  = "Eliminar";
}

$salidaHTML .= "</table>";
    $salidaHTML .= "<input type='hidden' name='tipo_accion' value='$iTipoAccion' />";
    $salidaHTML .= "<input type='radio' disabled='true' name='id' value='0' checked
style='visibility:hidden'>";
    $salidaHTML .= "<br><div id='validacion_selector'></div><br>";
    $salidaHTML .= "<input type='button' class='button' value='$value_boton'
onclick='xajax_$tipo_funcion(xajax.getFormValues(\"form\")); return false;'>";
    $salidaHTML .= "</form>";
  }
  $salidaHTML .= "</div>";
  // declaramos un objeto tipo respuesta (para las requeste que vendran.)
  $objResponse = new xajaxResponse();
  // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML
```

```
        $objResponse->assign("contenido", "innerHTML", $salidaHTML);
        // la asignacion anterior es la que se va a devolver para aplicar los cambios.
        return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnSeleccionarNoticia");

/**
* Funcion encargada de cargar el formulario para modificar una noticia
* de acerdo a la selecion relaliza en la funcion fnSelectorFormularioNoticia
*
* @param Array $aFormValues
* @return xajaxResponse
*/
function fnModificarNoticia($aFormValues){
    // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
    if($aFormValues['id']==0){
        $salidaHTML = "<font color='red'>Debes selecionar una noticia.</font>";
        $objResponse->assign("validacion_selector", "innerHTML", $salidaHTML);
    }
    else{
        // Declaramos los objetos respectivos
        $objGenerico = new Generico();
        $objNoticia = new Noticia($aFormValues['id']);
        $objUsuario = new Usuario();
```

```
$objFechas = new Fechas();


  $salidaHTML  = "<div class='contentheading'>Modificar Noticia<br /></div><br />";

  $salidaHTML .= "<div> </div>";

  $salidaHTML .= "<div id='subcontenido'>";

  $salidaHTML .= "<form id='form' onsubmit='return false;'>";

  $salidaHTML .= "<div class='module-form'>";

  $salidaHTML .= "<table class='contentpaneopen'>";

  $salidaHTML .= "<tr><td width='30%'>T&iacute;tulo Noticia</td><td width='35%'
colspan='2'><input type='text' class='inputbox' id='titulo' name='titulo' value='".$objNoticia-
>getTitulo()."' size='30' /></td><td><div id='validar_titulo'></div></td></tr>";

  $salidaHTML .= "<tr><td colspan='4'>Contendio Noticia<span
id='validar_contenido'></span>";

 $salidaHTML .= "<div id='contenido_frame'>".$objGenerico-
>fnCargarEditor($objNoticia->getDescripcion(),4,"contenido")."</div></td></tr>";

 // Obtenemos la fecha a registrar junto con la noticia

 $aFechaTemp = explode(" ",$objNoticia->getFecha());

 $aFechas = $objFechas->FormatoFecha($aFechaTemp[0]);

 $salidaHTML .= "<tr><td colspan='4'>Fecha: ".$aFechas['d']." de ".$aFechas['M']." de
".$aFechas['Y']."</td></tr>";

  $salidaHTML .= "</table>";

  $salidaHTML .= "</div>";

  $salidaHTML .= "<p> </p>";

  //$salidaHTML .= "<input type='button' class='button' value='Registrar'
onclick='xajax_fnAccionBDProyectoPedagogico(xajax.getFormValues(\"form\"),1);'/>";

  $salidaHTML .= "<input type='button' class='button' value='Actualizar' onclick=\"var
aFCKeditorXML=new Array();";

  $salidaHTML .=
"aFCKeditorXML['contenido']=FCKeditorAPI.GetInstance('contenido').GetXHTML();";
```

```
        $salidaHTML .=
"xajax_fnAccionBDNoticia(xajax.getFormValues('form'),2,aFCKeditorXML); return
false;\">";

          $salidaHTML .= "<input type='hidden' id='id' name='id' value='".$aFormValues['id']."'
/>";

          $salidaHTML .= "</form>";

          $salidaHTML .= "</div>";

        // Limpiamos la cache generada

        $objResponse->jquery->each("body>:not(.tail)","function(){\$(this).remove();}");

        $objResponse->jquery->each("body>:not(#tail)","function(){\$(this).remove();}");

          // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

          $objResponse->assign("contenido", "innerHTML", $salidaHTML);

          }

    // la asignacion anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnModificarNoticia");


/**

* Funcion encargada de cargar el formulario para eliminar un noticia

* de acuerdo a la selecion realizada en la funcion fnSelectorFormularioNoticia

*

* @param Array $aFormValues

* @return xajaxResponse

*/

function fnEliminarNoticia($aFormValues){
```

```
// declaramos un objeto tipo respuesta (para las requeste que vendran.)
$objResponse = new xajaxResponse();
if($aFormValues['id']==0){
    $salidaHTML = "<font color='red'>Debes selecionar una noticia.</font>";
    $objResponse->assign("validacion_selector", "innerHTML", $salidaHTML);
}
else{
    $objNoticia = new Noticia($aFormValues['id']);
    $iAccion = $aFormValues['tipo_accion'];

      $salidaHTML  = "<div class='contentheading'>Eliminar Noticia</div>";
      $salidaHTML .= "<div> </div>";
      $salidaHTML .= "<div id='subcontenido'>";
      $salidaHTML .= "<form id='form' onsubmit='return false;'>";
      $salidaHTML .= "<font color='red'>Esta Realmente seguro de eliminar la noticia
<b>\"".$objNoticia->getTitulo()."\"</b>.<br />";
      $salidaHTML .= "Con &eacute;sta acci&oacute;n se borraran la informaci&oacute;n
de la noticia en el sitio web.</font><p> </p>";
     $salidaHTML .= "<input type='hidden' name='id' value=\"".$aFormValues['id']."\">";
      $salidaHTML .= "<input type='button' class='button' value='Si'
onclick=\"xajax_fnAccionBDNoticia(xajax.getFormValues('form'),3); return
false;\">    ";
      $salidaHTML .= "<input type='button' class='button' value='No'
onclick=\"xajax_fnSeleccionarNoticia(".$aFormValues['tipo_accion']."');\">";
     $salidaHTML .= "</form>";
      $salidaHTML .= "</div>";
```

```
        // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML
        $objResponse->assign("contenido", "innerHTML", $salidaHTML);
    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnEliminarNoticia");

/**
* Funcion encargada de ejecutar las acciones de agregado, modificado y elimnado
* de un noticia en la base de datos
*
* @param Array $aFormValues
* @param int $iTipoAccion
* @param String $aFCKeditor
* @return xajaxResponse
*/
function fnAccionBDNoticia($aFormValues, $iTipoAccion, $aFCKeditor) {
    // Declaramos nuestro objeto Generico
    $objGenerico = new Generico();
    // Declaramos nuestro objeto enlace
    $objNoticia = new Noticia();
    $objUsuario = new Usuario();
    $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
    // declaramos un objeto tipo respuesta (para las request que vendran.)
```

```
    $objResponse = new xajaxResponse();

    // Variable de control
    $bValidacion = array();
    $bValidacion[] = true;

    if (($iTipoAccion == 1) or ($iTipoAccion == 2)) {
        $sTitulo    = $objGenerico->Caracter(trim($aFormValues['titulo']));
        $sContenido = $aFCKeditor['contenido'];
        $sFecha     = $aFormValues['fecha'];
        // Validamos las valores obtenidos
        $bValidacion[] = $objResponse->validation->text($sTitulo,"titulo","validar_titulo");
        $bValidacion[] = $objResponse->validation-
>empt($sContenido,"contenido_frame","validar_contenido");
    }

    if(!in_array(false,$bValidacion)){
        switch ($iTipoAccion){
            // Registramos una noticia
            case 1:
                $sContenido = ereg_replace("\"", "\'", $sContenido);
                //$sDescripcion = ereg_replace("<p>", "", $sDescripcion);
                //$sDescripcion = ereg_replace("</p>", "<br />", $sDescripcion);

                // Asignamos los valores obtenidos a la clase
                $objNoticia->setTitulo($sTitulo);
```

```
    $objNoticia->setDescripcion($sContenido);
    $objNoticia->setFecha($sFecha);
    $objNoticia->setUsuario($objUsuario->getId());

    // Registramos la noticia
    $bRegistro = $objNoticia->registrarNoticia();

    $salidaHTMLtemp = "<p>Noticia registrada correctamente.</p>";
    break;
    // Modificamos una noticia
case 2:
    $objNoticia->__construct($aFormValues['id']);
    $sContenido = ereg_replace("\"", "\'", $sContenido);

    // Asignamos los valores obtenidos a la clase
    $objNoticia->setTitulo($sTitulo);
    $objNoticia->setDescripcion($sContenido);

    // Actualizamos la noticia
    $bRegistro = $objNoticia->actualizarNoticia();

    $salidaHTMLtemp = "<p>Noticia modificada correctamente.</p>";
    break;
    // Eliminamos una noticia
case 3:
```

```
            $objNoticia->__construct($aFormValues['id']);
            // Eliminamos la noticia
            $bRegistro = $objNoticia->eliminarNoticia();

            $salidaHTMLtemp = "<p>Noticia eliminada correctamente.</p>";
            break;
        }

        // Si fallo la ejecucion del comando imprimimos un error.
        if (!$bRegistro) $salidaHTML .= "Error. Imposible realizar la operacion.<br><br>";
        // Si no notificamos que todo fue un exito
        else $salidaHTML .= $salidaHTMLtemp;

        $salidaHTML .= "<input type='button' class='button' value='Aceptar'
onclick=\"xajax_fnMostrarNoticia()\";'/>";
        $salidaHTML .= "</div>";
        // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML
        $objResponse->assign("subcontenido", "innerHTML", $salidaHTML);
    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnAccionBDNoticia");
```

Funciones para la administración de las diversas obras de arte

