UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO-CAMPUS SUR

CARRERA DE INGENIERÍA DE SISTEMAS

MENCIÓN TELEMÁTICA

"DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA DE GESTIÓN DE DEPÓSITOS BANCARIOS PARA LA ADQUISCIÓN DE BIENES INMUEBLES DE LA COMPAÑÍA DE PROMOCIÓN Y DESARROLLO HUMANO Y SOCIAL COOPUEBLO"

TESIS PREVIA A LA OBTENCIÓN DEL TITULO DE INGENIERO DE SISTEMAS

ANDRÉS FERNANDO NARVÁEZ CADENA

EDWIN GONZALO GARCÉS VELARDE

DIRECTOR: Ing. Rafael Jaya

Quito, Noviembre 2012

DECLARACIÓN

Nosotros, Andrés Fernando Narváez Cadena y Edwin Gonzalo Garcés Velarde, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Universidad Politécnica Salesiana, según lo establecido por la ley de Propiedad Intelectual, por su reglamento y por la normativa institucional vigente.

_____ _____

Andrés Fernando Narváez Cadena Edwin Gonzalo Garcés Velarde

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Andrés Fernando Narváez Cadena y Edwin Gonzalo Garcés Velarde.

_____

Ing.Rafael Jaya

Director de Tesis

DEDICATORIA

El presente trabajo se lo dedico a mis padres, a mi mamá Nelly Cadena, que con su apoyo incondicional me dio ánimo para la culminación de este proyecto.

Para mi padre Fernando Narváez, que con sus consejos me supo encaminar por el sendero del bien y así ser una persona de buenos principios.

También va dedicado para mi esposa y mis hijas, a mi esposa que ha sido el pilar para no desmayar y poder finalizar con mucho entusiasmo este peldaño de mi vida y en cuanto a mis hijas que son el motor de mi existencia para que se sientan orgullosos de su padre.

Andrés Fernando Narváez Cadena

AGRADECIMIENTO

Agradezco principalmente a Dios por darme la oportunidad de llegar a esta etapa de mi vida, y contar con las personas más importantes en los momentos más difíciles de mi vida, quienes me han brindado su apoyo incondicional y la fuerza para llegar a cumplir mí meta.

Agradecer de todo corazón a todas las personas que con su voz de aliento y ánimo me supieron dar excelentes consejos para realizar este proyecto de tesis.

Para todos ellos mil gracias

Andrés Fernando Narváez Cadena

DEDICATORIA

Quiero dedicar este trabajo a una persona muy especial que lastimosamente tuvo que partir antes de tiempo y no pudo estar conmigo estos momentos, “mi madre”. También una dedicatoria especial a toda mi familia que siempre estuvo conmigo en los momentos más difíciles de mi vida y de mi carrera universitaria.

Edwin Gonzalo Garcés Velarde

AGRADECIMIENTO

Agradezco de todo corazón a toda mi familia en especial a mis hermanos que me ayudaron en todo momento y durante toda mi carrera universitaria. Debemos mencionar también a los empleados y personal administrativo de la compañía de promoción y desarrollo humano y social Coopueblo que nos dieron su ayuda en todo momento. Muchas gracias a todos.

Edwin Gonzalo Garcés Velarde

PRESENTACIÓN

Para la descripción del proyecto. A continuación se realiza una breve introducción de cada uno de los capítulos que están contenidos en este documento.

CAPÍTULO I.

En este capítulo, se muestra información concerniente a la Compañía de Promoción y Desarrollo Humano y Social Coopueblo. La información mostrada en este capítulo ha sido proporcionada por la propia compañía a los desarrolladores del proyecto.

CAPÍTULO II

En el capítulo dos, se describe las tecnologías con las cuales se pretende desarrollar el sistema, se presentan conceptos de las diferentes tecnologías y las herramientas que serán utilizadas para el desarrollo del proyecto.

CAPÍTULO III

Aquí se trata sobre los requerimientos que tiene la compañía para la cual se desarrolla el proyecto, también se empieza con el sistema que luego será desarrollado.

CAPÍTULO IV

En el capitulo 4, se presenta el desarrollo íntegro del sistema, el código fuente más importante del software, así como las diferentes capas en las cuales se basa el diseño, la lógica del negocio y los datos.

CAPÍTULO V

En el capitulo 5, el sistema es sometido a pruebas y se muestran los resultados esperados.

CAPÍTULO VI

Finalmente, se incluyen las conclusiones del trabajo realizado así como las recomendaciones.

RESUMEN

El presente proyecto describe la implementación de un sistema de gestión de depósitos bancarios para la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, la cual fue creada con el objetivo de ayudar a personas de escasos recursos económicos a adquirir un bien inmueble, para ello la persona interesada deberá convertirse en socio de esta institución.

El software fue creado con el fin de ayudar a las personas que trabajan en la empresa, a llevar un control adecuado de las cuentas de los clientes, para ello el software cuenta con distintos módulos de administración dependiendo del tipo de usuario. Para la realización de este proyecto, primero se obtuvo una recopilación de la información de la compañía, con a fin de analizar las necesidades de los empleados de esta institución, y así realizar el diseño del sistema.

Posteriormente se fueron creando módulos de software funcionales, los mismos que luego de ser sometidos a pruebas, mostraron los resultados esperados tanto por los empleados de la compañía, así como por el equipo de desarrollo del sistema.

CONTENIDO

TÍTULO **PÁGINA.**

CAPÍTULO III

ANÁLISIS Y DISEÑO DEL SISTEMA

CAPÍTULO IV

DESARROLLO E IMPLEMENTACIÓN

CAPITULO V

PRUEBAS

ÍNDICE DE FIGURAS

TÍTULO **PÁGINA**

ÍNDICE DE TABLAS

TÍTULO **PÁGINA**

CAPÍTULO I

GENERALIDADES:

1. INFORMACIÓN DE LA COMPAÑÍA DE PROMOCIÓN Y DESARROLLO HUMANO Y SOCIAL COOPUEBLO.

La Compañía de Promoción y Desarrollo Humano y Social Coopueblo, surge de la necesidad de las personas de escasos recursos económicos ya que estas personas necesitaban una entidad en la cual depositen no solo sus ahorros para luego adquirir un bien inmueble sino que les brinde la confianza suficiente para que los socios sigan aportando, sigan ingresando nuevos cuenta ahorristas y así esta compañía crezca y se consolide para el beneficio de la comunidad.

Esta compañía tiene el aporte y ayuda de diversos organismos, también existen donaciones como por ejemplo: computadoras, sillas, mesas, etc.

La calidad del talento humano que labora en la compañía son personas preocupadas por los clientes que pertenecen a esta entidad.

La Compañía de Promoción y Desarrollo Humano y Social Coopueblo, con su vocación solidaria, puede considerarse esencialmente como un pueblo cooperativo. Todos sus cuenta ahorristas, y sus colaboradores exteriores, participan de un proyecto productivo común, con aspectos de índole productivos hacia los socios dándoles una confianza para que su dinero este bien resguardado para la adquisición de su bien inmueble. Lo novedoso es que entre cada una de las actividades es tal que funciona como un organismo especializado que realizan funciones indispensables.

La ventaja de considerar a Coopueblo como una unidad productiva coordinada centralmente por un grupo de gestores reside en que los flujos de capital generado por los socios son parte integral del capital del sistema productivo. De esta manera, todo el capital, tanto el líquido como los activos físicos y humanos quedan a disposición de los gestores. Es importante diseñar un sistema que permita el ingreso y sobre todo de llevar un control del movimiento de ingresos y egresos del dinero de sus cuenta-ahorristas para que las personas que pertenecen a esta entidad se sientan partícipes de este proyecto.

Este sistema va a facilitar no solo la realización de balances anuales, sino incluso la planificación productiva y social en forma unificada. Además, esto permite hacer planes de inversiones de formular metas globales, con sus correspondientes cronogramas y prioridades. Los resultados anuales permiten discutir en forma unificada políticas para incrementar inversiones y captar más clientes. Obviamente que el funcionamiento de dicho sistema es clave en la distribución de activos e ingresos por lo cual se entiende que dicho esquema es orgánico y habrá de evolucionar en la medida que la gente de la Compañía de Promoción y Desarrollo Humano y Social Coopueblo continúe dando el apoyo, confianza manteniendo el perfil de los objetivos sociales a corto, mediano y largo plazo.

1.1 MISIÓN

Somos una compañía ecuatoriana de capital multisectorial, orientada al desarrollo rural e integral del país, dirigida preferentemente al agricultor, comerciante, artesano y al empresario-propietario de la micro, pequeña y mediana empresa.

1.2 VISIÓN

Ocupar en el área rural la posición de una entidad líder en el financiamiento del desarrollo, a través de los centros financieros, ofreciendo un conjunto de productos que satisfagan las necesidades de nuestros clientes; con rentabilidad justa y con función social.

1.3 OBJETIVOS

1.3.1 OBJETIVO GENERAL

- Implementar un sistema informático para la gestión de depósitos bancarios para la adquisición de bienes inmuebles de la Compañía de Promoción y Desarrollo Humano y Social Coopueblo.

1.3.2 OBJETIVOS ESPECÍFICOS

- Implementar un registro de inscripciones de socios para la futura creación de cuentas con sus respectivos depósitos.

- Implementar cuentas por cada socio y llevar un historial de sus respectivas transacciones para la obtención de su bien.

- Construir un registro del socio donde se encuentre todos los certificados bancarios para poder consolidar la información y hacer la entrega del inmueble.

- Desarrollar un módulo de transferencias internas para poder trasladar los depósitos realizados por un socio a otro cliente previo al mutuo acuerdo.

- Construir un sistema que sea de gran ayuda para el personal administrativo de la compañía, de tal manera que puedan llevar un control adecuado de las cuentas de los socios y sus respectivos depósitos.

- Proporcionar a la compañía un sistema informático de calidad, el cual ayude a detectar posibles duplicaciones en las cuentas y depósitos de los socios, para que así el personal administrativo pueda tomar las acciones que considere pertinentes.

1.4 JUSTIFICACIÓN

El presente proyecto se lo va a realizar porque es una necesidad urgente e indispensable para la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, por su forma de gestionar los depósitos de cada socio, y se convierte en relevante importancia para el área administrativa de la compañía. Así mismo el proyecto le brindará una solución rápida y oportuna a los problemas informáticos actuales de la compañía. Por lo que el sistema les proporcionará información de cómo se están realizando las transacciones, diaria y mensualmente, de esa manera podrán tomar dediciones oportunas y gestionar las cuentas y transacciones de los socios de manera adecuada.

1.5 ALCANCE

Para definir el alcance del proyecto se ha tomado en cuenta las necesidades informáticas de la compañía de promoción y desarrollo humano y social Coopueblo.

Módulo de Inscripción.- En este módulo se va a crear una cuenta para un nuevo socio con todos los datos que la compañía crea pertinentes.

- Número de cuenta.
- Ingreso de datos del cliente.
- Mantenimiento de datos del cliente.
- Ingreso de solicitud de apertura de cuentas.
- Impresiones de inscripciones.
- Creaciones de tarjetas bancarias.

Módulo de gestión de usuarios.- En este módulo se definirán los perfiles para los diferentes empleados de la compañía.

- Datos de usuario.
- Definición de usuarios.
- Perfiles de los usuarios, que permitirán crear cuentas de acuerdo al perfil (Gerente, contador, administrador, etc.).

Módulo de Registro de depósitos bancarios.- aquí se visualizará todas las transacciones que un socio ha realizado hasta el momento.

- Depósitos.
- Transferencias entre cuentas.
- Registro contable de transacciones del socio.

Módulo de retiro de socios de la cooperativa.- En este módulo se realizará el trámite pertinente cuando el socio decida no pertenecer a la compañía.

- Gestión de solicitud de retiro.
- Impresión de la solicitud.

1.6 DESCRIPCIÓN GENERAL DEL PROYECTO.

En este proyecto se va a implementar un sistema informático para la gestión de depósitos bancarios para la adquisición de bienes inmuebles de la Compañía de Promoción y Desarrollo Humano y Social Coopueblo.

Dicho sistema se lo va a realizar porque es una necesidad urgente e indispensable para la Compañía Coopueblo, por su forma de gestionar los depósitos de cada socio, y se convierte en relevante importancia para el área administrativa de la compañía. Así mismo el proyecto le brindará una solución rápida y oportuna a los problemas informáticos actuales de la compañía. Por lo que el sistema les proporcionará información de cómo se están realizando las transacciones, de esa manera podrán tomar dediciones oportunas en el ambiente en el que ellos se desenvuelven para definir estrategias en el ámbito que la cooperativa lo requiera.

Los administradores de la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, podrán obtener información rápida y detallada con reportes diarios de los depósitos realizados por los cuenta ahorristas. Una vez que un socio de la cooperativa ha completado el valor total para la adquisición de un bien, el sistema mostrará un mensaje que le indique al operador que el socio ha completado el valor requerido y que por lo tanto ya puede adquirir su bien inmueble.

Dado que la compañía presta ayuda a personas de escasos recursos, y estas pueden realizar depósitos cuando ellos lo consideren necesario. Sin importar el límite de tiempo ni la cantidad del abono realizado a su cuenta. El software registrará cada abono realizado por el cliente y llevará un registro de todos los depósitos, teniendo en cuenta aspectos como: fecha de depósito y cantidad.

CAPÍTULO II

MARCO TEÓRICO

En este capítulo se realizará una breve introducción a cerca de las tecnologías necesarias para el desarrollo del proyecto, así como también las herramientas que se van a utilizar para realizar el software.

Sin duda se puede encontrar mucha información a cerca de las tecnologías y aplicaciones para el desarrollo web, ya sea en medios electrónicos o en libros específicos a cerca de algún tema o autor. Lo que se va a presentar a continuación constituye una breve introducción a cerca de las tecnologías y herramientas con las cuales se pueden realizar una aplicación web, se tratará de dar una idea a cerca de lo que significan estos conceptos y como se van a utilizar para la ejecución del proyecto en cuestión. Algunas de las herramientas que se van a describir pueden no ser las últimas versiones lanzadas al mercado por las diferentes compañías fabricantes, el hecho de utilizar una versión de una herramienta u otra por parte de los desarrolladores, puede ser debido a que: los implicados en el desarrollo del proyecto de construcción de un sitio web están acostumbrados a trabajar con estas herramientas de desarrollo, se sienten cómodos con estas herramientas, y están familiarizados y acostumbrados a realizar trabajos utilizando estas aplicaciones.

2.1 DEFINICIONES PARA LAS APLICACIONES WEB

Una aplicación web se puede definir como un conjunto de tecnologías de software mediante las cuales el usuario puede interactuar con un servidor web a través de internet o de una intranet. Como por ejemplo una intranet de una empresa que tiene su propio servidor.

Para acceder a una aplicación web el cliente necesita de un navegador como por ejemplo (internet Explorer, Mozilla, opera, etc.), que interprete el código enviado por el servidor. Por lo dicho hasta el momento una aplicación web es un software específico que se ejecuta en un servidor y es enviado hacia un cliente en donde es interpretado por un navegador.