```
/**
 * Funcion encargada de Mostrar todos los servicios registrados en la Base de Datos
 * @return HTTPReponse
 */
function fnMostrarGaleria($iId=null){
    require_once("modules/galeria/_obraarte.php");
    // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
    $aRegistrosGaleria = array();
    if(count($aRegistros)>0){
        if(null == $iId){
          $salidaHTML .= "<table class='contentpaneopen'>";
          $salidaHTML .= "<tr><th colspan='2'><b>Nombre obra de arte</b></th></tr>";
          foreach($aRegistros as $aRegistro){
             $salidaHTML .= "<tr>";
             $salidaHTML .= "<td width='4%'><a href='javascript:void(0)'
onclick='xajax_fnMostrarGaleria($aRegistro[0])'><img src=\"images/con_info.png\"></a>";
             $salidaHTML .= "<td>$aRegistro[1]</td>";
             $salidaHTML .= "</tr>";
          }
          $salidaHTML .= "</table>";
       }
       else{
          $objObraArte = new ObraArte();
          $objObraArte->__construct($iId);
```

```
        $objImagen = new Imagen();

        $salidaHTML .= "<h2>Informaci&oacute;n de la Obra de Arte</h2>";

        $salidaHTML .= "<table class='contentpaneopen'>";

        $salidaHTML .= "<tr><td colspan='2'>Nombre Obra de Arte</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getNombre()."</td></tr>";

        $salidaHTML .= "</table>";

        $aRegistrosGaleria = $objImagen-
>getImagenAll("ID_OBRA_ARTE='".$objObraArte->getId()."' ORDER BY ORDEN ASC");

        if(count($aRegistrosGaleria)>0){

            $salidaHTML .= "<div class='blog_more'><ul class='gallery'>";

            foreach ($aRegistrosGaleria as $aFoto){

                $salidaHTML .= "<li><a href='gallery/".$objObraArte->getId()."/$aFoto[3]'
rel='prettyPhoto[gallery]' title=''><img src='gallery/".$objObraArte-
>getId()."/thumbnail/$aFoto[3]' alt='$aFoto[1]' /></a></li>";

            }

            $salidaHTML .= "</ul></div>";

        }

        $salidaHTML .= "<table class='contentpaneopen'>";

        $salidaHTML .= "<tr><td colspan='2'>Descripci&oacute;n</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getDescripcion()."</td></tr>";

        $salidaHTML .= "<tr><td colspan='2'>Dimensi&oacute;n</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getDimension()."</td></tr>";

        $salidaHTML .= "<tr><td colspan='2'>Precio</td></tr>";

        $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getPrecio()."</td></tr>";
```

```
            $salidaHTML .= "<tr><td colspan='2'>Estado</td></tr>";

            $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getEstado()."</td></tr>";

            $salidaHTML .= "<tr><td colspan='2'>Tiempo de elaboraci&oacute;n</td></tr>";

            $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getTiempoElaboracion()."</td></tr>";

            $salidaHTML .= "<tr><td colspan='2'>Descripci&oacute;n</td></tr>";

            $salidaHTML .= "<tr><td width='5%'></td><td>".$objObraArte-
>getDescripcion()."</td></tr>";

            $salidaHTML .= "</table>";

            $salidaHTML .= "<br /><br /><input type='button' class='button' value='Regresar'
onclick=\"xajax_fnMostrarGaleria()\";'/>";

        }

    }

    $salidaHTML .= "</div>";

    // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    if(count($aRegistrosGaleria)>0){

        $objResponse->jquery->prettyPhoto("a[rel^='prettyPhoto']","{

            animationSpeed: 'normal', /* fast/slow/normal */

            padding: 40, /* padding for each side of the picture */

            opacity: 0.35, /* Value betwee 0 and 1 */

            showTitle: true, /* true/false */

            allowresize: true, /* true/false */

            counter_separator_label: '/', /* The separator for the gallery counter 1 'of' 2 */

            theme: 'light_rounded' /* light_rounded / dark_rounded / light_square / dark_square
*/
```

```
            }");
    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnMostrarGaleria");

/**
 * Funcion encargada de cargar el formulario para agregar un servicio
 * @return HTTPReponse
 */
function fnAgregarGaleria(){
    // Declaramos los objetos respectivos
    $objGenerico = new Generico();
    $objTecnica = new Tecnica();

    // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
    // Eliminamos la cache que probablemente se haya generado
    $objResponse->jquery->remove("iframe");
    $objResponse->jquery->remove(".ui-dialog");
    $objResponse->jquery->dialog("#dialogo_subirImagen","destroy");
    $objResponse->jquery->remove("#dialogo_subirImagen");

    // Eliminamos el archivo de configuracion en caso de que exista
```

```
    if(file_exists($objGenerico->serverpath()."/gallery/tmp/configuracion.dat")){
        fnBorrarDirectorio($objGenerico->serverpath()."/gallery/tmp/configuracion.dat");
    }


    $salidaHTML  = "<div class='contentheading'>Nuevo Proyecto - Galer&iacute;a de Im&aacute;genes</div>";
    $salidaHTML .= "<div> </div>";
    $salidaHTML .= "<div id='subcontenido'>";
    $salidaHTML .= "<form id='form' onsubmit='return false;'>";
    $salidaHTML .= "<div class='module-form'>";
    $salidaHTML .= "<table class='contentpaneopen'>";
    $salidaHTML .= "<tr><td width='30%'>Nombre Cuadro</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='nombre' name='nombre' value='' size='30' /></td><td><div id='validar_nombre'></div></td></tr>";
    $salidaHTML .= "<tr><td width='30%'>Tecnica</td><td width='35%' colspan='2'>";
    // Obtenemos todos los registros de las tecnicas
    $aRegistros = $objTecnica->getTecnicaAll();
    $salidaHTML .= "<select name='tecnica' id='tecnica'>";
    $salidaHTML .= "<option value='0'>Selecionar...</option>";
    foreach ($aRegistros as $aRegistro){
        $salidaHTML .= "<option value='$aRegistro[0]'>$aRegistro[1]</option>";
    }
    $salidaHTML .= "</select>";
    $salidaHTML .= "</td><td><div id='validar_tecnica'></div></td></tr>";
    $salidaHTML .= "<tr><td width='30%'>Dimensi&oacute;n</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='dimension' name='dimension' value='' size='30' /></td><td><div id='validar_dimension'></div></td></tr>";
```

```
    $salidaHTML .= "<tr><td width='30%'>Precio</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='precio' name='precio' value='' size='30' /></td><td><div id='validar_precio'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Tiempo Elaboraci&oacute;n</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='elaboracion' name='elaboracion' value='' size='30' /></td><td><div id='validar_elaboracion'></div></td></tr>";

    $salidaHTML .= "<tr><td width='30%'>Estado</td><td width='35%' colspan='2'>";

    $salidaHTML .= "<select name='estado' id='estado'><option value='0'>Selecionar...</option><option value='1'>Vendido</option><option value='2'>Disponible</option></select></td><td><div id='validar_estado'></div></td></tr>";

    $salidaHTML .= "<tr><td colspan='4'>Descripci&oacute;n<span id='validar_descripcion'></span>";

    $salidaHTML .= "<div id='descripcion_frame'>".$objGenerico->fnCargarEditor("",4,"descripcion")."</div></td></tr>";

    $salidaHTML .= "</table>";

    $salidaHTML .= "<br /><div id='progressbar' style='height:2px;'></div><br />";

    // Galeria

    /*$salidaHTML .= "<a href='javascript:void(0);' id='linkDialogo_subirImagen' class='ui-state-default ui-corner-all' onclick='\$(\"#formularioImagen\").empty();xajax_fnFormularioSubirImagen();'><span class='ui-icon ui-icon-image'></span>    Agregar Imagen</a>";

    $salidaHTML .= "<div id='dialogo_subirImagen' title='Subir Imagen'><div id='formularioImagen'></div></div>";*/

    $salidaHTML .= "<div id='formularioImagen'></div>";


    $salidaHTML .= "<br /><a href='javascript:void(0);' id='dialogo_link' class='ui-state-default ui-corner-all' onclick='\$(\"#incFormulario\").empty();xajax_incFormulario(\"fnAccionSubirImagen\");'><span class='ui-icon ui-icon-image'></span>    Buscar Imagen</a><br />";

    // Incluimos los archivos para la configuracion del formulario para subir archivos
```

```
    $objResponse->includeScriptOnce("modules/upload/upload.js");

    // Cargamos la configuracion de la subida

    $objResponse->script("xajax_fnCargarConfiguracionUpload()");

    $salidaHTML .= "<div id='dialogo' title='Seleccionar Imagen'><div
id='incFormulario'></div></div>";



    $salidaHTML .= "<div> </div>";

    $salidaHTML .= "<div class='demo ui-widget ui-helper-clearfix'>";

    $salidaHTML .= "<ul id='gallery' class='gallery ui-helper-reset ui-helper-clearfix'>";

    $salidaHTML .= "</ul>";

    // Papelera

    $salidaHTML .= "<div id='trash' class='ui-widget-content ui-state-hover'>";

    $salidaHTML .= "<h4 class='ui-widget-header ui-corner-all'><span class='ui-icon ui-icon-
trash'>Papelera</span> Papelera</h4>";

    $salidaHTML .= "</div>";

    $salidaHTML .= "</div>";



    $salidaHTML .= "</div>";

    $salidaHTML .= "<p> </p>";

    $salidaHTML .= "<input type='button' class='button' value='Registrar' onclick=\"var
aFCKeditorXML=new Array();";

    $salidaHTML .=
"aFCKeditorXML['descripcion']=FCKeditorAPI.GetInstance('descripcion').GetXHTML();";

    $salidaHTML .=
"xajax_fnAccionBDProyecto(xajax.getFormValues('form'),1,aFCKeditorXML,\$('#gallery').s
ortable('toArray')); return false;\">";

    $salidaHTML .= "</form>";
```

```
    $salidaHTML .= "</div>";


    $objResponse->removeScript("js/gallery.js");

    $objResponse->includeScriptOnce("js/gallery.js");

    $objResponse->removeCSS("css/gallery.css");


    // Eliminamos la cache generada por el jQuery

    $objResponse->jquery->each("body>:not(.tail)","function(){\$(this).remove();}");

    $objResponse->jquery->each("body>:not(#tail)","function(){\$(this).remove();}");

    // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    $objResponse->includeCSS("css/gallery.css");

    // Funciones jquery

    $objResponse->jquery->progressbar("#progressbar","{value:100}");

    /*$objResponse->jquery->hover("#linkDialogo_subirImagen","function() {
\$(this).addClass('ui-state-hover'); },function() { \$(this).removeClass('ui-state-hover'); }");

    $objResponse->jquery->addClass("#linkDialogo_subirImagen","dialogo_link");*/

    $objResponse->jquery->addClass("#trash","ui-corner-all");


    // Funciones jQuery para el dialogo de subir imagen

    //$sButtons  = "'Aceptar':function() {var aFCKeditorXML=new
Array();aFCKeditorXML['descripcionImagen']=FCKeditorAPI.GetInstance('descripcionImag
en').GetXHTML();xajax_fnValidarSubirImagen(xajax.getFormValues('formImagen'),aFCKe
ditorXML);}";

    //$sButtons .= ",'Cancelar':function() {\$(this).dialog('close');}";
```

```
    //$objResponse->jquery->click("#linkDialogo_subirImagen","function(){\$('#dialogo_subirImagen').dialog('open');return false;}");

    //$objResponse->jquery->dialog("#dialogo_subirImagen","{autoOpen: false, bgiframe: true,
resizable:false,closeOnEscape:false,height:350,width:420,modal:true,buttons:{".$sButtons."}}");


    $objResponse->jquery->hover("#dialogo_link","function() { \$(this).addClass('ui-state-hover'); },function() { \$(this).removeClass('ui-state-hover'); }");

    $objResponse->jquery->addClass("#dialogo_link","dialogo_link");

    // Funciones jQuery para el dialogo de selecionar la imagen

    $objResponse->jquery->click("#dialogo_link","function(){\$('#dialogo').dialog('open');return false;}");

    $objResponse->jquery->dialog("#dialogo","{autoOpen: false, bgiframe: true,
resizable:false,closeOnEscape:false,height:220,width:420,modal:true,buttons:{'Cerrar':function() {\$(this).dialog('close');}}}");


    // la asignacion anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnAgregarGaleria");


function fnCargarConfiguracionImagen($sNombreImagen){

    $objResponse = new xajaxResponse();

    // Abrimos el archvo como de solo lectura

    $archivoImagen = @fopen("gallery/tmp/configuracion.dat", "r");

    // Variable que va a tener el contenido del archivo

    $sContenidoArchivo = array();
```

```
    while($sLine = fgets($archivoImagen)){
    if($sLine !== "")
        $sContenidoArchivo[] = $sLine;
    }
    // Proceso en el cual se va a mirar si el $sNombre pasado como parametro esta
contenido detro
    // del vector que contiene el contenido del archivo, de ser asi lo eliminamos
    foreach($sContenidoArchivo as $sTemp){
        $iEncontrados   = preg_match("/$sNombreImagen/", $sTemp);
        if($iEncontrados != 0){
            // Formato del vector
            // 1||adDR9R4E.png||Proyecto midas||Esto es un proyecto muy...||250,568
            $aConfiguracion = explode("||", $sTemp);
            //$objResponse->script("alert('".$aConfiguracion[2]."')");
            $objResponse->jquery->empty("#formularioImagen");
            $objResponse-
>call("xajax_fnFormularioSubirImagen('$sNombreImagen','".base64_encode($aConfigurac
ion[2])."','".base64_encode($aConfiguracion[3])."')");

//fnFormularioSubirImagen(base64_encode($aConfiguracion[2]),base64_encode($aConfig
uracion[3]));
            $objResponse->jquery->dialog("#dialogo_subirImagen","open");
        }
    }
    @fclose($archivoImagen);
    return  $objResponse;
}
```

```
$xajax->register(XAJAX_FUNCTION, "fnCargarConfiguracionImagen");


/**
 * Funcion encargada de ejecutar las acciones una vez subida la imagen
 * @param array $fileArr
 * @return HTTPResponse $objReponse
 */
function fnAccionSubirImagen($fileArr){
	$objGenerico = new Generico();
	$objImage = new Image();
	$aFile = explode(",",$fileArr);
	$objResponse = new xajaxResponse();
	// Obtenemos la extension de la imagen
	$sExtension = substr($aFile[1], strrpos($aFile[1], "."));
	// Colocamos un nombre a imgen que se subio
	$sNameImagen = claveAleatoria(8).$sExtension;
	// Directorio donde se va a colocar la imagen con el nombre que acabos de crear
	$sTmpImage = "gallery/tmp";
	// Copiamos la imagen al directorio selecionado en le paso anterior
	fnCopiarArchivo($aFile[0], $objGenerico->serverpath()."/".$sTmpImage."/".$sNameImagen);
	//RedimensionarImagen($objGenerico->serverpath()."/".$sTmpImage, $sNameImagen);
	$objImage->image_path($sTmpImage."/".$sNameImagen);
	// Redimensionamos la imagen en caso de que sea mayor a 640 pixeles
	if(($objImage->get_image_width()>640)||($objImage->get_image_heigth()>480)){
```

```
            $objImage->image_resize(640,480);
          }
          // Creamos el thumbnail de la imagen subida
        $objImage->image_path($sTmpImage."/".$sNameImagen);
          //$objImage->image_thumbnail(72,58);
        $objImage->image_thumbnail(100,72);

        $objResponse->call("xajax_fnValidarSubirImagen('$sNameImagen','1')");
          return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnAccionSubirImagen");

/**
 * Funcion encargada de validar las imagenes a agregar a la galeria de imagenes del projecto
 * @param array $aFormValues
 * @param array $aFCKeditor
 * @return HTTPResponse $objResponse
 */
//function fnValidarSubirImagen($aFormValues,$aFCKeditor){
function fnValidarSubirImagen($sNombreImagen,$iAccion){
    // Declaramos un objeto tipo respuesta (para las requeste que vendran.
    $objResponse = new xajaxResponse();
    $objGenerico = new Generico();
    $objImage = new Image();
```

```
    $sTitulo = "";


    $objImage->image_path("gallery/tmp/".$sNombreImagen);


    // Generamos el archivo de configuracion para la galeria del projecto

    $objResponse->call("xajax_fnArchivoImagenes('1','$sNombreImagen','".$objImage->get_image_width()."-".$objImage->get_image_heigth()."')");


    if($iAccion==1){

        // Generamos el contenido de la imagen

        $salidaHTML  = "<li id='$sNombreImagen' class='ui-widget-content ui-corner-top'>";

        $salidaHTML .= "<img src='gallery/tmp/thumbnail/$sNombreImagen' alt='$sTitulo' width='100' height='72' />";

        $salidaHTML .= "<a href='gallery/tmp/$sNombreImagen' title='Ver imagen grande' width='".$objImage->get_image_width()."' heigth='".$objImage->get_image_heigth()."' class='ui-icon ui-icon-zoomin'>Ver grande</a>";

        $salidaHTML .= "<a href='javascript:void(0)' title='Modificar imagen' class='ui-icon ui-icon-wrench'>Modificar imagen</a>";

        $salidaHTML .= "<a href='javascript:void(0)' title='Eliminar esta imagen' class='ui-icon ui-icon-trash'>Eliminar imagen</a>";

        $salidaHTML .= "</li>";


        $objResponse->append("gallery", "innerHTML", $salidaHTML);

        $objResponse->removeScript("js/gallery.js");

        $objResponse->includeScriptOnce("js/gallery.js");

    }

    else {
```

```
        $objResponse->jquery->remove(".ui-dialog/
../img[src$='gallery/tmp/".$sNombreImagen."']");

        $objResponse->jquery-
>attr("img[src$='gallery/tmp/thumbnail/".$sNombreImagen."']","alt\",\"$sTitulo");

    }

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnValidarSubirImagen");


/**
```

*** Funcion encargada de generar el archivo de configuracion de la galeria de imagenes del projecto**

```
 * @param int $iAccion[1=Galeria; 2=Papelera]

 * @param string $sNombre

 * @param string $sTitulo[optional]

 * @param string $sDescripcion[optional]

 * @param string $sTamanio[optional]

 * @param int $iIdProjecto[optional]

 */

function fnArchivoImagenes($iAccion,$sNombre,$sTamanio=null,$iIdProyecto=null){

    // declaramos un objeto tipo respuesta (para las requeste que vendran.)