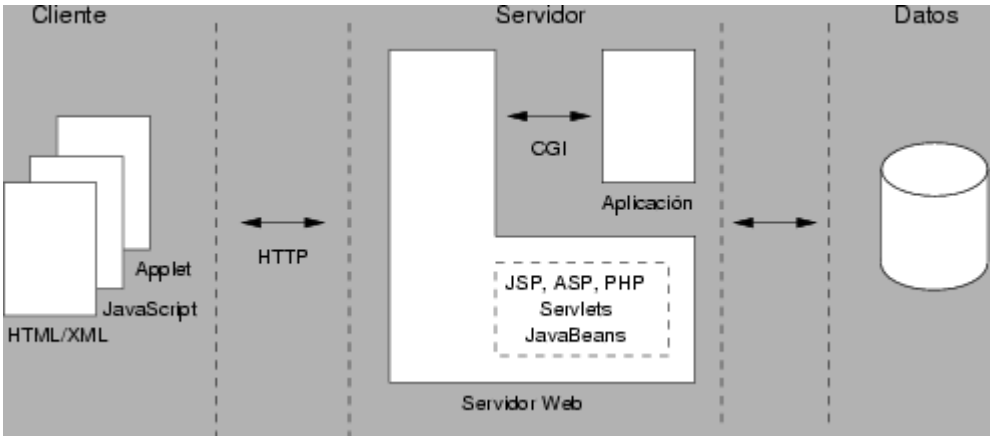


Figura1: Esquema general de las tecnologías web

Fuente: http: //www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node11.html

El uso de aplicaciones web hoy en día es algo muy común y a la vez bastante práctico, ya que la mayoría de las aplicaciones que se realizan, están basadas en aplicaciones web. Esto debido en gran medida a la difusión que ha tenido el internet en los últimos tiempos, y también debido a lo fácil que resulta ingresar e interactuar con una aplicación web. A medida que la tecnología avanza, también es más necesario el uso, la implementación, de dichas tecnologías, y del manejo de la información principalmente de las empresas, es por esto que la aplicación web que se pretende desarrollar para la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, debe estar enfocada a las necesidades de manejo de la información de esta compañía.

[1]En el amplio mundo tecnológico existe una variedad de navegadores web, algunos más populares y conocidos que otros. Se debe tener en cuenta que aunque el objetivo de un navegador es interpretar y mostrar al cliente la información enviada por el servidor, algunos navegadores se basan en la interpretación de cierto código en estándares reconocidos y ampliamente utilizados por la mayoría de los navegadores. Este el caso del Internet Explorer (en versiones anteriores a la 7.0) que no interpretan o no soportan varias propiedades como: CSS (Cascade Style Sheet) y DOM (Document Object Model) similar a como lo hacen por ejemplo Mozilla Firefox y Opera.

[1]Fuente: http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node11.html, Autor "Jesús Vegas", Titulo: "Introducción a las aplicaciones web"

Por estos motivos se tendrá en cuenta el desarrollo de una aplicación web que sea entendida e interpretada de la misma forma por la mayoría de los navegadores, de manera que la información se muestre de forma ordenada y que el usuario de la aplicación la pueda entender.

Hoy en día existen una serie de herramientas y tecnologías para el desarrollo de aplicaciones web. No existe una regla general que nos indique que debamos utilizar una tecnología u otra. Más bien el uso de las herramientas tecnológicas que tenemos a nuestra disposición, están basadas en la necesidad de crear aplicaciones consistentes y que puedan funcionar en forma óptima, satisfaciendo las necesidades de interactuar con el usuario y mostrando la información de forma adecuada y oportuna.

De igual manera se debe considerar que el uso inadecuado de una herramienta u otra puede llevar a tener inconsistencia en los datos o a generar problemas de visualización e incluso de conectividad. Por este motivo es importante tener claro cuál es el objetivo que se quiere alcanzar con el software, y si la herramienta que se pretende utilizar, va a permitir desarrollar y alcanzar todas las metas propuestas.

2.1.1 APLICACIONES WEB CON AJAX[2]

La mayoría de las aplicaciones que se encuentran en la web son de tipo HTML estático. Es decir el usuario solo puede ver el contenido, pero no puede interactuar con el sitio web o con el servidor de la aplicación web.

En este contexto surge una nueva forma de desarrollar aplicaciones basadas en la web conocida como ajax (Asinchronous JavaScript And XML). Ajax no es considerada como una tecnología, sino como un conjunto de tecnologías que pueden ser usadas para desarrollar aplicaciones web dinámicas.

En términos simples ajax es considerada como una capa intermedia entre el navegador y el servidor, de manera que es la capa responsable de interactuar con el servidor, haciendo que la comunicación con el servidor, sea transparente para el usuario.

[2]Fuente:http://www.librosweb.es/ajax/pdf/introduccion_ajax.pdf, autor: “Anónimo”, Titulo “Introducción a AJAX”

Ajax incorpora muchas tecnologías, cada una de las cuales se desarrollan y avanzan por si solas. Las tecnologías que incorpora ajax son:

Presentación basada en estándares usando XHTML y CSS;

Exhibición e interacción dinámicas usando el DocumentObjectModel;

Intercambio y manipulación de datos usando XML and XSLT;

Recuperación de datos asincrónica usando XML HttpRequest y JavaScript poniendo todo junto.

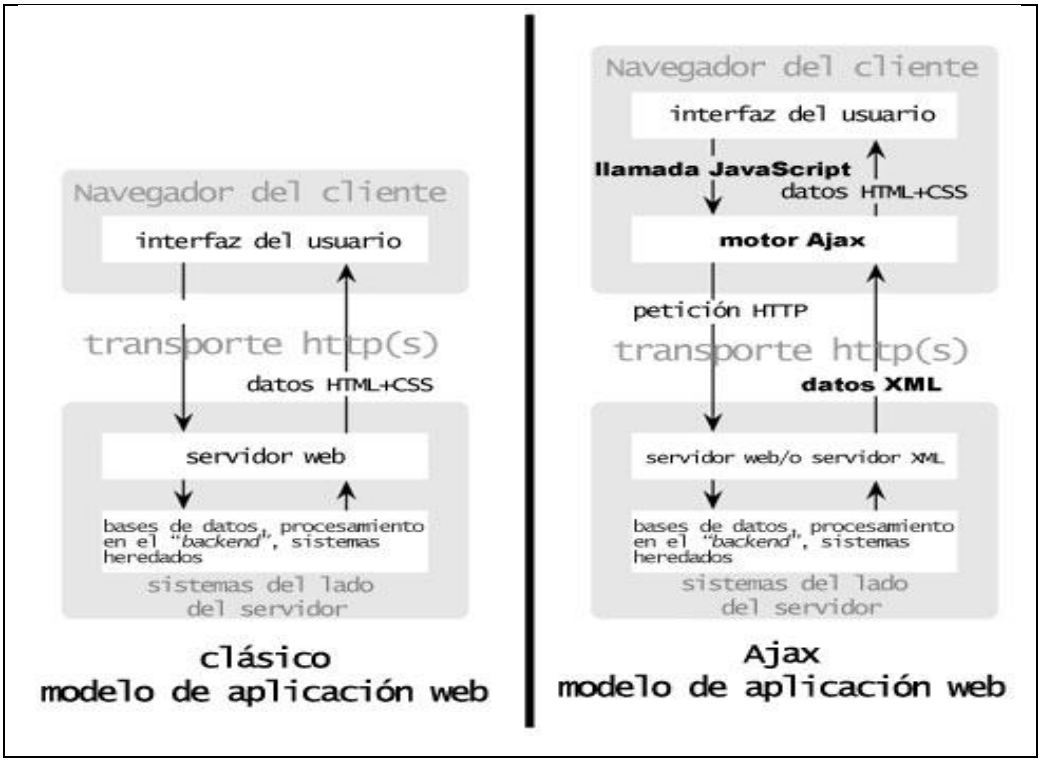


Figura 2: Comparación del modelo web clásico con el modelo web basado en ajax.

Fuente:http://www.librosweb.es/ajax/pdf/introduccion_ajax.pdf

El desarrollo de una aplicación web hoy en día, no puede dejar de incorporar las nuevas tecnologías de la web. Por este motivo se tendrán en cuenta estas nuevas tecnologías en el desarrollo de la aplicación.

2.2 IMPORTANCIA DEL DESARROLLO WEB.

En realidad el desarrollo web implica varias tecnologías tales como: bases de datos, lenguajes de programación, y para cada una de estas existe diferentes herramientas para su implementación. Por ejemplo las bases de datos pueden ser creadas en herramientas tales como: SQL Server, Oracle, MySQL, siendo esta última la más utilizada y ampliamente difundida, debido en gran medida a que se trata de una herramienta de software libre, y que puede ser descargada desde sitios web apropiados.

En cuanto a lo que implica el desarrollo web también existen muchas herramientas en las cuales se pueden crear aplicaciones. Entre las más populares y utilizadas se encuentran: Java, ASP.NET y PHP.

Sin duda estas herramientas serán de gran ayuda a la hora de realizar la aplicación, ya que son las más populares y utilizadas en la creación de sitios web. De estas herramientas se debe escoger la que mejor se ajuste a las necesidades requeridas, teniendo en cuenta también la herramienta con la cual se pueda trabajar a gusto y se pueda crear la aplicación que satisfaga las necesidades del cliente.

Para la creación de la base de datos, de igual manera se tomará en cuenta la herramienta con la que se tenga mayor conocimiento y experiencia en desarrollo en aplicaciones. Esto evitará problemas al momento de realizar las conexiones entre la base de datos y la aplicación web.

2.3 VENTAJAS DE LAS APLICACIONES WEB[3]

Las ventajas que puede ofrecer una aplicación web pueden resultar innumerables, existen muchos artículos ya sea en sitios electrónicos, o en libros dedicados a las aplicaciones web. Aquí se tratará de enumerar las más importantes, las que se considere se ajustan a las necesidades de la aplicación que se va a desarrollar, y las ventajas que puede tener la compañía beneficiaria de este software.

[3] Fuente: tesistas

Estar a la altura de la competencia.

Hoy en día casi todas las empresas poseen su sitio web, y esto se debe en gran medida a que quieren incrementar sus ventas y negocios, o simplemente darse a conocer. Por este motivo la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, no puede quedarse estancada y debe poseer su propio aplicativo, el cual facilite el trabajo diario de sus empleados.

Captación de nuevos clientes.

Al manejar una herramienta moderna como es un sitio web, la empresa agilitará sus trámites con los clientes, dando lugar a reducir el tiempo de solución en dichos trámites y disponer de mayor tiempo para atender a nuevos socios.

Disponibilidad las 24 horas.

Las personas que usen este software podrán acceder a su cuenta desde cualquier lugar y a la hora que deseen, ya sea para actualizar algún dato o para realizar cualquier acción pertinente. Cabe recalcar que cada usuario tendrá un permiso para acceder a la aplicación.

Reducción de costos.

La compañía al poseer su propio software, se ahorra un sin número de trámites engorrosos que antes lo realizaban en papel, como por ejemplo: inscripción de un nuevo socio, datos personales del socio que se lo realizaba de forma inadecuada, traspaso de cuentas, costos de impresión, etc.

Actualización.

Las aplicaciones basadas en web siempre están actualizadas con lo último de la versión, en este caso la compañía no se preocupará por estas actualizaciones ya se actualizará cada vez que sea necesario y sobre todo sin que los usuarios interfieran con sus hábitos de trabajo.

Menos Errores.

Las aplicaciones web deberían ser menos propensas a colgarse y a tener ciertos conflictos con otros programas y aplicaciones existentes que usa el personal interno de la compañía. Las aplicaciones web usan una misma versión y por tal motivo si se detecta un error puede ser corregido lo antes posible.

Información segura.

Las aplicaciones web permiten tener un avanzado sistema de consultas y modificaciones de datos provenientes de cualquier área de la empresa, lo cual mantendrá la información de la compañía siempre actualizada y segura.

Generar una imagen moderna y de profesionalismo.

Sin duda que después de haber implementado este sistema en la compañía, esta estará a la vanguardia y a la altura tecnológica de las más grandes empresas, esto infunde una imagen de competitividad y responsabilidad a sus clientes.

Información protegida

Al encontrarse la información almacenada en un medio electrónico, los datos solo podrán ser leídos por personas que tengan los permisos de software necesarios.

Despliegue de información adecuada

Cuando los usuarios del software realicen una consulta hacia las bases de datos, estos tendrán la certeza de que se mostró la información necesaria y no algún dato que no fue solicitado.

2.4 BASES DE DATOS.[4]

Una base de datos puede considerarse como un conjunto de datos o un tipo de información del mismo tipo, de alguna empresa o institución, la cual es almacenada para su uso posterior. Visto desde este contexto, una biblioteca puede considerarse como una gran base de datos, la cual almacena información en textos impresos y esa información es leída de acuerdo a la necesidad del lector.

[4]Fuente: http://es.wikipedia.org/wiki/Base_de_datos Autor: “Anónimo”, Titulo “Base de datos”

Una base de datos es considerada principalmente un medio informático que almacena la información en formato digital, y que puede ayudar a solucionar en gran medida problemas de acceso a la información de forma sistemática y ordenada.

Existen en la actualidad numerosos programas denominados sistemas gestores de bases de datos, los cuales ayudan a crear bases de datos y a acceder a la información de forma ordenada y estructurada de acuerdo a las diferentes necesidades. De manera general se puede usar una base de datos para numerosos proyectos y aplicaciones, pero el uso más común de una base de datos es para almacenar la información de empresas e instituciones que contiene un gran volumen de información.

Aunque se puede usar una base de datos para almacenar información de distinto tipo como pueden ser imágenes, textos, etc. Se debe tener cuidado con la información que se almacena en una base de datos, ya que puede ser información confidencial la cual solo le interesa a una empresa o institución o que puede ser de utilidad para una única persona.

Se debe tener en cuenta que el diseño de una base de datos debe ser de forma tal que muestre la información que se requiere de forma ordenada y precisa, es decir que se debe diseñar una base de datos para que muestre la información que ha sido solicitada. Por ejemplo si una persona realiza una consulta a una base de datos que almacena información sobre lugares turísticos lo que espera es que se muestre información sobre lugares turísticos, y no información que no necesita conocer.

Otro aspecto importante de las bases de datos es que pueden almacenar información que se puede considerar delicada o importante como por ejemplo: la información sobre una persona que se almacena en una base de datos de un registro civil o la información sobre cuentas bancarias que almacena un banco. Por este motivo es importante proteger la información de manera tal que solo pueda ser vista o solo puedan acceder a ella, personas que tengan permisos y le den un trato adecuado a la información.

Las bases de datos se pueden clasificar de diferentes maneras. Aquí se consideran los dos aspectos más importantes.

Bases de datos estáticas.- son bases de datos de solo lectura. Aquí se puede almacenar por ejemplo información sobre historia.

Bases de datos dinámicas.- son bases de datos con las cuales el usuario puede interactuar. Es decir realizar acciones como modificar, eliminar datos, y por supuesto realizar acciones de consultas.

Se pueden encontrar muchos artículos sobre bases datos y estos les dan varias clasificaciones. Como por ejemplo de acuerdo a la información que almacenan, según el contenido, etc.

Para el propósito que es el de diseñar una base de datos que almacenará información a cerca de los socios de la "Compañía de Promoción y Desarrollo Humano y Social Coopueblo". Se ha considerado estos dos aspectos como los más importantes y sobre los cuales estará enfocado el desarrollo. Para mayor información se recomienda visitar el documento sugerido en el pie de página, que es el documento del cual fue extraída esta información.