    $objResponse = new xajaxResponse();

    // Si el archivo no existe lo creamos

    if(!file_exists("gallery/tmp/configuracion.dat")){

        $archivoImagen = fopen("gallery/tmp/configuracion.dat", "w");

        fclose($archivoImagen);

    }
```

```
    // Abrimos el archvo como de solo lectura
    $archivoImagen = fopen("gallery/tmp/configuracion.dat", "r");
    // Variable que va a tener el contenido del archivo
    $sContenidoArchivo = array();
    if($archivoImagen){
        while($sLine = fgets($archivoImagen)){
          if($sLine !== "")
             $sContenidoArchivo[] = $sLine;
        }
       // Proceso en el cual se va a mirar si el $sNombre pasado como parametro esta
contenido detro
       // del vector que contiene el contenido del archivo, de ser asi lo eliminamos
          foreach($sContenidoArchivo as $sTemp){
           $iEncontrados   = preg_match("/$sNombre/", $sTemp);
           if($iEncontrados != 0){
              $sTemporal = $sTemp;
              $sTemporal = substr($sTemporal, 1, strlen($sTemporal));
              $sTemporal = $iAccion.$sTemporal;
              $sContenidoArchivo = array_remval($sTemp, $sContenidoArchivo);
           }
          }
    }
    fclose($archivoImagen);

    // Formato archivo
    // 1||adDR9R4E.png||Proyecto midas||Esto es un proyecto muy...||250,568
```

```
    if(null !== $sTamanio)
        $sContenidoArchivo[] = $iAccion."||".$sNombre."||".$sTamanio."\n";
    else
        $sContenidoArchivo[] = $sTemporal;

    // Almacemos el contenido del vector dentro de la siguiente variable para introducirlo nuevamente dentro del archivo de configuracion
    $sArchivo = "";
    foreach ($sContenidoArchivo as $sTemp){
        $sArchivo .= $sTemp;
    }
    // Almacenamos el contenido del archivo dentro del archivo
    $archivoImagen = fopen("gallery/tmp/configuracion.dat", "w");
    fwrite($archivoImagen, $sArchivo);
    fclose($archivoImagen);
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnArchivoImagenes");

/**
* Remove a value from a array
* @param string $val
* @param array $arr
* @return array $array_remval
* @example $stack=Array('apple','banana','pear','apple', 'cherry', 'apple');<br />
```

```
* array_remval("apple", $stack);<br />
* output: Array('banana','pear', 'cherry')
*/
function array_remval($val, &$arr){
        $array_remval = $arr;
        for($x=0;$x<count($array_remval);$x++){
         $i=array_search($val,$array_remval);
         if (is_numeric($i)) {
                $array_temp  = array_slice($array_remval, 0, $i );
                $array_temp2 = array_slice($array_remval, $i+1, count($array_remval)-1 );
                $array_remval = array_merge($array_temp, $array_temp2);
         }
        }
        return $array_remval;
}

/**
 * Funcion encargada de seleccionar un Servicio a modificar o eliminar en la Base de Datos
 * @return HTTPReponse
 * @param int $iTipoAccion - Accion a ejecutar 2=Modificar; 3=Eliminar
 */
function fnSeleccionarGaleria($iTipoAccion){
    require_once("modules/galeria/_obraarte.php");
    if(count($aRegistros)>0){
        $salidaHTML  = "<div class='contentheading'>Seleccionar obra de arte</div>";
```

```
        $salidaHTML .= "<div> </div>";

        $salidaHTML .= "<form id='form' onsubmit='return false;'>";

        $salidaHTML .= "<table class='contentpaneopen'>";

        $salidaHTML .= "<tr><th width='15%' style='text-
align:center;'><b>Selecionar</b></td><th><b>Nombre Proyecto</b></td></tr>";


        $iContador = 0;

      foreach ($aRegistros as $aRegistro) {

          $salidaHTML .= "<tr>";

          $salidaHTML .= "<td><div align='center'><input type='radio' name='id'
id='id$iContador' value='$aRegistro[0]'></div></td>";

          $salidaHTML .= "<td><label for='id$iContador' style='font-
size:14px;cursor:pointer'>$aRegistro[1]</label></td>";

          $salidaHTML .= "</tr>";

          $iContador++;

      }


      // Selecionamos que accion se va a realizar cuando se de click en enviar

      // 2 = Modificar Director

      // 3 = Eliminar Director

      if ($iTipoAccion == 2) {

          $tipo_funcion = "fnModificarGaleria";

          $value_boton  = "Modificar";

      }

      else {

          $tipo_funcion = "fnEliminarGaleria";

          $value_boton  = "Eliminar";
```

```
        }


        $salidaHTML .= "</table>";

        $salidaHTML .= "<input type='hidden' name='tipo_accion' value='$iTipoAccion' />";

        $salidaHTML .= "<input type='radio' disabled='true' name='id' value='0' checked
style='visibility:hidden'>";

        $salidaHTML .= "<br><div id='validacion_selector'></div><br>";

        $salidaHTML .= "<input type='button' class='button' value='$value_boton'
onclick='xajax_$tipo_funcion(xajax.getFormValues(\"form\")); return false;'>";

        $salidaHTML .= "</form>";

    }

    $salidaHTML .= "</div>";

    // declaramos un objeto tipo respuesta (para las requeste que vendran.)

    $objResponse = new xajaxResponse();

    // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML

    $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    // la asignacion anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnSeleccionarGaleria");


/**

 * Funcion encargada de cargar el formulario de modificar la informacion de un
Servicio en la Base de Datos

 * @return HTTPReponse

 * @param array $aFormulario - Parametros enviados por el formulario
```

```
 */
function fnModificarGaleria($aFormulario){
    // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
    if($aFormulario['id']==0){
        $salidaHTML = "<font color='red'>Debes selecionar una galeria.</font>";
        $objResponse->assign("validacion_selector", "innerHTML", $salidaHTML);
    }
    else{
        // Declaramos los objetos respectivos
        $objGenerico = new Generico();
        $objObraArte = new ObraArte($aFormulario['id']);
      $objImagen = new Imagen();
      $objTecnica = new Tecnica();

      // Obtenemos las imagenes registradas en la base de datos del proyecto
      $aRegistros = $objImagen->getImagen("ID_OBRA_ARTE='".$aFormulario['id']."'");

      // Eliminamos la cache que probablemente se haya generado
      $objResponse->jquery->remove("iframe");
      $objResponse->jquery->remove(".ui-dialog");
      $objResponse->jquery->dialog("#dialogo_subirImagen","destroy");
      $objResponse->jquery->remove("#dialogo_subirImagen");

      // Eliminamos el archivo de configuracion en caso de que exista
```

```
        if(file_exists($objGenerico->serverpath()."/gallery/tmp/configuracion.dat")){
            fnBorrarDirectorio($objGenerico->serverpath()."/gallery/tmp/configuracion.dat");
        }


        $salidaHTML  = "<div class='contentheading'>Nuevo Proyecto - Galer&iacute;a de Im&aacute;genes</div>";
        $salidaHTML .= "<div> </div>";
        $salidaHTML .= "<div id='subcontenido'>";
        $salidaHTML .= "<form id='form' onsubmit='return false;'>";
        $salidaHTML .= "<div class='module-form'>";
        $salidaHTML .= "<table class='contentpaneopen'>";
        $salidaHTML .= "<tr><td width='30%'>Nombre Cuadro</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='nombre' name='nombre' value='".$objObraArte->getNombre()."' size='30' /></td><td><div id='validar_nombre'></div></td></tr>";
        $salidaHTML .= "<tr><td width='30%'>Tecnica</td><td width='35%' colspan='2'>";
        // Obtenemos todos los registros de las tecnicas
        $aRegistros = $objTecnica->getTecnicaAll();
        $salidaHTML .= "<select name='tecnica' id='tecnica'>";
        //$salidaHTML .= "<option value='0'>Selecionar...</option>";
        $sSelecion = "";
        foreach ($aRegistros as $aRegistro){
            if($aRegistro[0]==$objObraArte->getTecnica()->getId()) $sSelecion = "selected";
            $salidaHTML .= "<option value='$aRegistro[0]' $sSelecion>$aRegistro[1]</option>";
        }
        $salidaHTML .= "</select>";
```

```
$salidaHTML .= "</td><td><div id='validar_tecnica'></div></td></tr>";

$salidaHTML .= "<tr><td width='30%'>Dimensi&oacute;n</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='dimension' name='dimension' value='".$objObraArte->getDimension()."' size='30' /></td><td><div id='validar_dimension'></div></td></tr>";

$salidaHTML .= "<tr><td width='30%'>Precio</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='precio' name='precio' value='".$objObraArte->getPrecio()."' size='30' /></td><td><div id='validar_precio'></div></td></tr>";

$salidaHTML .= "<tr><td width='30%'>Tiempo Elaboraci&oacute;n</td><td width='35%' colspan='2'><input type='text' class='inputbox' id='elaboracion' name='elaboracion' value='".$objObraArte->getTiempoElaboracion()."' size='30' /></td><td><div id='validar_elaboracion'></div></td></tr>";

$salidaHTML .= "<tr><td width='30%'>Estado</td><td width='35%' colspan='2'>";

$salidaHTML .= "<select name='estado' id='estado'><option value='1' selected>Vendido</option><option value='2' ";if($objObraArte->getEstado()=="Disponible")$salidaHTML .= "selected";$salidaHTML .= ">Disponible</option></select></td><td><div id='validar_estado'></div></td></tr>";

$salidaHTML .= "<tr><td colspan='4'>Descripci&oacute;n<span id='validar_descripcion'></span>";

$salidaHTML .= "<div id='descripcion_frame'>".$objGenerico->fnCargarEditor($objObraArte->getDescripcion(),4,"descripcion")."</div></td></tr>";

$salidaHTML .= "</table>";

$salidaHTML .= "<br /><div id='progressbar' style='height:2px;'></div><br />";

// Galeria


$salidaHTML .= "<div id='formularioImagen'></div>";


$salidaHTML .= "<br /><a href='javascript:void(0);' id='dialogo_link' class='ui-state-default ui-corner-all' onclick='\$(\"#incFormulario\").empty();xajax_incFormulario(\"fnAccionSubirImagen\");'><span class='ui-icon ui-icon-image'></span>    Buscar Imagen</a><br />";
```

```
        // Incluimos los archivos para la configuracion del formulario para subir archivos

        $objResponse->includeScriptOnce("modules/upload/upload.js");

        // Cargamos la configuracion de la subida

        $objResponse->script("xajax_fnCargarConfiguracionUpload()");

        $salidaHTML .= "<div id='dialogo' title='Seleccionar Imagen'><div
id='incFormulario'></div></div>";


        $salidaHTML .= "<div> </div>";

        $salidaHTML .= "<div class='demo ui-widget ui-helper-clearfix'>";

        $salidaHTML .= "<ul id='gallery' class='gallery ui-helper-reset ui-helper-clearfix'>";

        $aRegistros = $objImagen->getImagenAll("ID_OBRA_ARTE='".$aFormulario['id']."'
ORDER BY ORDEN ASC");

        if(count($aRegistros)>0){

            $sContenido = "";

            foreach ($aRegistros as $aRegistro){

                // Formato archivo

                // 1||adDR9R4E.png||250,568

                $sContenido .= "1||".$aRegistro[3]."||".$aRegistro[1]."\n";

                // Copiamos las imagenes al directorio temporal

                fnCopiarArchivo($objGenerico->serverpath()."/gallery/".$objObraArte-
>getId()."/".$aRegistro[3], $objGenerico->serverpath()."/gallery/tmp/".$aRegistro[3]);

                fnCopiarArchivo($objGenerico->serverpath()."/gallery/".$objObraArte-
>getId()."/thumbnail/".$aRegistro[3], $objGenerico-
>serverpath()."/gallery/tmp/thumbnail/".$aRegistro[3]);


                $aTamanio = explode("-",$aRegistro[1]);

                $salidaHTML .= "<li id='$aRegistro[3]' class='ui-widget-content ui-corner-top'>";
```

```
            $salidaHTML .= "<img src='gallery/tmp/thumbnail/$aRegistro[3]' alt='' width='100' height='72' />";

            $salidaHTML .= "<a href='gallery/tmp/$aRegistro[3]' title='Ver imagen grande' width='$aTamanio[0]' heigth='$aTamanio[1]' class='ui-icon ui-icon-zoomin'>Ver grande</a>";

            $salidaHTML .= "<a href='javascript:void(0)' title='Modificar imagen' class='ui-icon ui-icon-wrench'>Modificar imagen</a>";

            $salidaHTML .= "<a href='javascript:void(0)' title='Eliminar esta imagen' class='ui-icon ui-icon-trash'>Eliminar imagen</a>";

            $salidaHTML .= "</li>";

        }

        $archivoImagen = fopen("gallery/tmp/configuracion.dat", "w");

        fwrite($archivoImagen ,$sContenido);

        fclose($archivoImagen);

    }

    $salidaHTML .= "</ul>";

    // Papelera

    $salidaHTML .= "<div id='trash' class='ui-widget-content ui-state-hover'>";

    $salidaHTML .= "<h4 class='ui-widget-header ui-corner-all'><span class='ui-icon ui-icon-trash'>Papelera</span> Papelera</h4>";

    $salidaHTML .= "</div>";

    $salidaHTML .= "</div>";



    $salidaHTML .= "</div>";

    $salidaHTML .= "<p> </p>";

    $salidaHTML .= "<input type='button' class='button' value='Actualizar' onclick=\"var aFCKeditorXML=new Array();";

    $salidaHTML .= "aFCKeditorXML['descripcion']=FCKeditorAPI.GetInstance('descripcion').GetXHTML();";
```

```
	$salidaHTML .= "xajax_fnAccionBDProyecto(xajax.getFormValues('form'),2,aFCKeditorXML,\$('#gallery').sortable('toArray')); return false;\">";

	$salidaHTML .= "<input type='hidden' id='id' name='id' value='".$aFormulario['id']."' />";

	$salidaHTML .= "</form>";

	$salidaHTML .= "</div>";


	$objResponse->removeScript("js/gallery.js");

	$objResponse->removeCSS("css/gallery.css");

	// Eliminamos la cache generada por el jQuery

	$objResponse->jquery->each("body>:not(.tail)","function(){\$(this).remove();}");

	$objResponse->jquery->each("body>:not(#tail)","function(){\$(this).remove();}");

	// le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML

	$objResponse->assign("contenido", "innerHTML", $salidaHTML);

	$objResponse->includeScriptOnce("js/gallery.js");

	$objResponse->includeCSS("css/gallery.css");

	// Funciones jquery

	$objResponse->jquery->progressbar("#progressbar","{value:100}");

	$objResponse->jquery->addClass("#trash","ui-corner-all");


	$objResponse->jquery->hover("#dialogo_link","function() { \$(this).addClass('ui-state-hover'); },function() { \$(this).removeClass('ui-state-hover'); }");

	$objResponse->jquery->addClass("#dialogo_link","dialogo_link");

	// Funciones jQuery para el dialogo de selecionar la imagen

	$objResponse->jquery->click("#dialogo_link","function(){\$('#dialogo').dialog('open');return false;}");
```

```
        $objResponse->jquery->dialog("#dialogo","{autoOpen: false, bgiframe: true,
resizable:false,closeOnEscape:false,height:220,width:420,modal:true,buttons:{'Cerrar':func
tion() {\$(this).dialog('close');}}}");


    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnModificarGaleria");


/**
 * Funcion encargada de cargar el formulario de eliminar la informacion de un
servicio registrado en la Base de Datos
 * @return HTTPReponse
 * @param array $aFormulario - Parametros enviados por el formulario
 */
function fnEliminarGaleria($aFormulario){
    // declaramos un objeto tipo respuesta (para las requeste que vendran.)
    $objResponse = new xajaxResponse();
    if($aFormulario['id']==0){
        $salidaHTML = "<font color='red'>Debes selecionar una galeria de arte.</font>";
        $objResponse->assign("validacion_selector", "innerHTML", $salidaHTML);
    }
    else{
        $objObraArte = new ObraArte($aFormulario['id']);
        $iAccion = $aFormulario['tipo_accion'];
```

```
        $salidaHTML  = "<div class='contentheading'>Eliminar galeria de arte</div>";

        $salidaHTML .= "<div> </div>";

        $salidaHTML .= "<div id='subcontenido'>";

        $salidaHTML .= "<form id='form' onsubmit='return false;'>";

        $salidaHTML .= "<font color='red'>Esta Realmente seguro de eliminar la obra de arte <b>\"".$objObraArte->getNombre()."\"</b>.<br />";

        $salidaHTML .= "Con &eacute;sta acci&oacute;n se borrara la informaci&oacute;n de la obra de arte en el sitio web, junto con las im&aacute;genes anexadas al proyecto.</font><p> </p>";


        $salidaHTML .= "<input type='hidden' name='id' value=\"".$aFormulario['id']."\">";

        $salidaHTML .= "<input type='button' class='button' value='Si' onclick=\"xajax_fnAccionBDProyecto(xajax.getFormValues('form'),3); return false;\">    ";

        $salidaHTML .= "<input type='button' class='button' value='No' onclick=\"xajax_fnSeleccionarGaleria(".$aFormulario['tipo_accion'].");\">";

        $salidaHTML .= "</form>";

        $salidaHTML .= "</div>";


        // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable salidaHTML

        $objResponse->assign("contenido", "innerHTML", $salidaHTML);

    }

    // la asignacion anterior es la que se va a devolver para aplicar los cambios.

    return $objResponse;

}

$xajax->register(XAJAX_FUNCTION,"fnEliminarGaleria");
```

```
/**
 * Funcion encargada de ejecutar las acciones de Agregar, Modificar o Eliminar la informacion de un Proyecto Pedagogico en la Base de Datos
 * @return HTTPReponse
 * @param array $aFormulario - Parametros enviados por el formulario
 * @param int $iTipoAccion - Accion a ejecutar en la base de datos 1=Agregar; 2=Modificar; 3=Eliminar
 */
function fnAccionBDProyecto($aFormulario, $iTipoAccion, $aFCKEditor = null, $aOrden = null){
    // Declaramos los objetos respectivos
    $objGenerico = new Generico();
    $objObraArte = new ObraArte();
    $objUsuario  = new Usuario();
    $objImagen   = new Imagen();
    //$objGaleria  = new Galeria();
    // Declaramos un objeto tipo respuesta (para las requeste que vendran.
    $objResponse = new xajaxResponse();
    // Variable de control
    $bValidacion = array();
    $bValidacion[] = true;
    if(($iTipoAccion == 1)or($iTipoAccion == 2)){
        // Rescatamos las variables enviadas por el formulario
        $sNombre        = $objGenerico->Caracter(trim($aFormulario['nombre']));
      $sTecnica       = $aFormulario['tecnica'];
      $sDimension     = $objGenerico->Caracter($aFormulario['dimension']);
      $sPrecio        = $aFormulario['precio'];
```

```
        $sTiempo        = $objGenerico->Caracter($aFormulario['elaboracion']);
        $sEstado        = $aFormulario['estado'];
          $sDescripcion  = $aFCKEditor['descripcion'];


          // Validaremos cada una de las respectivas variables
          $bValidacion[] = $objResponse->validation-
>text($sNombre,"nombre","validar_nombre");
        $bValidacion[] = $objResponse->validation-
>select($sTecnica,"tecnica","validar_tecnica");
        $bValidacion[] = $objResponse->validation-
>text($sDimension,"dimension","validar_dimension");
        $bValidacion[] = $objResponse->validation-
>number($sPrecio,"precio","validar_precio");
        $bValidacion[] = $objResponse->validation-
>text($sTiempo,"elaboracion","validar_elaboracion");
        $bValidacion[] = $objResponse->validation-
>select($sEstado,"estado","validar_estado");
          $bValidacion[] = $objResponse->validation-
>empt($sDescripcion,"descripcion_frame","validar_descripcion");
    }
    if(!in_array(false,$bValidacion)){
        $objUsuario->getUsuario("LOGIN='".$_SESSION['usuario']."'");
        // Ejecutamos las respectivas acciones con la base de datos
        switch($iTipoAccion){
          case 1:
            $sTemporal = "";
            foreach ($aOrden as $sOrden){
              $sTemporal .= $sOrden." - ";
```