Considerando que se va a desarrollar una aplicación basada en la web y que esta va a ser implementada en ASP.NET el esquema general de acceso a la información es el siguiente.

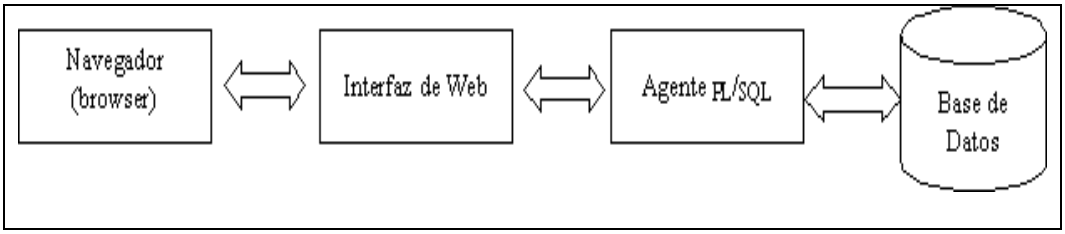


Figura 3: Modelo de acceso a la base de datos

Fuente: http://www.hipertexto.info/documentos/b_datos.htm

En el esquema se observa:

Navegador (Browser)._ Es la herramienta con la cual se va acceder a la información. Por tratarse de que la aplicación va a ser construida en ASP.NET y este utiliza el servidor IIS, el navegador a utilizar será Internet Explorer.

Interfaz web.- Proporciona un mecanismo de visualización del código HTML y de la información de las bases de datos.

Agente Transact/SQL.- Es el nexo entre el navegador del cliente y la base de datos. El agente ejecutará una llamada a un procedimiento almacenado en el servidor. Este procedimiento creará una página HTML dinámica como salida, y el agente devolverá

dicha salida al cliente a través del navegador empleando de igual manera la Interfaz de Web.

Base de datos.- Lugar en donde se almacena la información, entrega los datos que hayan sido solicitados por el agente, y es mostrada al cliente a través del navegador.

2.5 TECNOLOGÍA UTILIZADA PARA EL DESARROLLO DEL SOFTWARE.

En este punto se realizará una explicación minuciosa a cerca de la tecnología que se va a utilizar para el desarrollo web del sistema. Así como también se analizarán sus ventajas y desventaja.

El mismo procedimiento se realizará con las herramientas para la creación de bases de datos más populares. Tanto de las tecnologías de desarrollo web como de las herramientas para la creación de bases de datos, se escogerá la que más se ajuste a nuestras necesidades y conocimiento de cada tecnología, y con ellas se realizará el trabajo de desarrollo del sitio web.

2.5.1 ASP.NET

Asp.net es un framework para aplicaciones web, se usa en gran medida para desarrollar aplicaciones web dinámicas, servicios web y servicios web XML. Asp.net es la sucesora de la tecnología ASP (Active Server Pages), y está construido sobre el Common language Runtime, lo cual permite a los desarrolladores escribir aplicaciones en cualquiera de los lenguajes soportados por el .net Framework como C#, Vb y otros.

[5]Microsoft introdujo la tecnología llamada Active Server Pages. Es parte del Internet Information Server (IIS) desde la versión 3.0 y es una tecnología de páginas activas que permite el uso de diferentes scripts y componentes en conjunto con el tradicional HTML para mostrar páginas generadas dinámicamente. La definición contextual de Microsoft es que Las Active Server Pages son un ambiente de aplicación abierto y gratuito en el que se puede combinar código HTML, scripts y componentes ActiveX del servidor para crear soluciones dinámicas y poderosas para el web".

[5]http://es.wikipedia.org/wiki/ASP.NET, Autor:"anónimo", Titulo: "ASP.NET", Año publicación: 2012

Las páginas ASP pueden hacer uso de objetos COM (Component Object Model) que son objetos en algún otro lenguaje (ej. ejecutables en C++ o Java); de manera que si ya se tiene algo programado las páginas ASP a través del IIS pueden hacer uso de los métodos en estos objetos.

Las páginas de ASP.NET, conocidas oficialmente como "web forms" (formularios web), son el principal medio de construcción para el desarrollo de aplicaciones web. Los formularios web están contenidos en archivos con una extensión ASPX; en jerga de programación, estos archivos típicamente contienen etiquetas HTML o XHTML estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor y Controles de Usuario donde los desarrolladores colocan todo el código estático y dinámico requerido por la página web. Adicionalmente, el código dinámico que se ejecuta en el servidor puede ser colocado en una página dentro de un bloque que es muy similar a otras tecnologías de desarrollo como PHP, JSP y ASP, pero esta práctica es, generalmente, desaconsejada excepto para propósitos de enlace de datos pues requiere más llamadas cuando se genera la página.

ASP.NET sólo funciona sobre el servidor de Microsoft IIS, lo que supone una desventaja respecto a otros lenguajes del lado de servidor, ejecutables sobre otros servidores más populares como Apache. Ejemplos de esto son PHP, Perl o Python.

2.5.1.1 VENTAJAS DE ASP.NET

- es un lenguaje completamente orientado a objetos.
- Controles de usuarios personalizados.
- División entre la capa de aplicación o diseño y el código.
- Soporta varios lenguajes admitidos por el .NET Framework.
- Facilita el mantenimiento de grandes aplicaciones.
- Incremento en la velocidad de respuesta del servidor.
- Mayor seguridad.

2.5.1.2 DESVENTAJA

Mayor consumo de recursos.

ASP.Net cumple con muchos de los requisitos para convertirse en el lenguaje preferido para el desarrollo de aplicaciones web, ya que posee una interfaz gráfica muy amigable, la cual está en constate desarrollo para ofrecer mayores facilidades a los desarrolladores. Además que tiene una gran compatibilidad con otras tecnologías como AJAX, la cual se usará de manera íntegra durante todo el proceso de desarrollo de la aplicación.

2.5.2 ADO.NET[6]

En pocas palabras ADO.NET es un conjunto de aplicaciones de software que nos permiten el acceso a una fuente de datos, es decir permite la lectura y escritura en un determinado sistema gestor de base de datos tanto relacionales como no relacionales.

ADO.NET es solo una parte de un conjunto de aplicaciones que forman parte del .NET FRAMEWORK, también se puede decir que es una parte de una biblioteca de clases base incluidas en el .NET Framework.

ADO.NET consta de dos partes principales:

DATA PROVIDER

Provee el acceso a distintas fuentes de datos tales como: Microsoft SQL Server y Oracle. Cada una tiene su propia fuente de datos de objetos del proveedor. Pero a su vez comparten un conjunto de clases de utilidad.

Connection.- provee de una conexión a una fuente de datos.

Command.- se usa para ejecutar alguna acción en una fuente de datos relacional tales como: lectura, actualización o eliminación de datos.

Parameter.- proporciona parámetros que pueden ser usados por ejemplo por un procedimiento almacenado.

[6]http://www.ehu.es/mrodriguez/archivos/csharppdf/ADONET/ADONET.pdf, Autor: “Marco Besteiro y Miguel Rodriguez”, Titulo “ADO.NET”, Año publicación: 2011

DataAdapter.- es utilizado como puente para trasferir datos entre una fuente de datos relacional y los DataSet.

DataReader.- se utiliza para procesar una lista grande de resultados, un registro a la vez.

DATASETS

Los DataSets son un conjunto de objetos que describen una base de datos relacional posee un conjunto de clases, que forman una jerarquía de contención.

Un objeto DATASET contiene las relaciones entre las tablas de una base de datos, así mismo puede contener una base de datos entera o parte de ella.

DataTable.- Representa una tabla de una base de datos, tiene filas, columnas y un nombre.

DataView.- Actúa sobre un DataTable, y ordena los datos como una cláusula "OrderBy de AQL".

DataColumn.- Representa una columna de una tabla así como también su nombre y tipo de dato.

DataRow.- Representa una fila en la tabla, a la vez que permite leer y actualizar datos de la fila, así como la recuperación de cualquier fila.

DataRowView.- representa una fila en un DataView, la diferencia entre un DataRow y el DataRowView es importante cuando se está interactuando sobre un ResultSet.

DataRelation.- es una relación entre tablas, tal como una relación entre clave primaria y clave foránea, es útil ya que permite la funcionalidad del DataRow, de recuperar filas relacionadas.

Constraint.- propiedad que se debe cumplir, por ejemplo: los valores de una clave primaria, que deben ser únicos. Causará excepciones a medida que los datos se vayan actualizando, y se viole la regla de clave primaria.

Un **DataSet** representa un conjunto completo de datos, incluyendo las tablas que contienen, ordenan y restringen los datos, así como las relaciones entre las tablas.

2.6 BASE DE DATOS UTILIZADA PARA EL DESARROLLO DEL SOFTWARE.

En este punto se realizará un breve resumen de la base de datos que se va a utilizar para el desarrollo del software.

2.6.1 SQL SERVER.[7]

SQL Server es un sistema de gestión de bases de datos relacional, desarrollado y distribuido por Microsoft, nace como una alternativa a otros potentes sistemas de gestión de bases de datos como Oracle, MySQL y PostgreSQL; el lenguaje de consulta para una base de datos realizada en SQL Server es el T-SQL.

Transact-SQLes el lenguaje que utiliza SQL Server para enviar peticiones tanto de consultas, inserciones, modificaciones, y de borrado a las tablas, así como otras peticiones que el usuario necesite sobre los datos. En definitiva, es un lenguaje que utiliza SQL Server para poder gestionar los datos que contienen las tablas.

El lenguaje estándar SQL (Structured Query Language) se emplea para los sistemas de bases de datos relacionales RDBMS (Relational Database Management System), es el estándar ANSI (American National Standards Institute). También es utilizado por otros sistemas como: Oracle, Access, Sybase, etc.

SQL Server puede gestionar el sistema operativo, recursos del servidor,todo lo relacionado con la información que va a almacenar, para esto SQL-Server utiliza algunos componentes para gestionarlo.

Net-Library: Es el componente que controla las conexiones de diferentes protocolos y redes. Habilita SQL Server para escuchar a múltiples protocolos al mismo tiempo. Se puede configurar el servidor fácilmente para escuchar múltiples protocolos, empleando utilidades de red del servidor bajo esta herramienta. Cuando se intenta conectar a SQL Server y este devuelve un error de comunicación, lo primero que hay que comprobar es el componente Net-library.

[7]Fuente: http://html.rincondelvago.com/base-de-datos_3.html, Autor “anónimo”, Titulo “Base de datos”

Open Data Services (ODS): Es el componente que está escuchando para nuevas conexiones y respuestas. ODS controla las conexiones a SQL Server. ODS también controla las desconexiones inesperadas y deja libres los recursos del sistema.

Tabular Data Stream (TDS): Es un protocolo privado que SQL Server emplea para cifrar los datos y comunicarse con las estaciones clientes.

Motores de SQL Server: Existen dos motores muy importantes en SQL Server:
El motor relacional: Incluye los componentes necesarios para la consulta de datos.
El motor de almacenaje: Gestiona el almacenaje físico de los datos y la actualización de los datos en disco.
Ahora veamos una forma gráfica de cómo trabaja SQL Server.

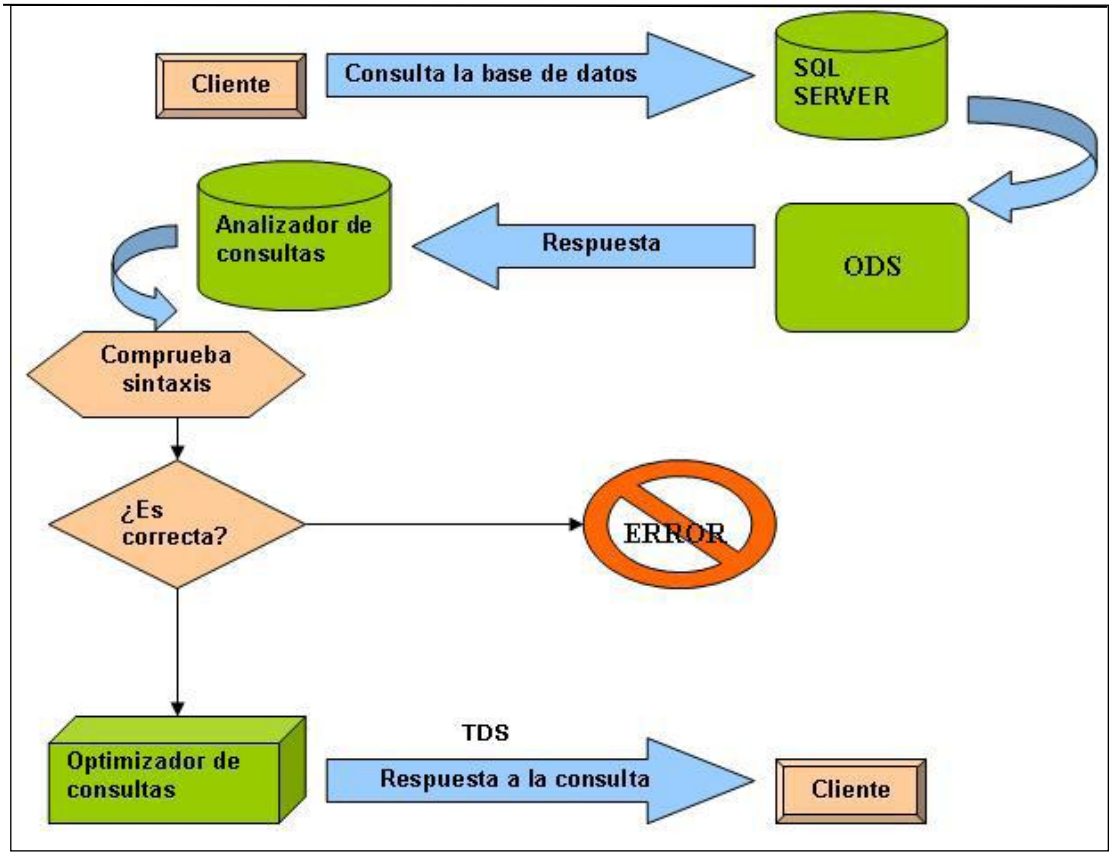


Figura 4: Proceso de consulta con SQL Server
Fuente: http://msdn.microsoft.com/es-es/library/ms190623(v=sql.105).aspx

En realidad SQL Server realiza todo ese trabajo en 6 pasos independientes.

Paso1. Un cliente realiza una consulta a la base de datos.

Paso2. El analizador de consultas recibe la respuesta del ODS.

Paso3. El analizador de consultas comprueba la sintaxis.

Paso4. Si la sintaxis es incorrecta, el analizador de consultas devuelve un error.

Paso5. Si la sintaxis es correcta la respuesta se pasa al optimizador de consultas.

Paso6. Se devuelve la respuesta al cliente.

2.6.2 VENTAJAS DE SQL SERVER.

- Posibilidad de activar la seguridad integrada con NT.
- Es un Sistema de Gestión de Bases de Datos Relacionales (SGBDR).
- Puede ser útil para manejar y/o obtener datos de la red de redes.
- Ofrece una potente forma de unir SQL e Internet.
- El Transact SQL, soporta la definición, modificación y eliminación de bases de datos, tablas, atributos, índices, etc., es decir, el lenguaje de definición de datos (LDD), así como la consulta, actualización y borrado de tuplas de tablas.
- Un punto importante a favor de SQL Server es la interfaz de acceso OLE DB y ADO. Aunque se trata de una interfaz universal, SQL Server es una de las primeras bases de datos en soportarla.
- Mejor utilización de la CPU.
- Menor necesidad de limpieza de las memorias intermedias durante el procesamiento de las transacciones.
- SQL Server registra las transacciones de tal modo que las actualizaciones en una de ellas siempre se puedan recuperar o reducir al último estado consistente si el equipo cliente o servidor falla.
- Cuando SQL Server no tiene tareas de usuario para procesar, comienza a escribir automáticamente las memorias intermedias sucias del caché al disco. Como estas escrituras se realizan en los ciclos de inactividad del servidor, se denominan escrituras libres.

- Menor tiempo de recuperación.

2.6.3 DESVENTAJAS.

- MSSQL usa Address Windowing Extention (AWE) para hacer el direccionamiento de 64-bit. Esto le impide usar la administración dinámica de memoria y sólo le permite alojar un máximo de 64GB de memoria compartida.
- MSSQL no maneja compresión de datos (excepto la versión 2008 Enterprise Edition incluye esta característica), por lo que puede llegar a ocupar mucho espacio en disco.
- MSSQL requiere de un sistema operativo Microsoft Windows.
- La relación calidad-precio está muy debajo comparado con Oracle.

Los gestores de bases de datos tienen sus Pro y contras. Dependiendo de la necesidad y el uso que el usuario les quiera dar, pero existe un detalle muy importante que hace que SQL Server sea uno de los gestores de base de datos más usado en el mundo. Y es que tiene una muy buena conexión con .Net, es decir las conexiones con estas herramienta de desarrollo es muy estable, y además que; son herramientas desarrolladas por la misma empresa. Por lo que tienen una gran compatibilidad.

Por este motivo el gestor de base de datos que se usará para el desarrollo del sistema, es SQL Server, ya que además; es una herramienta fácil de usar y con la cual el equipo de desarrollo está muy familiarizado, y tiene mucha experiencia en el manejo de esta herramienta.

2.7 METODOLOGIA PARA EL DESARROLLO DEL SISTEMA

2.7.1 EXTREME PROGRAMMING (XP).

Extreme Programming surge como una nueva alternativa para desarrollar proyectos de software, proponiendo una metodología que se basa esencialmente en la simplicidad y agilidad. Las metodologías tradicionales de desarrollo de software (ciclo de vida en cascada, modelo en espiral, modelo evolutivo, modelo iterativo,

etc.) al ser comparados con el modelo de desarrollo XP son considerados pesados y poco eficientes. La crítica más fuerte a estas metodologías clásicas es que son demasiado burocráticos, es decir consisten en una serie pasos y fases muchas veces innecesarios para lograr algo muchas veces simple o para alcanzar el objetivo final que es el software. Además son tantos pasos que hay que seguir que muchas de las veces el trabajo se retarda, provocando de esta forma que los plazos para la entrega del trabajo no se cumplan.

En respuesta a estas metodologías poco eficientes han nacido en los últimos años metodologías denominadas "metodologías ágiles", estos nuevos métodos buscan un punto intermedio entre la ausencia de procesos y el abuso de los mismos, proponiendo un proceso cuyo esfuerzo valga la pena.

Algunas de las características más importantes de las metodologías denominadas ágiles son las siguientes:

Las metodologías ágiles son adaptables en lugar de predictivas. Los métodos "clásicos" tienden a intentar planear una gran parte del proceso del softwareen gran detalle para un plazo largo de tiempo. Esto funciona bien hasta quelas cosas cambian. Así que su naturaleza es resistirse al cambio. Para losmétodos ágiles, no obstante, el cambio es bienvenido. Intentan serprocesos que se adaptan y crecen en el cambio.

Los métodos ágiles son orientados a la gente y no orientados al proceso. Elobjetivo de los métodos clásicos es definir un proceso que funcionará bien independientemente de quien lo utilice. Los métodos ágiles afirman queningún proceso podrá nunca maquillar las habilidades del equipo dedesarrollo, de modo que el papel del proceso es apoyar al equipo dedesarrollo en su trabajo.

XP es una de las llamadas metodologías ágiles de desarrollo de software más exitosas de los tiempos recientes. La metodología propuesta en XP está diseñada para entregar el software que los clientes necesitan en el momento en que lo necesitan. XP alienta a los desarrolladores a responder a los requerimientos cambiantes de los clientes, aún en fases tardías del ciclo de vida del desarrollo.

La metodología también enfatiza el trabajo en equipo. Tanto gerentes comoclientes y desarrolladores son partes del mismo equipo dedicado a entregar software de calidad.

Cabe mencionar que la metodología XP está diseñada para proyectos pequeños y cuya entrega es en plazos cortos, ya sea de semanas o de meses. Desde este punto de vista este modelo se adapta perfectamente a un trabajo de proyecto de titulación, ya que entre otras cosas propone la programación en pares.

La metodología XP se basa en cuatro principios fundamentales para su desarrollo. Estos son:

Se valora a los individuos y las interacciones sobre los procesos y las herramientas.

Se valora a las aplicaciones que funcionan sobre la documentación exhaustiva.

Se valora la colaboración del cliente sobre las negociaciones contractuales.

Se valora la respuesta al cambio sobre el seguimiento de un plan.

Los ítems que más se valoran son los dos primeros, pero esto no quiere decir que los otros dos no sean importantes para el desarrollo de un proyecto.

Estos ítems se ponen de manifiesto en detalle en los siguientes principios de la metodología XP.

La mayor prioridad es satisfacer al cliente mediante entregas continuas y tempranas.

Los cambios a los requerimientos son bienvenidos aun en fases tardías del desarrollo.

Entregar software que funcione en periodos cortos de semanas o meses, prefiriendo los periodos más cortos.

Desarrolladores, clientes y gerente deben trabajar juntos a lo largo del proyecto.

Construir el proyecto alrededor de personas motivadas, dándoles el entorno y soporte que necesitan, y confiando en que realizarán el trabajo.

El método más eficiente y efectivo de trasmitir la información entre desarrolladores es la conversación frontal.

Tener software que funciona es la medida primaria del progreso.

El proceso ágil promueve el desarrollo sostenible. Los sponsors, desarrolladores y usuarios deben ser capaces de mantener un ritmo de trabajo constante en forma permanente a lo largo del proyecto.

A intervalos regulares, el equipo debe reflexionar sobre cómo ser más efectivos, y ajustar su comportamiento de acuerdo a ello.

Los desarrollos de software ágil, no son anti-metodológicos. Por el contrario, siguen su metodología, diferente a la de los métodos clásicos de desarrollo. Se trata de lograr un equilibrio, en el que, por ejemplo, la documentación es concreta y útil, y no burocrática y los planesexisten, pero reconociendo sus limitaciones en el actual mundo en permanente cambio.

2.7.1.1 CICLO DE VIDA DEL SOFTWARE EN XP.

Al igual que en otras metodologías, el ciclo de vida del software en XP incluye entender lo que el cliente necesita, estimar el esfuerzo para el desarrollo del proyecto, crear la solución del proyecto y, entregar la solución definitiva al cliente. A diferencia de las metodologías tradicionales de desarrollo de software, XP propone un modelo de desarrollo dinámico. Es decir que se adapte al cambio en cualquier etapa del proyecto, esto es de suma importancia para un equipo de desarrolladores, ya que en ciertas ocasiones el cliente olvida comunicar algunos aspectos importantes que debe contener el software final. Por esto se trata de realizar ciclos de desarrollo cortos (iteraciones), con entregas funcionales al finalizar cada ciclo. Es decir que si al cliente se le olvidó comunicar algún detalle importante u observa que falta algún componente en el software, este pueda ser corregido e implementado si fuere el caso.

Por esta razón se trata de realizar ciclos de desarrollo cortos (llamados iteraciones), con entregas de software funcionales al finalizar cada ciclo. Es decir se deberá entregar partes software en ciclos cortos, pero cada una de estas partes deberá funcionar de manera óptima.

Un proceso de desarrollo de software en el modelo XP lleva típicamente de 10 a ciclos o iteraciones. En la siguiente figura se hace una comparación de los modelos en cascada e iterativo con el modelo XP.

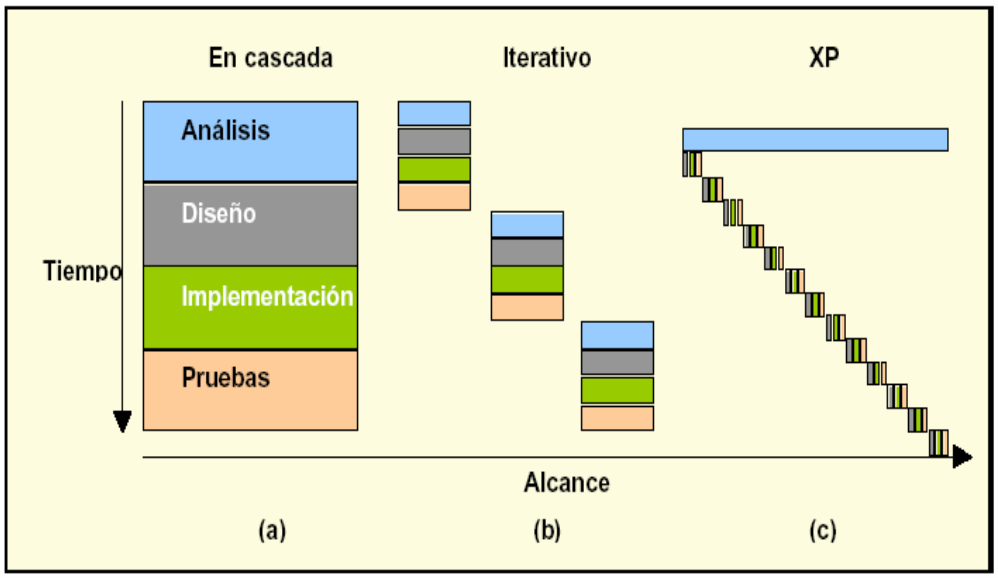


Figura 5: Comparación de XP con otros modelos

Fuente:http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf

La metodología XP establece cuatro variables para el proceso de desarrollo de todo proyecto de software. Estas son:

- **Costo.**
- **Tiempo.**
- **Calidad y.**
- **Alcance.**

De estas cuatro variables XP promueve que solo tres de ellas pueden ser fijadas por agentes externos, es decir; ya sea por el cliente o por el gerente o en este caso por la empresa encargada del proyecto, y una de las variables debe ser establecida por el equipo de desarrolladores. Por ejemplo: si el cliente o la empresa encargada del proyecto eligiesen: Costo, tiempo y calidad, el equipo de desarrolladores estará en plena libertad de elegir el alcance del proyecto.

En el caso concreto de la "compañía de promoción y desarrollo humano y social Coopueblo", esta ha establecido las variables de tiempo, calidad y alcance, y el equipo de desarrolladores ha establecido el costo del proyecto.

2.7.1.2 FASES DEL MODELO XP.

Para cumplir con el objetivo de entregar software de calidad. El modelo XP se puede separar en seis fases. Cada una de las tareas de estas fases será descrita en la sección posterior Reglas y prácticas.

FASE I: EXPLORACIÓN.

En esta etapa se tiene una visión general del proyecto. Se empieza por recopilar las historias de los usuarios, y se realiza una estimación del tiempo que tardará el proyecto, así como también del tiempo que tardará en desarrollarse cada historia.

En este punto de partida del proyecto, el cliente redacta con sus propias palabras las necesidades del sistema mediante la redacción de sencillas historias de usuario. Cada historia será desarrollada en un lapso de tiempo de dos a tres semanas. En base a esta información los programadores estiman el tiempo de esfuerzo que les tomará desarrollar cada una de las historias. Cabe recalcar que la información aquí obtenida, es solo una estimación a cerca del tiempo total de proyecto, ya que los datos aquí obtenidos son de alto nivel y no representan el tiempo real del proyecto.

La creación de las historias de usuarios, son un punto de partida muy importante, de hecho son la parte esencial del proyecto, ya que mediante estas el equipo de desarrolladores. Puede empezar a planificar los tiempos de entrega de cada historia. Principalmente el plazo de entrega de la primera entrega.

De igual manera para el equipo de desarrolladores empieza a familiarizase con las herramientas que utilizará en el desarrollo del sistema, así como también con las diferentes tecnologías que estas implican. Dependiendo del tamaño del proyecto y la familiaridad que los programadores tengan con las tecnologías implicadas, esto puede tardar desde unas semanas hasta unos meses. El equipo de desarrolladores empieza por construir un prototipo de lo que será el sistema final, para de esta manera estimar el tiempo total del proyecto.

FASE II: PLANIFICACIÓN.

Esta es una de las fases más cortas del proyecto, y consiste en que todo el personal involucrado en el proyecto se reúne para acordar el orden de entrega de las historias de usuario creadas en la sección anterior. Esto se logra mediante reuniones con todo el equipo de trabajo como son: cliente, gerente, y equipo de desarrolladores. En este punto también se acuerdan los tiempos de entrega de cada una de las historias. Esta fase consta de una o varias reuniones grupales, de acuerdo a la necesidad que el proyecto lo requiera.

En la fase de planificación también se plantea la prioridad de cada historia. Es decir el orden en que deben ser desarrolladas de acuerdo al grado de importancia que estas tienen según lo plantee el cliente de acuerdo a sus necesidades más urgentes, o de acuerdo a las posibilidades de desarrollo del equipo desarrollo. Se debe realizar un cronograma de entrega para cada iteración, y sobre todo lo más importante, se debe establecer el plazo de entrega para la primera historia a desarrollarse, el plazo de entrega de la primera historia no debe ser superior a tres meses, sin embargo el tiempo de vida de esta fase es de unos pocos días.

Para estimar el tiempo que tardará en ser creada cada historia, el equipo de desarrollo realiza una valoración de cada historia de usuario, de acuerdo al grado de complejidad que se lo considere, esto se lo realiza mediante una escala de 1 a 3 en donde 3 es el grado de complejidad mayor que se puede dar a una historia. Se considera que el desarrollo de un punto en la escala tomará de una semana de desarrollo.

La planificación de los tiempos de entrega se los realiza tomando en cuenta la cantidad de historias que pueden ser desarrolladas en un lapso de tiempo dado de acuerdo a los puntos de la escala antes mencionada.

FASE III: ITERACIONES.

Se considera la parte principal del ciclo de vida de un proyecto utilizando la metodología XP, en este parte del proyecto se desarrolla todas las funcionalidades del sistema, es decir es la parte en la que se desarrollan todas las historias de usuario antes creadas, probadas y entregadas de acuerdo al cronograma fijado en secciones anteriores. Aquí se entregan bloques de software funcionales, es decir partes de

código que funcionen y que cumplan con el objetivo de cada historia. Dado que las historias de usuario no son lo suficientemente claras, y no contienen en detalle lo que el sistema debe realizar, es necesario que el cliente intervenga de forma activa para poder recopilar toda la información en detalle a cerca de lo que el sistema debe realizar. De esta manera no se pueden omitir detalles importantes que a lo mejor no fueron consideradas en la creación de las historias del usuario. El progreso de cada iteración indica de forma clara el avance del proyecto, y esto es muy importante para poder cumplir con el cronograma establecido.

Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable, pero llevadas a cabo por parejas de programadores.

FASE IV: PRODUCCIÓN.