```
            }
            //$objResponse->script("alert('".$sTemporal."')");
            // Asignamos los valores al objeto Servicio
            $objObraArte->setNombre($sNombre);
            $objObraArte->setTecnica($sTecnica);
            $objObraArte->setDimension($sDimension);
            $objObraArte->setPrecio($sPrecio);
            $objObraArte->setTiempoElaboracion($sTiempo);
            if($sEstado==1)$sEstado = "Vendido";
            if($sEstado==2)$sEstado = "Disponible";
            $objObraArte->setEstado($sEstado);
            $objObraArte->setDescripcion($sDescripcion);

        // Registramos la Galeria de Arte

        $bRegistro = $objObraArte->registrarObraArte();

            // Creamos los directorios donde se van almacenar la galeria de imagenes del
projecto.
            mkdir($objGenerico->serverpath()."/gallery/".$objObraArte->getId());
            mkdir($objGenerico->serverpath()."/gallery/".$objObraArte-
>getId()."/thumbnail");
            // Abrimos el archivo de configuracion del projecto para objetner la informacion
de las imagenes
            // para almacenarlas en la base de datos y copiarlas en su respectivo lugar.
            $sContenidoArchivo = array();
            if(file_exists("gallery/tmp/configuracion.dat")){
```

```
    $archivoImagen = fopen("gallery/tmp/configuracion.dat", "r");
    while($sLine = fgets($archivoImagen)){
        $sContenidoArchivo[] = $sLine;
    }
    fclose($archivoImagen);
}

if(count($sContenidoArchivo)>0){
    foreach($sContenidoArchivo as $aImage){
        // Formato array
        // 1||adDR9R4E.png||Proyecto midas||Esto es un proyecto muy...||250,568
        $aImage = explode("||", $aImage);
        if($aImage[0]==1){
            // Introducimos la infomracion dentro de la base de datos
            $objImagen->setTamanio($aImage[2]);
            $objImagen->setPath($aImage[1]);
            // Obtenemos el orden de la imagen
            $iContador = 1;
            $iOrden = 1;
            foreach ($aOrden as $sOrden){
                if ($sOrden == $aImage[1]) $iOrden = $iContador;
                $iContador ++;
            }
            $objImagen->setOrden($iOrden);
            $objImagen->setObraArte($objObraArte->getId());
```

```
                    // Registramos la informacion de la imagen
                    $objImagen->registrarImagen();
                    fnCopiarArchivo($objGenerico->serverpath()."/gallery/tmp/".$aImage[1],
$objGenerico->serverpath()."/gallery/".$objObraArte->getId()."/".$aImage[1]);
                    fnCopiarArchivo($objGenerico-
>serverpath()."/gallery/tmp/thumbnail/".$aImage[1], $objGenerico-
>serverpath()."/gallery/".$objObraArte->getId()."/thumbnail/".$aImage[1]);
                }
            }
            fnBorrarDirectorio($objGenerico->serverpath()."/gallery/tmp/");
            // Creamos los directorios donde se van almacenar la galeria de imagenes
temporales
            mkdir($objGenerico->serverpath()."/gallery/tmp");
            mkdir($objGenerico->serverpath()."/gallery/tmp/thumbnail");
        }

    $salidaHTMLtemp = "<p>Obra de Arte registrado correctamente.</p>";
    break;
    case 2:
        $sTemporal = "";
        foreach ($aOrden as $sOrden){
            $sTemporal .= $sOrden." - ";
        }
        $objObraArte->__construct($aFormulario['id']);
        // Asignamos los valores al objeto Servicio
    $objObraArte->setNombre($sNombre);
        $objObraArte->setTecnica($sTecnica);
```

```
            $objObraArte->setDimension($sDimension);
            $objObraArte->setPrecio($sPrecio);
            $objObraArte->setTiempoElaboracion($sTiempo);
            if($sEstado==1)$sEstado = "Vendido";
            if($sEstado==2)$sEstado = "Disponible";
            $objObraArte->setEstado($sEstado);
            $objObraArte->setDescripcion($sDescripcion);
        // Registramos el Projecto
        $bRegistro = $objObraArte->actualizarObraArte();

            // Elimamos el ddirectorio del proyecto, para volver a crearlo
            fnBorrarDirectorio($objGenerico->serverpath()."/gallery/".$objObraArte->getId());
            // Creamos los directorios donde se van almacenar la galeria de imagenes del
projecto.
            mkdir($objGenerico->serverpath()."/gallery/".$objObraArte->getId());
            mkdir($objGenerico->serverpath()."/gallery/".$objObraArte-
>getId()."/thumbnail");
            // Abrimos el archivo de configuracion del projecto para objetner la informacion
de las imagenes
            // para almacenarlas en la base de datos y copiarlas en su respectivo lugar.
            $sContenidoArchivo = array();
            if(file_exists("gallery/tmp/configuracion.dat")){
               $archivoImagen = fopen("gallery/tmp/configuracion.dat", "r");
               while($sLine = fgets($archivoImagen)){
                  $sContenidoArchivo[] = $sLine;
               }
               fclose($archivoImagen);
```

```
}

if(count($sContenidoArchivo)>0){
    foreach($sContenidoArchivo as $aImage){
        // Formato array
        // 1||adDR9R4E.png||250,568
        $aImage = explode("||", $aImage);
        $aRegistros = $objImagen->getImagenAll();
        if($aImage[0]==1){
            // Cargamos la informacion de la imagen de auerdo al PATH en caso de
existir
            foreach ($aRegistros as $aRegistro){
                if ($aRegistro[3] == $aImage[1]) {
                    $objImagen->getImagen("PATH='".$aImage[1]."'");
                    $objImagen->eliminarImagen();
                }
            }
            // Introducimos la infomracion dentro de la base de datos
            $objImagen->setTamanio($aImage[2]);
            $objImagen->setPath($aImage[1]);
            // Obtenemos el orden de la imagen
            $iContador = 1;
            $iOrden = 1;
            foreach ($aOrden as $sOrden){
                if ($sOrden == $aImage[1]) $iOrden = $iContador;
                $iContador ++;
```

```
                    }

                    $objImagen->setOrden($iOrden);

                    $objImagen->setObraArte($objObraArte->getId());

                    // Registramos la informacion de la imagen

                    $objImagen->registrarImagen();

                    fnCopiarArchivo($objGenerico->serverpath()."/gallery/tmp/".$aImage[1],
$objGenerico->serverpath()."/gallery/".$objObraArte->getId()."/".$aImage[1]);

                    fnCopiarArchivo($objGenerico-
>serverpath()."/gallery/tmp/thumbnail/".$aImage[1], $objGenerico-
>serverpath()."/gallery/".$objObraArte->getId()."/thumbnail/".$aImage[1]);

                }

                elseif($aImage[0]==2){

                    // Cargamos la informacion de la imagen de auerdo al PATH

                    $objImagen->getImagen("PATH='".$aImage[1]."'");

                    // Eliminamos la informacion de la base de datos

                    $objImagen->eliminarImagen();

                }

            }

            fnBorrarDirectorio($objGenerico->serverpath()."/gallery/tmp/");

            // Creamos los directorios donde se van almacenar la galeria de imagenes
temporales

            mkdir($objGenerico->serverpath()."/gallery/tmp");

            mkdir($objGenerico->serverpath()."/gallery/tmp/thumbnail");

        }


    $salidaHTMLtemp = "<p>Galeria de arte modificada correctamente.</p>";

    break;
```

```
            case 3:
                // Iniciamos el objeto Proyecto
            $objObraArte->__construct($aFormulario['id']);
            // Eliminamos el proyecto
            $bRegistro = $objObraArte->eliminarObraArte();

                // Eliminamos el directorio del proyecto
                fnBorrarDirectorio($objGenerico->serverpath()."/gallery/".$aFormulario['id']."/");

            $salidaHTMLtemp = "<p>Obra de arte eliminada correctamente.</p>";
            break;
        }
          // Si fallo la ejecucion del comando imprimimos un error.
        if (!$bRegistro) $salidaHTML = "Error. Imposible realizar la operacion.<br><br>";
            // Si no notificamos que todo fue un exito
        else $salidaHTML = $salidaHTMLtemp;

        $salidaHTML .= "<input type='button' class='button' value='Aceptar'
onclick=\"xajax_fnMostrarGaleria()\";'/>";
          $salidaHTML .= "</div>";
          // le decimos que debe cambiar el div 'mainbody' por el contenido de la variable
salidaHTML
          $objResponse->assign("subcontenido", "innerHTML", $salidaHTML);
    }
    // la asignacion anterior es la que se va a devolver para aplicar los cambios.
    return $objResponse;
```

```
}
$xajax->register(XAJAX_FUNCTION,"fnAccionBDProyecto");
```

Formularios de subida de archivos

```
/**
 * Funcion generadora de un Sid Randomico
 * @return
 */
function getSid(){
	$objResponse = new xajaxResponse();
	$sid = md5(uniqid(rand()).date("YmdHis"));
	$objResponse->script("sid='".$sid."';");
	return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"getSid");

/**
 * Funcion encargada de cargar y modificar la lista de los tipos de archivos aceptados
 * @param string $sNuevoFileTypes
 */
function fnCargarConfiguracionUpload($sNewFileTypes=null){
	$objResponse = new xajaxResponse();
	global $tmpDirInCGI,$uploadDir,$cgiPath,$max_file,$maxFiles,$autoFirst,$cgiDebug,$uploadSpeed,$fileTypes,$extMode,$blankHTMLPath,$showMax,$funcionUpload;
```