En este punto del proyecto se realizan principalmente tareas de ajustes y correcciones a cada iteración antes mencionada, dado que en cada iteración se la entrega módulos funcionales y sin errores, puede ser que el cliente no desee utilizar el sistema en partes, sino poner en producción el sistema final, con la entrega final del producto, por este motivo en este puno se ponen a prueba todos los bloques funcionales del sistema para verificar posibles errores y corregirlos.

FASE V: MATENIMIENTO.

En este punto se requiere de un mayor esfuerzo para satisfacer las necesidades del cliente, principalmente consiste en realizar tareas de soporte al cliente para satisfacer necesidades actuales, es decir puede ser necesario realizar una iteración faltante, o que se tenga que modificar alguna para satisfacer necesidades actuales del sistema. Por este motivo la fase de mantenimiento puede requerir de nuevo personal o que se realicen cambios en el equipo de trabajo para realizar las nuevas tareas requeridas.

FASE VI: MUERTE DEL PROYECTO.

Esto sucede cuando no existan más historias de usuario para ser desarrolladas, y cuando se ha cumplido con las expectativas del cliente, ya sea en aspectos de rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto

también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo.

CAPÍTULO III

ANÁLISIS Y DISEÑO DEL SISTEMA.

En este capítulo se analizará como la Compañía de Promoción y Desarrollo Humano y Social Coopueblo funcionaba antes de que se procediera con el diseño e implementación del nuevo sistema, con qué requisitos contaba tanto de hardware como de software, aspectos fundamentales para poder tener una visión clara de que es lo que necesita la compañía.

3.1 SITUACIÓN ACTUAL

La compañía está funcionando por alrededor de 10 años, empezó con cuatro computadoras las cuales no tenían instalado ningún programa para poder realizar ingresos de los socios, actualización de datos, ingreso de comprobantes de depósitos y todo lo que requiere una compañía que se dedica a la venta de bienes inmuebles.

La compañía fue creada para personas de escasos recursos económicos, y las personas que tuvieron está visión no contaron con asesoramiento adecuado para realizar esta actividad.

En ese tiempo la compañía llevaba todos los registros de sus clientes en libros contables, puesto que no tenían mucha demanda de clientes, y en ese momento no tenían la necesidad de implementar un sistema para realizar el registro de los socios.

Pero al pasar el tiempo, se incrementaban los socios y con ello las necesidades de la empresa en brindar un servicio de mejor calidad tanto en infraestructura como personal más capacitado y los más importante tener tecnología de punta.

La compañía estuvo inmersa en una serie de desfases a nivel económico, ya que por el aumento de socios, el personal que se encargaba del ingreso de los comprobantes de depósitos en los libros contables se empezó a dar cuenta que existían comprobantes de depósitos duplicados lo cual encadeno una pérdida económica fuerte.

Este problema lo tuvieron por algunos meses, puesto que los clientes se empezaron a dar cuenta que se podía entregar el mismo comprobante varias veces sin ningún problema.

Luego de que el contador general de la compañía realizó una revisión minuciosa de los libros contables, informó al gerente general y a los encargados del funcionamiento de la compañía lo que estaba sucediendo.

Al ver lo que ocasionó este gran inconveniente la secretaria y el contador general contactaron a varias empresas que se dedicaban al desarrollo e implementación de sistemas contables para poder solucionar el inconveniente, pero les costaba mucho dinero la implementación del programa.

Mientras tanto siguieron registrando los números del comprobante de depósito en libros contables mientras conseguían un proveedor que les pudiera cobrar menos.

A continuación se darán a conocer todos los requerimientos necesarios que necesita la compañía para poner en marcha el funcionamiento del nuevo software.

3.2 REQUERIMIENTOS DEL USUARIO

En este punto se tratará los requerimientos y necesidades que tiene el usuario, para llevar un control ordenado de los registros y cuentas de los socios. Los puntos que aquí se traten se centrarán en cada uno de los módulos que contendrá el sistema informático propuesto.

3.2.1 Inscripción de socios

En este módulo se registrara al nuevo socio de la compañía. Es decir cuando una persona quiere adquirir un bien inmueble lo primero que debe hacer es registrarse como socio en la compañía mencionada; además puede ingresar a la compañía con un valor inicial de inscripción, el cual previamente debió depositar en la cuenta de la institución.

Para que una persona pueda registrarse como socio, debe otorgar cierta información a la compañía, la cual será ingresada en el sistema por un operario del software. Los datos que se solicita al socio es la información personal de cada persona, como por ejemplo: dirección, teléfono, etc.

Si una persona se registró como socio de la compañía, se le entregará un comprobante en donde conste su inscripción como nuevo socio de la compañía, en este documento se le imprimirá los datos del socio como: un código individual para

cada socio, generado por el sistema, datos del socio, el valor de la inscripción, entre otros.

Una vez ingresado el socio, también se lo puede editar para corregir cualquier error de ingreso o ya sea el caso de que el socio quiera actualizar sus datos, como por ejemplo. Cambio de domicilio o teléfono, o si fue por un error de digitación de un operario del software, por cualquier motivo de estos, el software tendrá un módulo de edición de los socios.

De igual manera, una vez que el socio haya sido registrado, el sistema tendrá una opción de búsqueda de los socios, ya sea por su número de cedula o por los nombres y apellidos de los socios.

3.2.2 Registro de depósitos bancarios

En este módulo, se podrá administrar las cuentas de los socios. Una persona que se ha convertido en socio de la compañía puede abrir más de una cuenta, y el sistema lo permitirá realizar esta operación, debido a que una persona puede adquirir más de un bien, y además realizar abonos a la cuenta, por diversos motivos como son:

- Gastos administrativos
- Abono del bien.
- Obra.

El sistema permitirá administrar estas cuentas por separado, para mantener un control adecuado de cada una de ellas. Cada cuenta manejará el tipo de depósitos mencionados anteriormente por separado.

Cuando un socio quiere ingresar un nuevo depósito debe especificar la cuenta a la cual quiere hacer el abono (en caso de tener más de una) y el tipo de depósito, siendo este por los tres tipos de depósitos descritos anteriormente.

Así mismo el sistema le permitirá al operario, realizar consultas acerca de cada una de las cuentas de los socios y verifica los saldos de cada una de las cuentas de los socios. En este módulo también se podrá verificar los datos de los socios como son los datos personales del socio y los datos referentes a cada cuenta del socio.

El sistema generará un número de transacción en el depósito del socio. Este número de transacción le permitirá al operario del software cancelar un depósito de cierta cuenta, por alguno de los tres motivos siguientes:

- Error de digitación
- Duplicidad del documento
- Otro

Un depósito se puede cancelar por alguno de los tres motivos especificados. De esta manera el operario del software, podrá llevar un control ordenado de las cuentas de los socios, y además verificar que no existan irregularidades en las cuentas de los socios.

3.2.3 Creación de cuentas

El sistema permitirá la creación de cuentas para los socios, debido a que los clientes de la compañía pueden adquirir más de un bien, se les debe crear una cuenta para cada bien que el socio adquiera, ya que los depósitos que este realiza, son abonos independientes que realiza a sus cuentas.

Es de vital importancia la creación de cuentas para los socios de la compañía, ya que para realizar el traspaso de saldos entre socios, se lo debe realizar mediante sus cuentas.

3.2.4 solicitud de transferencias

Para que un socio transfiera su saldo de su cuenta hacia la cuenta de otro socio, es necesario que esta transacción sea aprobada por la gerente de la compañía. Para ello el sistema contará con un módulo en donde se gestione las solicitudes de los socios, y otro en donde estas sean aprobadas o rechazadas.

3.2.5 Transferencias

El módulo de transferencias el permitirá al socio de la compañía realizar transferencias entre cuentas hacia otros socios de le compañía. Para ello primeramente el sistema verificará que existan las cuentas de los socios para poder realizar la transferencia. Una vez verificados los datos, se debe especificar el motivo

y alguna observación por la cual se desea realizar la transacción, para luego generar la solicitud de transferencia, la cual deberá ser aprobada por la gerente de la compañía para que la transacción se realice.

De igual manera se puede consultar un listado de las transferencias entre cuentas que se hayan realizado hasta el momento.

Esta operación, servirá para crear un número de solicitud, la cual se usará para realizar la transacción final de los saldos entre cuentas.

De igual manera si un socio desea retirarse de la compañía, el sistema le permitirá realizar esta gestión, para lo cual se debe crear una solicitud y que esta sea aprobada por el gerente de la compañía.

3.2.6 Retiros

Un socio puede retirarse de la compañía cuando lo desee, por este motivo se debe gestionar los retiros de los socios. Llevar los depósitos de los socios de una forma ordenada y consistente, le confiere al usuario del sistema la seguridad necesaria para realizar las transacciones de manera rápida y segura.

3.2.7 Creación de usuarios del sistema

El sistema informático permitirá la creación de nuevos usuarios. De acuerdo al perfil que tengan. Pudiendo ser estos de varios tipos como son:

- Administrador.
- Gerente.
- Contador.
- Operador.

El sistema también permitirá la creación de nuevos perfiles, de acuerdo a los requerimientos de la compañía.

Así mismo el sistema permitirá la asignación de roles de acuerdo al perfil de los usuarios. Permitiéndoles realizar ciertas operaciones de acuerdo a las políticas que la compañía crea pertinentes. Por ejemplo: a un usuario dentro de un rol se le permitirá

que solo pueda consultar saldos mas no ingresar nuevas cuentas o socios en el sistema.

3.3 DISEÑO DEL SISTEMA

Antes de empezar con el proceso de desarrollo, es necesario describir las necesidades del usuario, así como también conocer la estructura y funcionamiento de la compañía.

Compañía Coopueblo

La compañía para cual se desarrolló el sistema consta de varios departamentos, los cuales están distribuidos según se muestra en la siguiente figura:

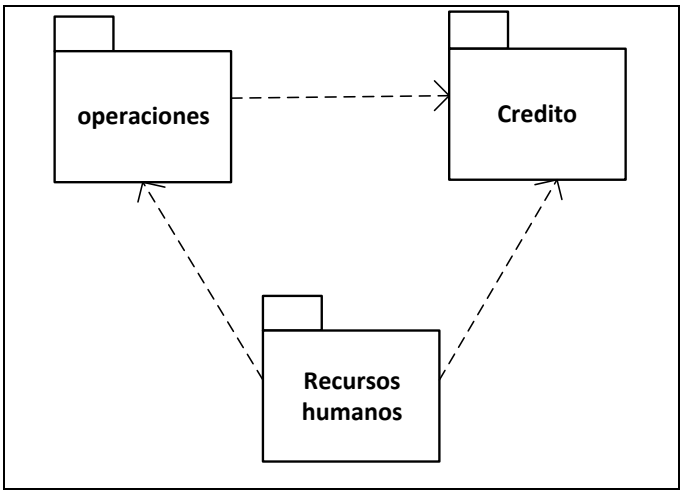


Figura 6: Departamentos de la compañía

Fuente: Tesistas

3.3.1 REQUISITOS DE LA COMPAÑÍA

Un cliente debe cumplir con varios requisitos antes de convertirse en socio de la compañía.

Nuevo socio

Para que el usuario del sistema ingrese un nuevo socio, debe seguir algunos pasos, tal como se describe en el siguiente diagrama.

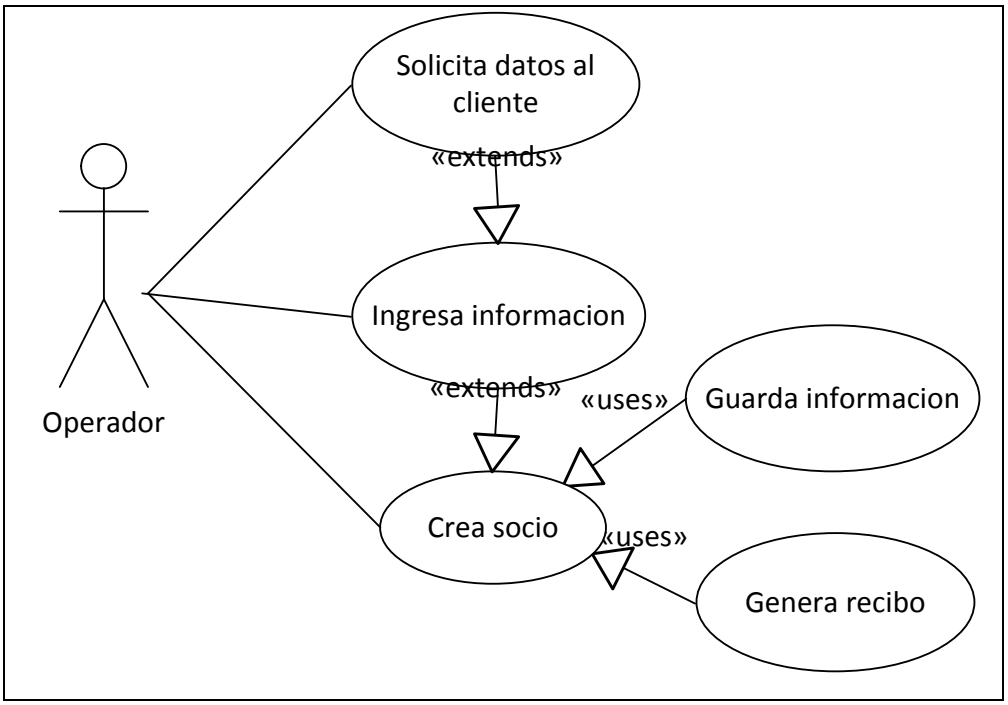


Figura 7: Ingreso de un nuevo socio

Fuente: Tesistas

En el diagrama anterior, se observa como el operador del sistema ingresara un nuevo socio, para ello realiza los siguientes pasos:

1. Solicita la información a la persona a convertirse en socio de la compañía.
2. Ingresa la información en el formulario.
3. En caso de que la persona desee cancelar el valor estipulado por la compañía para convertirse en socio, cancela ese valor, o simplemente se ingresan sus datos, para que luego pueda cancelar ese valor.
4. Guarda la información en el sistema.
5. El sistema genera un recibo, en el cual consta su inscripción como nuevo socio de la compañía, además de sus datos personales, así como también un código que le otorga el sistema.

Buscar socio.

El sistema proporciona una opción de búsqueda de todos los socios de la compañía, de esta manera el usuario del sistema, puede realizar búsquedas de los socios, para fines que considere necesarios.

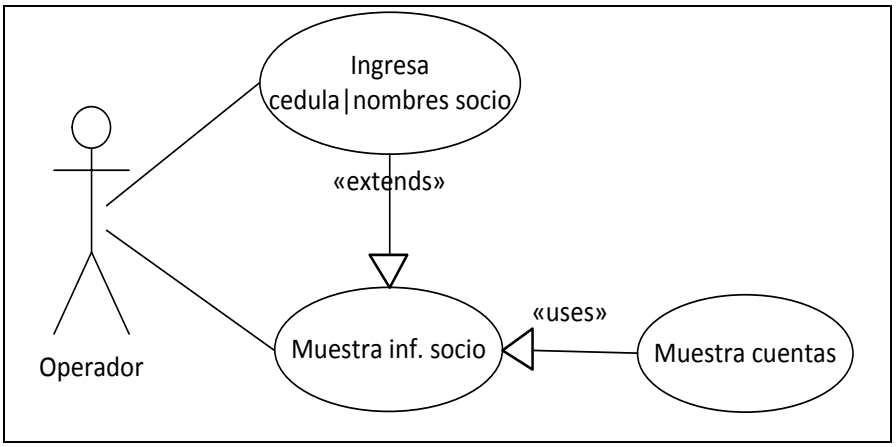


Figura 8: Búsqueda de socio

Fuente: Tesistas

Para buscar un socio en el sistema, el operador tiene dos opciones, ya sea por los nombres y apellidos del socio, o por su número de cédula, en los dos casos, el operador realiza las siguientes acciones:

1. El operador ingresa los apellidos y nombres del socio o su numero de cedula.
2. El sistema muestra la información del socio en mención como: los datos del socio, la fecha en que se convirtió en socio de la compañía, etc. Y las cuentas que tiene creado el socio.

De esta manera el operador obtiene información valiosa del socio, como por ejemplo: las transacciones que ha realizado, los saldos que tiene el socio en cada cuenta, y además tiene la opción de realizar nuevos aportes a partir de estos datos.

Cabe mencionar que al momento de ingresar los datos del socio, inmediatamente el sistema le mostrará ayudas al operador, ya sea de sus apellidos y nombres, o si fuere el caso de su número de cedula. De esta manera el operador del sistema puede elegir el socio indicado con las opciones que le muestra el sistema.

Editar socio

La edición de socios en el sistema, se utiliza principalmente para actualizar datos de los socios de la compañía. Puede ser necesario actualizar los datos de los socios por diversos motivos, como por ejemplo: por cambio de domicilio, por cambios en sus números telefónicos, entre otros. Así se mantiene la información siempre actualizada en el sistema y se lleva un control adecuado de los datos de los socios de la compañía.

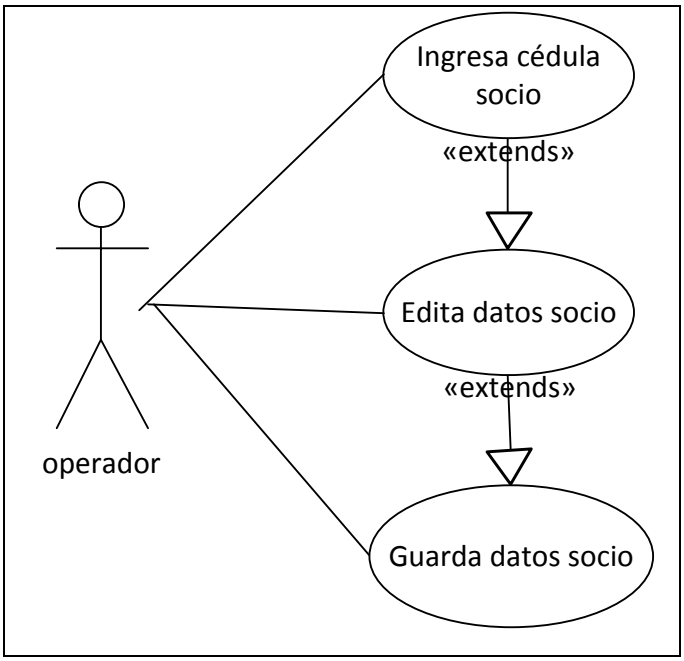


Figura 9: Edición de socios

Fuente: Tesistas

Para que el operador edite los datos del socio, realiza las siguientes acciones:

1. Ingresa la cédula del socio como opción de búsqueda, se desplegaran los datos de la persona, si los datos no son correctos el sistema mostrará un mensaje de error en la búsqueda del socio.
2. Una vez que el sistema muestre la información, el operador puede modificar los datos que considere se deban actualizar.
3. Finalmente se guarda la información en el sistema, con lo cual los datos del socio quedarán actualizados en su totalidad.

Creación de nueva cuenta

Un socio de la compañía puede tener más de una cuenta registrada en el sistema, debido a que el socio puede adquirir más de un bien, por este motivo se le debe crear una cuenta para cada bien, ya que los abonos que el socio realice pueden ser a cualquiera de sus bienes, o ya sea para gastos administrativos, o de obra según considere la compañía.

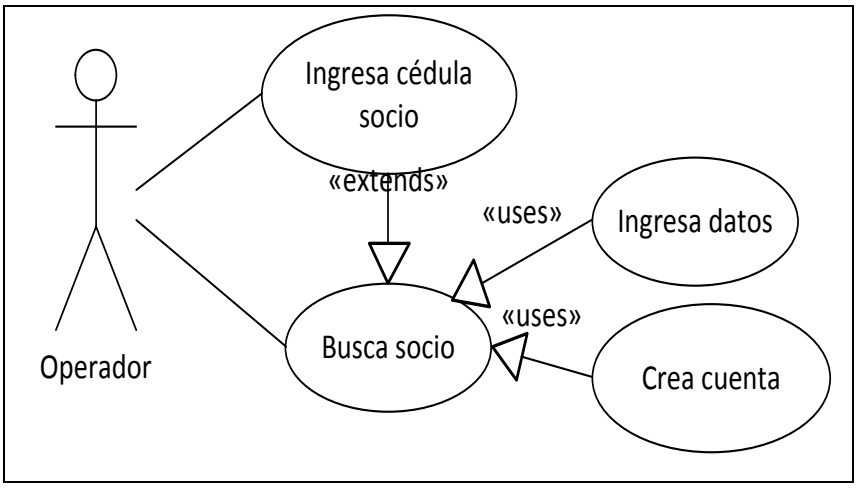


Figura 10: Creación de nueva cuenta

Fuente: Tesistas

Para la creación de una cuenta, el operador del sistema realiza las siguientes acciones.

1. Ingresa la cédula del socio como opción de búsqueda.
2. El sistema muestra la información del socio, en caso de que ya tenga alguna cuenta, muestra la información de esta cuenta, a partir de aquí, el usuario selecciona la creación de una nueva cuenta.
3. Ingresa los datos de la nueva cuenta, así como el monto que va a abonar a dicha cuenta, y.
4. Finalmente ingresa la información en el sistema y el socio ahora dispone de una cuenta.

Cabe mencionar que: si una persona se registró como socio de la compañía, no necesariamente implica que tenga una cuenta creada en el sistema. La creación de la cuenta se lo realiza cuando el socio está interesado en la adquisición de un bien, de esta manera realizará aportes a dicha cuenta, hasta que complete el valor total del bien.

Nuevo depósito

Una vez que el socio de la compañía, ya tenga creada una cuenta, podrá realizar abonos a dichas cuentas, o bien realizar abonos por gastos administrativos y obras, según lo disponga compañía.

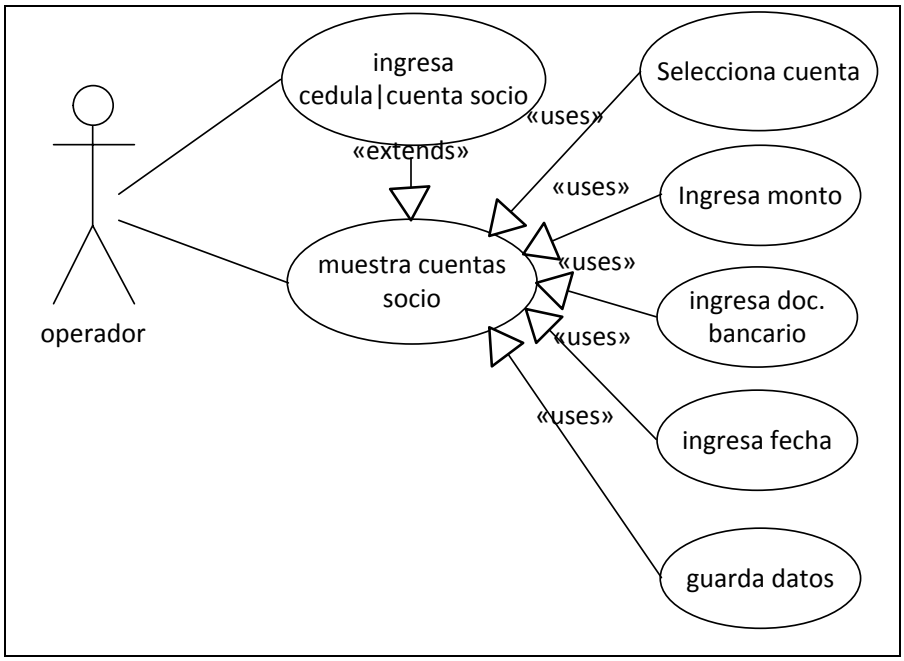


Figura 11: Ingreso de nuevo depósito

Fuente: Tesistas

En el diagrama se observa los pasos que el operador del sistema debe realizar para ingresar un abono a una cuenta. Estos pasos son los siguientes:

1. El operador busca los datos del socio, es decir hace una búsqueda del socio en el sistema, mediante su número de cédula.
2. El sistema muestra los datos del socio, con las cuentas que este tenga, en caso de no tener ninguna se procede con la creación de la cuenta.
3. El operador elige la cuenta del usuario a la cual se realizará el abono.
4. Se ingresa el monto.
5. Previo el abono a la cuenta, el usuario debió realizar un depósito bancario, así que se ingresa ese código de depósito bancario.
6. Se ingresa la fecha en la que fue realizada el abono, y.
7. Finalmente se guardan los datos en el sistema.

Cancelar depósito.

Un depósito se puede cancelar debido a que el cliente quiere abonar un monto a otra cuenta, por motivos de digitación, o por cualquier otro motivo.

En cualquier caso, el operador del sistema deberá especificar el motivo por el cual se va a cancelar el depósito.

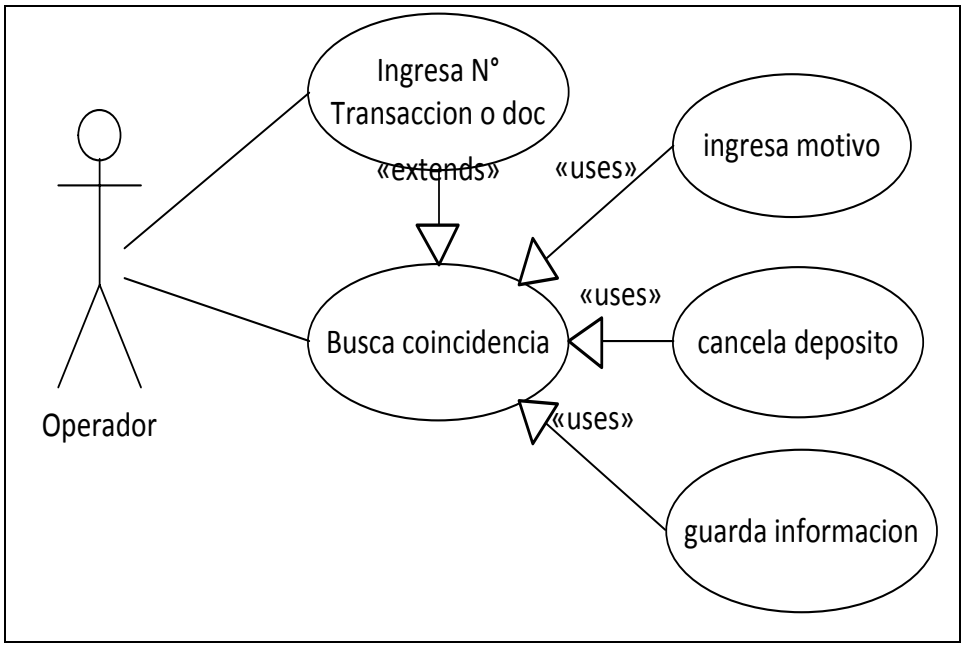


Figura 12: Cancelación de depósitos

Fuente: Tesistas

El diagrama muestra la forma en que se cancela el depósito de un socio.

1. Se ingresa el numero de transacción que fue entregado al socio al momento de realizar el abono, o el numero de documento bancario que el socio entrego en el momento de ingresar ese abono a la cuenta.
2. El sistema busca la coincidencia de estos documentos.
3. El operador debe ingresa el motivo por el cual se va a cancelar el depósito.
4. Se cancela el depósito, y.
5. Finalmente se guarda la información en el sistema.

Puede darse el caso de que el socio simplemente decida recuperar al abono que ha realizado y no abonarlo a ninguna cuenta. En este caso la compañía le entregará su valor sin ningún tipo de recargo ni descuento.

Consulta de saldos

Como ya se a mencionado, el operador puede obtener toda la información de un socio de la compañía, mediante el ingreso de su número de cédula o sus nombres y apellidos en la sección de búsqueda de socios, en esta sección podrá los detalles acerca de una cuenta específica.

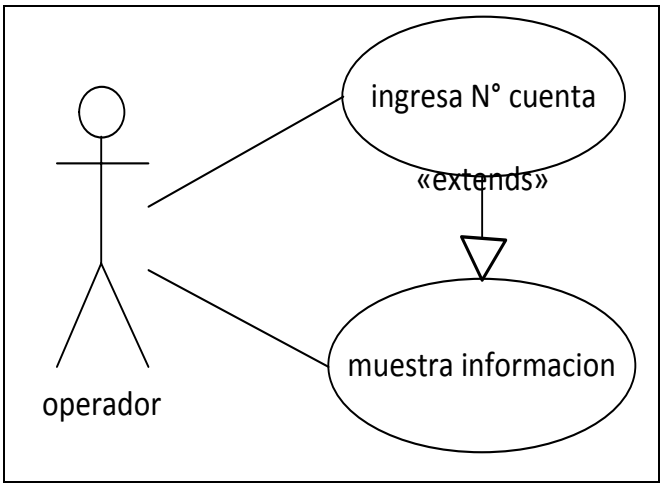


Figura 13: Consulta de saldos

Fuente: Tesistas

Para consultar los saldos de los socios, el operador debe digitar el número de cuenta del socio, la misma que fue entregada en el momento de su creación.

Es importante realizar esta búsqueda mediante el número de cuenta, ya que un socio puede tener mas de una cuenta, y si se realizará una búsqueda mediante su cédula como sucede en otras secciones, el sistema mostrará toda la información, es decir todas las cuentas que el socio tenga creadas.

Por este motivo se le entrega al cliente un número de cuenta único al momento de la creación de una cuenta. El sistema realiza la búsqueda de la cuenta específica en dos sencillos pasos:

1. Se ingresa el número de cuenta del socio.
2. Se muestra la información de dicha cuenta.

Es importante en todo momento que el socio conozca los códigos de sus cuentas, o a su vez tenerlas por escrito en un documento, como el que se lo entrega en el momento de su inscripción.

Solicitud de retiro.

Puede darse el caso de que una persona decida no seguir mas como socio de la compañía, decida retirar todos sus abonos de las cuentas que tenga hasta el momento. O si fuera el caso, puede retirar sus abonos solo de una de sus cuentas, para este motivo, se le deberá generar una solicitud.