```
// Creamos la contenido del script de configuracion de la subida de archivos
$jsOut = "";
if ($tmpDirInCGI != 1)  $jsOut .=  "uploadDir='".$uploadDir."';";
$jsOut .= "cgiPath='".$cgiPath."';";
$jsOut .= "maxFile=".$max_file.";";
$jsOut .= "maxNumFiles=".$maxFiles.";";
$jsOut .= "autoFirst=".$autoFirst.";";
$jsOut .= "cgiDebug=".$cgiDebug.";";
$jsOut .= "uploadSpeed=".$uploadSpeed.";";
if(null !== $sNewFileTypes) $jsOut .= "filetypes='".$sNewFileTypes."';";
else $jsOut .= "filetypes='".$fileTypes."';";
$jsOut .= "extMode=".$extMode.";";
$jsOut .= "blankHTMLPath='".$blankHTMLPath."';";
$jsOut .= "showMax=".$showMax.";";

// Verificamos si el archivo existe para eliminarlo y crear uno nuevo
if(file_exists("modules/upload/upload_config.js")) {
 unlink("modules/upload/upload_config.js");
}
$nf = fopen("modules/upload/upload_config.js", "w");
@fwrite($nf, $jsOut);
@fclose($nf);

$objResponse->removeScript("modules/upload/upload_config.js");
$objResponse->includeScriptOnce("modules/upload/upload_config.js");
```

```
	return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"fnCargarConfiguracionUpload");


/**
 * Funcion encargada de insertar el formulario para subir el archivo
 * @return HTTPResponse
 * @param string $max_file
 */
function incFormulario($sFuncionUpload,$iNum=null){
	$objResponse = new xajaxResponse();
	global $max_file,$blankHTMLPath;//,$parametro;
	$sFormulario  = "<div class='uld'>";
	$sFormulario .= "<form id='file_upload' enctype='multipart/form-data' action='' method='post'>";
	$sFormulario .= "<div id='file_inputs' style='float:left;'>";
	$sFormulario .= "<!--hidden file inputs will be dynamically inserted here-->";
	$sFormulario .= "</div>";
	$sFormulario .= "<div style='float:left;clear:none;'>";
	$sFormulario .= "<input type='button' id='uldCancel' name='uldCancel' value='Cancel' onclick='cancel();' />";
	$sFormulario .= "</div>";
	$sFormulario .= "<div style='float:left;clear:none;'>";
	$sFormulario .= "<input type='button' id='uldSubmit' name='uldSubmit' value='Send' onclick='postIt();' disabled='disabled' />";
	$sFormulario .= "</div>";
```

```
$sFormulario .= "<div id='moreinfo1' class='more'>Tama&ntilde;o maximo del archivo es ".format_size($max_file).".</div>";

$sFormulario .= "<div id='moreinfo2' class='more'>NOTA: Para subir m&aacute;s de un archivo al mismo tiempo, construir la lista antes de darle click en el boton Send.</div>";

$sFormulario .= "<div id='okstatus' class='notice'  style='display:none;'></div>";

$sFormulario .= "<br /><br />";

$sFormulario .= "<div id='progress_bar'>";

$sFormulario .= "<div id='load_bar'></div>";

$sFormulario .= "</div>";

$sFormulario .= "<div id='loadtext'></div>";

$sFormulario .= "<ul id='file_list' style='list-style-type:none;clear:left;'>";

$sFormulario .= "<li></li>";

$sFormulario .= "<!--list of filenames  will be dynamically inserted here-->";

$sFormulario .= "</ul>";

$sFormulario .= "<iframe src='$blankHTMLPath' name='destination0' id='destination0' height='0' width='0' frameborder='0'></iframe>";

$sFormulario .= "</form>";

$sFormulario .= "</div>";

$sContenidoFileUpload = "|".$sFuncionUpload;

$objResponse->assign("incFormulario", "innerHTML", $sFormulario);

if(null !== $iNum)

 $sContenidoFileUpload .= "||".$iNum;

else

 $sContenidoFileUpload .= "|| ";

$objResponse->jquery->css("#moreinfo1,#moreinfo2","display","none");

$objResponse->jquery->css("#moreinfo1,#moreinfo2","clear","left");

$objResponse->jquery->css("#moreinfo1,#moreinfo2","font-size","8pt");
```

```
	$objResponse->jquery->css("#loadtext","font-size","8pt");
	$objResponse->script("startOver()");
	// Generamos el archivo que contiene la informacion de la funcion y los parametros de la funcion a ejecutar
	// una vez finalice la subida de archivo
	$nf = fopen("modules/upload/function_upload.js", "w");
	@fwrite($nf, $sContenidoFileUpload);
	@fclose($nf);

	return $objResponse;
}
$xajax->register(XAJAX_FUNCTION,"incFormulario");

/**
 * Funcion encargada de gestionar las diferentes funciones para la subida de archivos
 * @return HTTPResponse
 * @param string $sid
 * @param int $recursion
 */
function uploadHandler($sid,$recursion){
	global $startDelay,$refreshDelay,$cgiDebug,$uploadDir;//,$parametro;
	$objResponse = new xajaxResponse();
	//$objResponse->script("alert('".$funcionUpload."')");
	$exts = array("_flength","_postdata","_ err","_signal","_qstring");
	if (!empty($sid)){
```

```
//$objResponse->script("alert('".$uploadDir.$sid."')");

// Proceso ha sido cancelado por el usuario

if($recursion<0){

        foreach($exts as $ext) {

                if(file_exists($uploadDir.$sid.$ext)) {

                        unlink($uploadDir.$sid.$ext);

                }

        }

        $objResponse->assign("okstatus","innerHTML","<div class='ui-state-error
ui-corner-all' style='padding: 0 .7em;'><p><br /><span class='ui-icon ui-icon-alert'
style='float: left; margin-right: .3em;'></span><strong>Alerta: </strong>Subida ha sido
cancelada.</p></div>");

        $objResponse->jquery->show("#okstatus");

        $objResponse->assign("progress_bar", "innerHTML", "<div id='load_bar'
style='height:10px;'></div>");

        $objResponse->jquery->progressbar("#load_bar","{value: 0}");

        $objResponse->jquery->hide("#progress_bar");

        $objResponse->jquery->hide("#loadtext");

        $objResponse->script("startOver();");

        return $objResponse;

}

// Error encontrado, no se logro subir el archivo

elseif (file_exists ( $uploadDir.$sid."_err" ) ){

        $mes = file_get_contents($uploadDir.$sid."_err");

        $objResponse->assign("okstatus","innerHTML","<div class='ui-state-error
ui-corner-all' style='padding: 0 .7em;'><p><br /><span class='ui-icon ui-icon-alert'
style='float: left; margin-right: .3em;'></span><strong>Alerta: </strong>Fallo!
$mes.</p></div>");
```

```
                $objResponse->jquery->show("#okstatus");
                $objResponse->assign("progress_bar", "innerHTML", "<div id='load_bar'
style='height:10px;'></div>");
                $objResponse->jquery->progressbar("#load_bar","{value: 0}");
                $objResponse->jquery->hide("#progress_bar");
                $objResponse->jquery->hide("#loadtext");
                $objResponse->script("startOver();");
                return $objResponse;
        }
        // Proceso terminado correctamente
        elseif ( file_exists ( $uploadDir.$sid."_signal" ) ){
                // Colocamos la informacion del archivo dentro de un array
                $qstr = file_get_contents($uploadDir.$sid."_qstring");
                parse_str($qstr);
                foreach($exts as $ext) {
                        if(file_exists($uploadDir.$sid.$ext)) {
                                unlink($uploadDir.$sid.$ext);
                        }
                }

                $nf = fopen("modules/upload/function_upload.js", "r");
                while($sLine = fgets($nf)){
                        $sFuncion = substr($sLine, strpos($sLine, "|")+1,strpos($sLine, "||")-
1);
                        $sParametro = substr($sLine, strpos($sLine, "||")+2);
                }
```

```
                @fclose($nf);

                if($sParametro!="")

                        $objResponse-
>script("xajax_$sFuncion('".fileAction($file)."','".$sParametro."')");

                else $objResponse->script("xajax_$sFuncion($file)");

                $objResponse->assign("okstatus","innerHTML","<div class='ui-state-
highlight ui-corner-all' style='margin-top: 20px; padding: 0 .7em;'><p><br /><span
class='ui-icon ui-icon-circle-check' style='float: left; margin-right: .3em;'></span>Archivo
subido correctamente.</p></div>");

                $objResponse->jquery->show("#okstatus");

                $objResponse->assign("progress_bar", "innerHTML", "<div id='load_bar'
style='height:10px;'></div>");

                $objResponse->jquery->progressbar("#load_bar","{value: 0}");

                $objResponse->jquery->hide("#progress_bar");

                $objResponse->jquery->hide("#loadtext");

                $objResponse->script("startOver();");

                return $objResponse;

        }

        // Proceso de subida de archivo en proceso

        elseif ( file_exists ( $uploadDir.$sid."_postdata" ) ) {

                usleep($refreshDelay);

                $total_size      = file_get_contents ( $uploadDir.$sid."_flength" );

                $loaded_size  = filesize ( $uploadDir.$sid."_postdata" );

                $percent_loaded= round ( $loaded_size / $total_size * 100 );

                $objResponse->assign("progress_bar", "innerHTML", "<div id='load_bar'
style='height:10px;'></div>");

                $objResponse->jquery->progressbar("#load_bar","{value:
$percent_loaded}");
```

```
                $objResponse->jquery->show("#progress_bar");

                $objResponse->assign("loadtext","innerHTML","Subiendo:
".format_size($loaded_size)."/".format_size($total_size));

                $objResponse->script("getProgress();");

                return $objResponse;

        }

        // No ha empezado la subida

        else {

                if ($recursion<5){ // WE WILL GIVE IT 5 LOOPS TO GET STARTED.