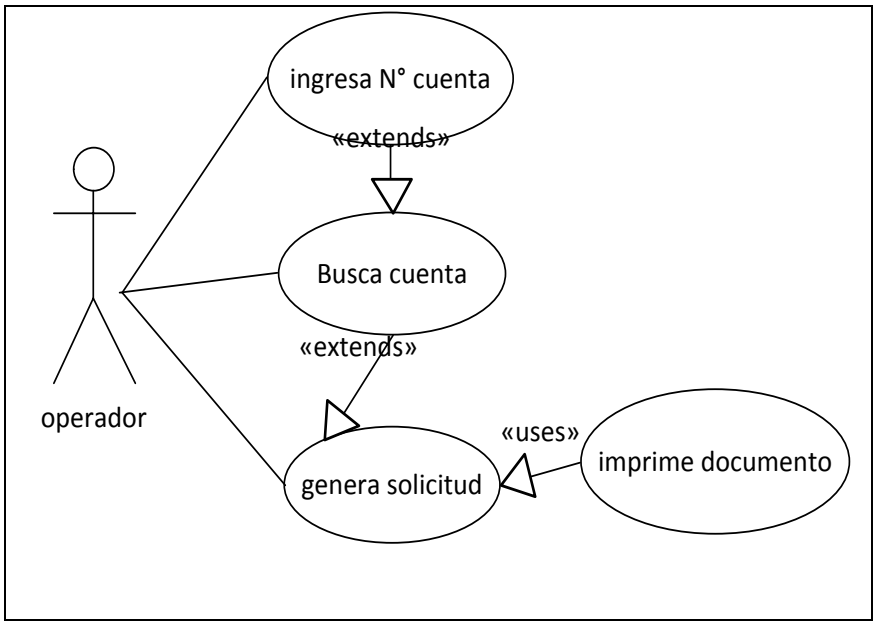


Figura 14: Solicitud de retiro

Fuente: Tesistas

En el diagrama se observa los pasos que el operador del sistema debe seguir para crear una solicitud de retiro de saldos de los socios. Estos pasos son:

1. El operador ingresa un número de cuenta, del cual se va a hacer el retiro.
2. El sistema muestra toda la información referente a esa cuenta.
3. El sistema genera una solicitud, con un número único respectivo.
4. Se imprime la solicitud, para que esta sea aprobada.

El sistema generará la solicitud con un número único. Esta solicitud servirá para que sea aprobada por su respectivo personal administrativo, en este caso la gerente de la compañía.

Para cualquier tipo de retiro, ya sea retiro de saldos o retiro definitivo como socio de la compañía, se requiere la aprobación de la gerente de la "compañía de promoción y desarrollo humano y social coopueblo". Es por este motivo que el sistema siempre genera una solicitud con un número único.

Retiro de saldos

Como se mencionó en el apartado anterior, un socio puede retirar sus abonos de una cuenta específica, para lo cual se le debió generar una solicitud. En este punto la gerente de la compañía, aprueba o no las solicitudes de los socios.

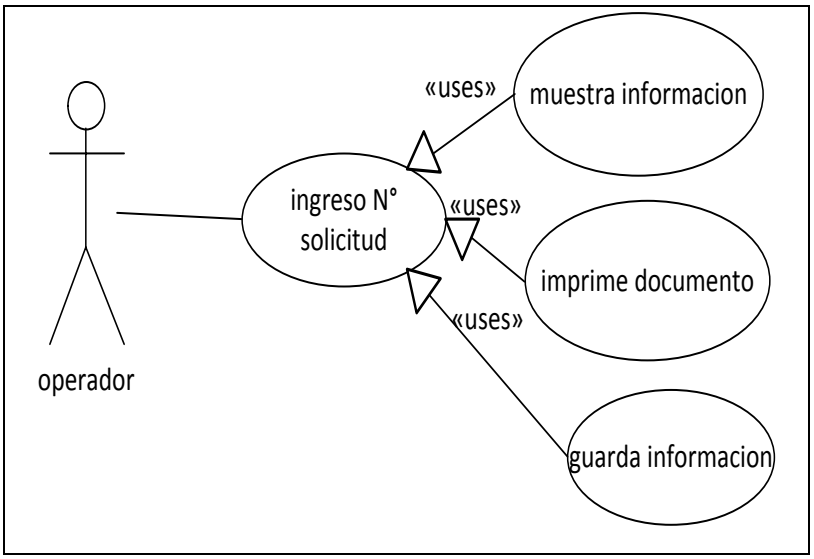


Figura 15: Retiro de saldos

Fuente: Tesistas

Para generar la solicitud de retiro de saldos, el operador del sistema debe seguir los siguientes pasos:

1. Ingresar el número de solicitud que se le entrego en el apartado anterior
2. Se genera la solicitud.
3. Se imprime la solicitud
4. Y se guarda la información en el sistema.

En este punto la gerente de la compañía, aprobará o no la solicitud de retiro de un socio. Para ello el socio deberá proporcionar la gerente, el número de solicitud que se le otorgó en solicitudes de retiro. Si no se encuentra ninguna coincidencia el sistema mostrará un mensaje de error.

Por políticas de la institución, no se descuenta absolutamente nada del monto que un socio haya aportado hasta ese momento, es decir si realizó dos abonos de mil dólares cada uno, se le entregará los dos mil dólares que ha aportado hasta ese momento sin ningún tipo de descuento, siempre y cuando los depósitos hayan sido para abonos del bien.

Si los depósitos se realizaron por motivos como: obras o gastos administrativos, este valor no le será devuelto al socio.

Solicitud de transferencias

Uno de los aspectos más importantes y novedosos de este sistema, es que permite a los socios realizar transferencias a cuentas de otros socios de la compañía. Para ello necesita de la aprobación de la gerente de la compañía.

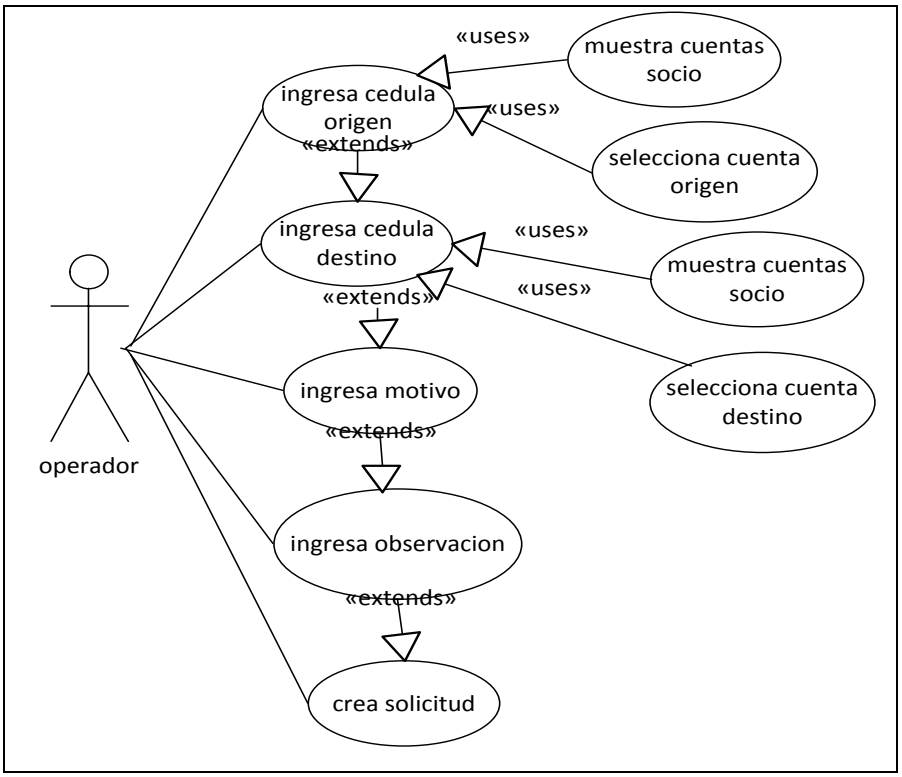


Figura 16: Solicitud de transferencias

Fuente: Tesistas

Para generar una solicitud de transferencia, se siguen los siguientes pasos.

1. Se realiza la búsqueda de las cuentas del socio de origen (socio que va a transferir saldo). Mediante su número de cedula.
2. Se ingresa la cédula del socio de destino (socio al cual se le va a trasferir el saldo).
3. Ingreso de un motivo por el cual se va a transferir el saldo de una cuenta.
4. Se ingresa una observación de la transacción que se va a realizar.
5. Posteriormente se genera la respectiva solicitud para que sea aprobada por la gerente de la compañía.

Al momento de realizar la búsqueda por la cédula respectiva de cada socio, el sistema mostrará las cuentas que cada socio tiene. Así se podrá elegir la cuenta de origen y la cuenta de destino para que la transacción se realice correctamente.

Para que esta operación se realice con éxito, es necesario que tanto el beneficiario como el socio que traspasa un saldo, tengan creadas cuentas en el sistema, caso contrario no se podrá realizar la operación.

Transferencia de saldos.

En este punto los saldos de los socios serán traspasados, para ello es importante que la solicitud haya sido generada en el apartado anterior.

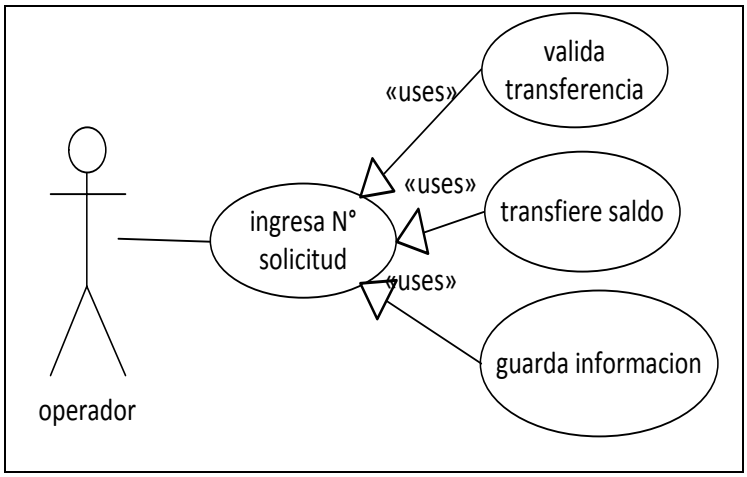


Figura 17: Transferencia de saldos

Fuente: Tesistas

El diagrama muestra los pasos que se deben seguir para realizar una transferencia de saldos.

1. Ingresar el número de solicitud generado en el punto anterior (Solicitud de transferencia).
2. La gerente de la compañía aprueba la solicitud.
3. Se transfiere los saldos entre los socios, y
4. Finalmente se guarda la información.

De esta manera un socio puede transferir la totalidad de su saldo a otro, o a su vez parte de su saldo de alguna de sus cuentas hacia otro socio. Este proceso de transferencia es de vital importancia para la compañía, ya que cuando no se disponía de un sistema informático, los trámites podían tornarse demasiado largos, lo cual podía generar un ambiente de desconfianza entre sus socios.

Perfiles.

El sistema permite crear perfiles. En el sistema se encuentran cinco perfiles principales, que son:

- Administrador.
- Administrador del sistema.
- Operador.
- Contador.
- Gerente.

Pero se pueden crear los que se desee.

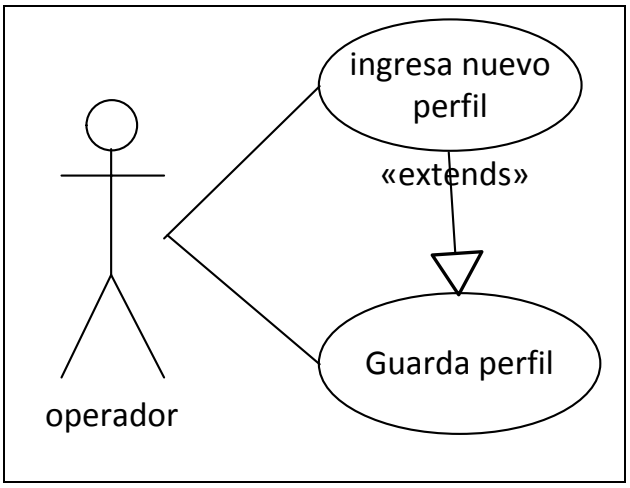


Figura 18: Creación de Perfiles

Fuente: Tesistas

La creación de perfiles se lleva a cabo de una manera muy sencilla. En dos simples pasos como se observa en la imagen.

1. El operador ingresa a la creación de perfiles y tiene la opción de agregar un nuevo perfil.
2. Se ingresa un perfil y se guarda los cambios.

La creación de perfiles es importante, ya que luego se puede asignar perfiles a los usuarios del sistema, y dependiendo de estos se otorgará permisos a los distintos usuarios.

Usuarios

Se puede agregar usuarios al sistema, dependiendo de las necesidades que tenga la compañía, de las personas que vayan a utilizarlo, la creación de usuarios se lleva a cabo de una manera muy sencilla.

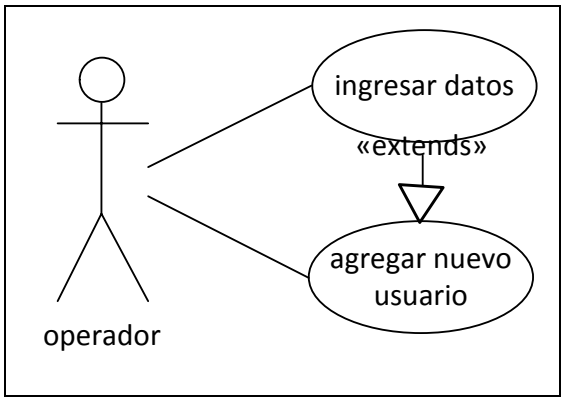


Figura 19: Creación de usuarios

Fuente: Tesistas

Para el ingreso de un nuevo usuario, basta con ingresar a la opción de creación de usuarios, introducir los datos del usuario, y guardarlos en el sistema. Es importante asignarle al usuario el perfil al que pertenece, y darle el estado de Activo o Inactivo.

Roles

La creación de roles se lleva a cabo de forma similar que el ejemplo anterior, se ingresa a la opción de crear rol, y se lo agrega. Cabe mencionar que el sistema existen tres roles principales que son:

- Cooperativa,
- Compañía, y.
- Administración del sistema.

Estos roles son los que se le asignan al usuario para otorgarle los permisos y asignarle únicamente las opciones necesarias para que cumpla con su trabajo.

Asignación de Roles

Se vio la creación de Roles, luego el sistema permite asignar estos roles a los usuarios.

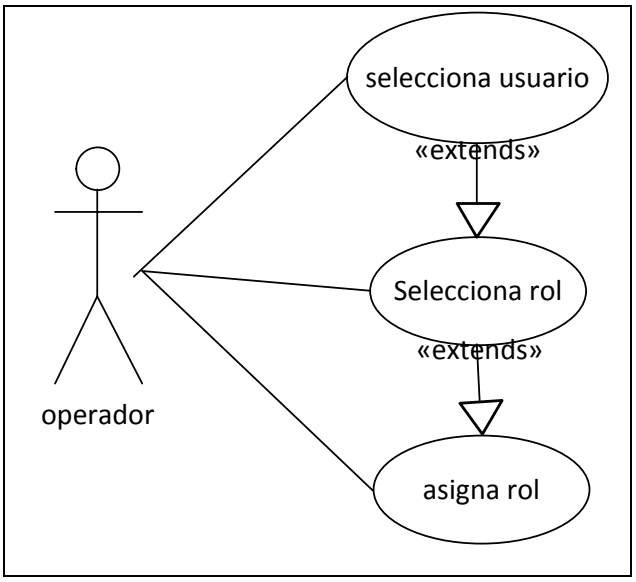


Figura 20: Asignación de Roles

Fuente: Tesistas

La asignación de roles a los usuarios se lleva a cabo de manera muy sencilla.