                        usleep($startDelay);

                        for ($i;strlen($dots)<$recursion; $dots.=".");

                        $objResponse->assign("loadtext","innerHTML","Preparando
subida.".$dots);

                        if ($cgiDebug){

                                if(file_exists ($uploadDir)) $objResponse-
>assign("okstatus","innerHTML","<div class='ui-state-error ui-corner-all' style='padding: 0
.7em;'><p><br /><span class='ui-icon ui-icon-info' style='float: left; margin-right:
.3em;'></span><strong>Alerta: </strong>Streaming ".$sid."_postdata.</p></div>");

                                else $objResponse->assign("okstatus","innerHTML","<div
class='ui-state-error ui-corner-all' style='padding: 0 .7em;'><p><br /><span class='ui-icon
ui-icon-alert' style='float: left; margin-right: .3em;'></span><strong>Alerta: </strong>PHP
no puede abrir el directorio temporal!! - (".$uploadDir.").</p></div>");

                                $objResponse->jquery->show("okstatus");

                        }

                        $objResponse->script("getProgress();");

                } else { // IF TOO MANY LOOPS, KICK OUT SCRIPT.

                        $objResponse->assign("loadtext","innerHTML","<div class='ui-state-
error ui-corner-all' style='padding: 0 .7em;'><p><br /><span class='ui-icon ui-icon-alert'
style='float: left; margin-right: .3em;'></span><strong>Alerta: </strong>Fallo al comenzar
la subida. Su navegador no soporta esta caracteristica.</p></div>");
```

```
                }

                return $objResponse;

         } // if file exists

        }

        else { //NO SID FOUND

         $objResponse->assign("okstatus","innerHTML","<div class='ui-state-error ui-corner-all' style='padding: 0 .7em;'><p><br /><span class='ui-icon ui-icon-alert' style='float: left; margin-right: .3em;'></span><strong>Alerta: </strong>Fallo!! Por favor refresca el navegador e intenta de nuevo.</p></div>");

         return $objResponse;

        }

}

$xajax->register(XAJAX_FUNCTION,"uploadHandler");



function fileAction($fileArr){

        $x=0;

        $infoArchivo = "";

        while (!empty($fileArr['name'][$x])) {

         $infoArchivo .=  $fileArr['tmp_name'][$x].",";

         $infoArchivo .=  $fileArr['name'][$x].",";

         $infoArchivo .=  $fileArr['size'][$x].",";

         $x++;

        }

        $infoArchivo = substr($infoArchivo, 0, strrpos($infoArchivo, ",")-1);

        return $infoArchivo;

}
```

Código de una de las clases creadas para todo el manejo de con la base de datos

```
/**
Class Noticia
*/
class Noticia {
	// Variables globales de mi clase Departamento
	private $sId;
	private $sTitulo;
	private $sDescripcion;
	private $sFecha;
	private $CUsuario;

	/**
	* Constructor Clase Noticia
	* @param int $iId[optional]
	*/
	public function __construct($iId = null) {
	 if(null !== $iId){
		// Creamos un objeto ConexionSql
		$mysqlConector=new ConexionSql();
		// Conectamos al servidor MySQL
		$bConectado = $mysqlConector->Conectar();

		if ($bConectado) {
```

```
                $sSqlQuery = "SELECT
TITULO,DESCRIPCION,FECHA,ID_USUARIO FROM noticia WHERE ID = '".$iId."'";
                $rResultQuery = mysql_query($sSqlQuery);
                $this->sId = $iId;
                $this->sTitulo = mysql_result($rResultQuery, 0, 0);
                $this->sDescripcion = mysql_result($rResultQuery, 0, 1);
                $this->sFecha = mysql_result($rResultQuery, 0, 2);
                $this->CUsuario = new Usuario(mysql_result($rResultQuery, 0, 3));
            }
            // Desconectamos del servidor MySQL
            $mysqlConector->Desconectar();
        }
    }
    /**
    * Funcion que retorna Id de la Clase Noticia
    * @return string sId
    */
    public function getId() {
     return $this->sId;
    }
    /**
    * Funcion que asigna Id a la Clase Noticia
    */
    public function setId($sId) {
     $this->sId = $sId;
    }
```

```
/**
* Funcion que retorna Titulo de la Clase Noticia
* @return string sTitulo
*/
public function getTitulo() {
 return $this->sTitulo;
}

/**
* Funcion que asigna Titulo a la Clase Noticia
*/
public function setTitulo($sTitulo) {
 $this->sTitulo = $sTitulo;
}

/**
* Funcion que retorna Descripcion de la Clase Noticia
* @return string sDescripcion
*/
public function getDescripcion() {
 return $this->sDescripcion;
}

/**
```

```
* Funcion que asigna Descripcion a la Clase Noticia
*/
public function setDescripcion($sDescripcion) {
 $this->sDescripcion = $sDescripcion;
}

/**
* Funcion que retorna Fecha de la Clase Noticia
* @return string sFecha
*/
public function getFecha() {
 return $this->sFecha;
}

/**
* Funcion que asigna Fecha a la Clase Noticia
*/
public function setFecha($sFecha) {
 $this->sFecha = $sFecha;
}

/**
* Funcion que retorna Usuario de la Clase Noticia
* @return object Usuario
*/
```

```
public function getUsuario() {
 return $this->CUsuario;
}

/**
* Funcion que asigna Usuario a la Clase Noticia
*/
public function setUsuario($UsuarioId) {
 $this->CUsuario = new Usuario($UsuarioId);
}

/**
*Funcion que asigna los registros de la tabla noticia a las propiedades de la Clase Noticia
*/
public function getNoticia($sCondicion = null) {
 // Creamos un objeto ConexionSql
 $mysqlConector=new ConexionSql();
 // Conectamos al servidor MySQL
 $bConectado = $mysqlConector->Conectar();
 if ($bConectado) {
        $sSqlQuery = "SELECT ID,TITULO,DESCRIPCION,FECHA,ID_USUARIO FROM noticia";
        if(null !== $sCondicion)
            $sSqlQuery .= " WHERE ".$sCondicion.";";
        $rResultQuery = mysql_query($sSqlQuery);
```

```
            // Desconectamos del servidor MySQL
            $mysqlConector->Desconectar();

            if(mysql_numrows($rResultQuery)>0){;
                    $this->sId = mysql_result($rResultQuery, 0, 0);
                    $this->sTitulo = mysql_result($rResultQuery, 0, 1);
                    $this->sDescripcion = mysql_result($rResultQuery, 0, 2);
                    $this->sFecha = mysql_result($rResultQuery, 0, 3);
                    $this->CUsuario = mysql_result($rResultQuery, 0, 4);
                    return true;
            }
 }
 return false;
}

/**
* Funcion que me retorna todos los registros de la tabla noticia
* @return Array
*/
public function getNoticiaAll($sCondicion = null) {
 // Creamos un objeto ConexionSql
 $mysqlConector=new ConexionSql();
 // Conectamos al servidor MySQL
 $bConectado = $mysqlConector->Conectar();
```

```
        if ($bConectado) {
                $sSqlQuery = "SELECT ID,TITULO,DESCRIPCION,FECHA,ID_USUARIO
FROM noticia";
                 if(null !== $sCondicion)
                        $sSqlQuery .= " WHERE ".$sCondicion.";";
                $rResultQuery = mysql_query($sSqlQuery);
                $aRegistros = array();

                while ($sTemp = mysql_fetch_array($rResultQuery, MYSQL_NUM)) {
                        $sTemp[4] = new Usuario($sTemp[4]);
                        $aRegistros[] = $sTemp;
                }
                // Desconectamos del servidor MySQL
                $mysqlConector->Desconectar();
                return $aRegistros;
         }
        }

        /**
        * Funcion encargada de registrar Noticia
        * @return boolean
        */
        public function registrarNoticia() {
         // Creamos un objeto ConexionSql
         $mysqlConector=new ConexionSql();
```

```
        // Conectamos al servidor MySQL
        $bConectado = $mysqlConector->Conectar();
        if ($bConectado) {
                $sSqlQuery = "INSERT INTO
noticia(TITULO,DESCRIPCION,FECHA,ID_USUARIO) VALUES('".$this-
>getTitulo()."','".$this->getDescripcion()."','".$this->getFecha()."','".$this->getUsuario()-
>getId()."')";
                mysql_query($sSqlQuery);
                // Desconectamos del servidor MySQL
                $mysqlConector->Desconectar();
                return true;
        }
        return false;
       }

       /**
       * Funcion encargada de actualizar Noticia
       * @return boolean
       */
       public function actualizarNoticia() {
        // Creamos un objeto ConexionSql
        $mysqlConector=new ConexionSql();
        // Conectamos al servidor MySQL
        $bConectado = $mysqlConector->Conectar();
        if ($bConectado) {
        $sSqlQuery = "UPDATE noticia SET TITULO='".$this-
>getTitulo()."',DESCRIPCION='".$this->getDescripcion()."',FECHA='".$this-
```

```
>getFecha().'",ID_USUARIO="'.$this->getUsuario()->getId().'" WHERE ID="'.$this->getId().'"';
                mysql_query($sSqlQuery);
                // Desconectamos del servidor MySQL
                $mysqlConector->Desconectar();
                return true;
        }
        return false;
    }
    /**
    * Funcion encargada de eliminar Noticia
     */
    public function eliminarNoticia() {
     // Creamos un objeto ConexionSql
     $mysqlConector=new ConexionSql();
     // Conectamos al servidor MySQL
     $bConectado = $mysqlConector->Conectar();
     if ($bConectado) {
                $sSqlQuery = "DELETE FROM noticia WHERE ID="'.$this->getId().'"';
                mysql_query($sSqlQuery);
                $mysqlConector->Desconectar();
                return true;
        }
        return false;
    }
}
```