1. Se selecciona al usuario indicado para asignarle el rol.
2. Se selecciona los roles necesarios
3. Finalmente se le otorga los roles a los usuarios.

En este punto únicamente se ha otorgado los roles en un paso posterior se le asignará todos los permisos necesarios dentro de cada rol.

Asignación de perfiles.

La asignación de perfiles se lleva a cabo de manera similar a la asignación de roles, basta con ingresar a la opción de asignar perfiles, seleccionar el usuario, y asignarle el perfil requerido.

Asignación de permisos

Uno de los aspectos más importantes del sistema es asignar permisos a los usuarios. Dependiendo de los permisos que se otorguen se podrá ver en las cuentas, los enlaces necesarios para realizar su trabajo, según sea su perfil y su cargo asignado en la compañía.

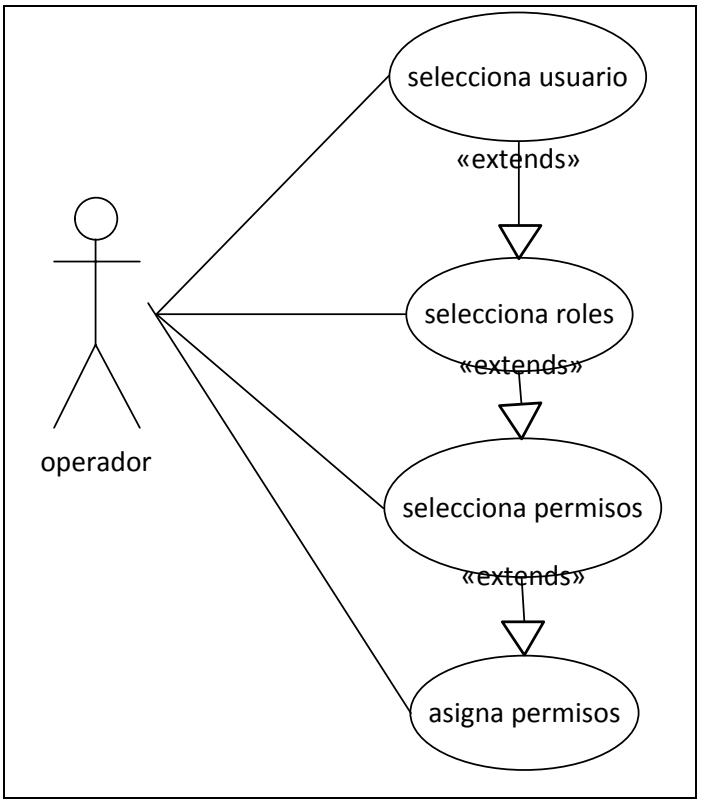


Figura 21: Asignación de permisos

Fuente: Tesistas

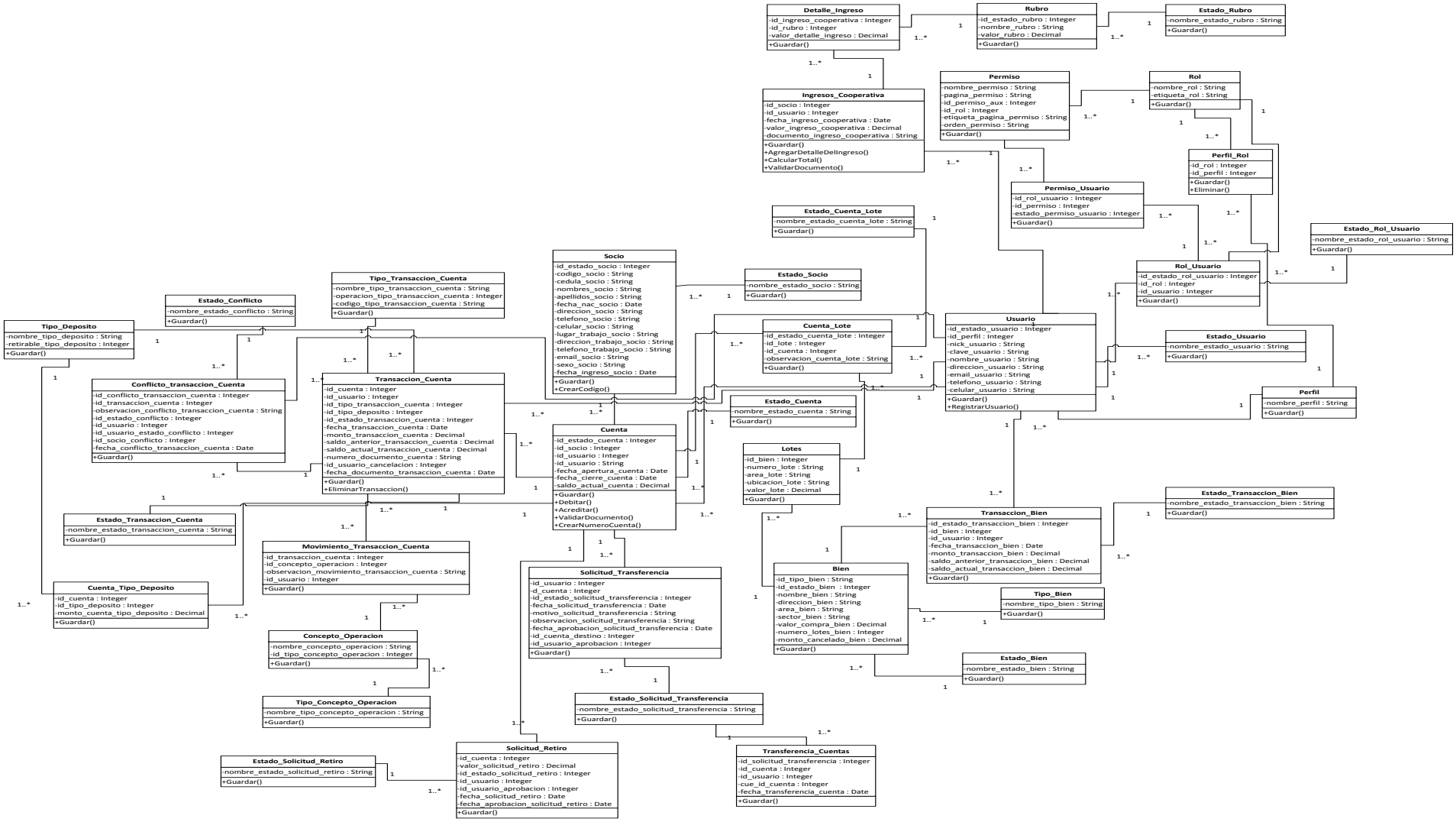
En la figura se observa los pasos necesarios para otorgarle permisos a un usuario. Para ello primero se le debió otorgar el rol respectivo, ya que dentro de estos se encuentran los enlaces que se le otorgan en este punto. Los pasos a seguir son:

1. El operador selecciona el usuario para otorgarle los permisos respectivos.
2. Selecciona el rol o roles que se le ha asignado.
3. Dentro de cada rol se selecciona los enlaces necesarios a los que tendrá acceso.
4. Finalmente los permisos quedan asignados, y se guarda los cambios realizados.

De esta manera se le asigna al usuario los permisos necesarios para que realice su trabajo sin inconvenientes, restringiendo los enlaces a los que no debería acceder, y otorgándole únicamente los necesarios.

3.4 DIAGRAMA DE CLASES

El diagrama de clases describe la forma de cómo se estructura la información de los socios de la compañía. Por tratarse de una compañía que requiere de un alto grado de consistencia y seguridad en los datos, el diagrama que se muestra resulta algo complejo, debido a que no solo almacena información de los socios, sino también proporciona un control adecuado de los usuarios del sistema.


Detalle_Ingreso
-id_ingreso_cooperativa : Integer
-id_rubro : Integer
-valor_detalle_ingreso : Decimal
+Guardar()
Rubro
-id_estado_rubro : Integer
-nombre_rubro : String
-valor_rubro : Decimal
+Guardar()
Estado_Rubro
-nombre_estado_rubro : String
+Guardar()
Ingresos_Cooperativa
-id_socio : Integer
-id_usuario : Integer
-fecha_ingreso_cooperativa : Date
-valor_ingreso_cooperativa : Decimal
-documento_ingreso_cooperativa : String
+Guardar()
+AgregarDetalleDelIngreso()
+CalcularTotal()
+ValidarDocumento()
Permiso
-nombre_permiso : String
-pagina_permiso : String
-id_permiso_aux : Integer
-id_rol : Integer
-etiqueta_pagina_permiso : String
-orden_permiso : String
+Guardar()
Rol
-nombre_rol : String
-etiqueta_rol : String
+Guardar()
Perfil_Rol
-id_rol : Integer
-id_perfil : Integer
+Guardar()
+Eliminar()
Permiso_Usuario
-id_rol_usuario : Integer
-id_permiso : Integer
-estado_permiso_usuario : Integer
+Guardar()
Estado_Cuenta_Lote
-nombre_estado_cuenta_lote : String
+Guardar()
Estado_Rol_Usuario
-nombre_estado_rol_usuario : String
+Guardar()
Rol_Usuario
-id_estado_rol_usuario : Integer
-id_rol : Integer
-id_usuario : Integer
+Guardar()
Socio
-id_estado_socio : Integer
-codigo_socio : String
-cedula_socio : String
-nombres_socio : String
-apellidos_socio : String
-fecha_nac_socio : Date
-direccion_socio : String
-telefono_socio : String
-celular_socio : String
-lugar_trabajo_socio : String
-direccion_trabajo_socio : String
-telefono_trabajo_socio : String
-email_socio : String
-sexo_socio : String
-fecha_ingreso_socio : Date
+Guardar()
+CrearCodigo()
Estado_Socio
-nombre_estado_socio : String
+Guardar()
Tipo_Transaccion_Cuenta
-nombre_tipo_transaccion_cuenta : String
-operacion_tipo_transaccion_cuenta : Integer
-codigo_tipo_transaccion_cuenta : String
+Guardar()
Estado_Conflicto
-nombre_estado_conflicto : String
+Guardar()
Tipo_Deposito
-nombre_tipo_deposito : String
-retirable_tipo_deposito : Integer
+Guardar()
Cuenta_Lote
-id_estado_cuenta_lote : Integer
-id_lote : Integer
-id_cuenta : Integer
-observacion_cuenta_lote : String
+Guardar()
Usuario
-id_estado_usuario : Integer
-id_perfil : Integer
-nick_usuario : String
-clave_usuario : String
-nombre_usuario : String
-direccion_usuario : String
-email_usuario : String
-telefono_usuario : String
-celular_usuario : String
+Guardar()
+RegistrarUsuario()
Estado_Usuario
-nombre_estado_usuario : String
+Guardar()
Perfil
-nombre_perfil : String
+Guardar()
Conflicto_transaccion_Cuenta
-id_conflicto_transaccion_cuenta : Integer
-id_transaccion_cuenta : Integer
-observacion_conflicto_transaccion_cuenta : String
-id_estado_conflicto : Integer
-id_usuario : Integer
-id_usuario_estado_conflicto : Integer
-id_socio_conflicto : Integer
-fecha_conflicto_transaccion_cuenta : Date
+Guardar()
Transaccion_Cuenta
-id_cuenta : Integer
-id_usuario : Integer
-id_tipo_transaccion_cuenta : Integer
-id_tipo_deposito : Integer
-id_estado_transaccion_cuenta : Integer
-fecha_transaccion_cuenta : Date
-monto_transaccion_cuenta : Decimal
-saldo_anterior_transaccion_cuenta : Decimal
-saldo_actual_transaccion_cuenta : Decimal
-numero_documento_cuenta : String
-id_usuario_cancelacion : Integer
-fecha_documento_transaccion_cuenta : Date
+Guardar()
+EliminarTransaccion()
Estado_Cuenta
-nombre_estado_cuenta : String
+Guardar()
Cuenta
-id_estado_cuenta : Integer
-id_socio : Integer
-id_usuario : Integer
-id_usuario : String
-fecha_apertura_cuenta : Date
-fecha_cierre_cuenta : Date
-saldo_actual_cuenta : Decimal
+Guardar()
+Debitar()
+Acreditar()
+ValidarDocumento()
+CrearNumeroCuenta()
Lotes
-id_bien : Integer
-numero_lote : String
-area_lote : String
-ubicacion_lote : String
-valor_lote : Decimal
+Guardar()
Estado_Transaccion_Bien
-nombre_estado_transaccion_bien : String
+Guardar()
Transaccion_Bien
-id_estado_transaccion_bien : Integer
-id_bien : Integer
-id_usuario : Integer
-fecha_transaccion_bien : Date
-monto_transaccion_bien : Decimal
-saldo_anterior_transaccion_bien : Decimal
-saldo_actual_transaccion_bien : Decimal
+Guardar()
Estado_Transaccion_Cuenta
-nombre_estado_transaccion_cuenta : String
+Guardar()
Movimiento_Transaccion_Cuenta
-id_transaccion_cuenta : Integer
-id_concepto_operacion : Integer
-observacion_movimiento_transaccion_cuenta : String
-id_usuario : Integer
+Guardar()
Solicitud_Transferencia
-id_usuario : Integer
-d_cuenta : Integer
-id_estado_solicitud_transferencia : Integer
-fecha_solicitud_transferencia : Date
-motivo_solicitud_transferencia : String
-observacion_solicitud_transferencia : String
-fecha_aprobacion_solicitud_transferencia : Date
-id_cuenta_destino : Integer
-id_usuario_aprobacion : Integer
+Guardar()
Bien
-id_tipo_bien : String
-id_estado_bien : Integer
-nombre_bien : String
-direccion_bien : String
-area_bien : String
-sector_bien : String
-valor_compra_bien : Decimal
-numero_lotes_bien : Integer
-monto_cancelado_bien : Decimal
+Guardar()
Tipo_Bien
-nombre_tipo_bien : String
+Guardar()
Estado_Bien
-nombre_estado_bien : String
+Guardar()
Cuenta_Tipo_Deposito
-id_cuenta : Integer
-id_tipo_deposito : Integer
-monto_cuenta_tipo_deposito : Decimal
+Guardar()
Concepto_Operacion
-nombre_concepto_operacion : String
-id_tipo_concepto_operacion : Integer
+Guardar()
Tipo_Concepto_Operacion
-nombre_tipo_concepto_operacion : String
+Guardar()
Estado_Solicitud_Transferencia
-nombre_estado_solicitud_transferencia : String
+Guardar()
Transferencia_Cuentas
-id_solicitud_transferencia : Integer
-id_cuenta : Integer
-id_usuario : Integer
-cue_id_cuenta : Integer
-fecha_transferencia_cuenta : Date
+Guardar()
Estado_Solicitud_Retiro
-nombre_estado_solicitud_retiro : String
+Guardar()
Solicitud_Retiro
-id_cuenta : Integer
-valor_solicitud_retiro : Decimal
-id_estado_solicitud_retiro : Integer
-id_usuario : Integer
-id_usuario_aprobacion : Integer
-fecha_solicitud_retiro : Date
-fecha_aprobacion_solicitud_retiro : Date
+Guardar()

3.5 MODELO RELACIONAL

El sistema desarrollado permite el control de usuarios, esto ayuda a verificar que únicamente los usuarios autorizados accedan a los módulos que les han sido asignados. El siguiente diagrama describe la forma de como se controla el acceso de los usuarios al sistema.

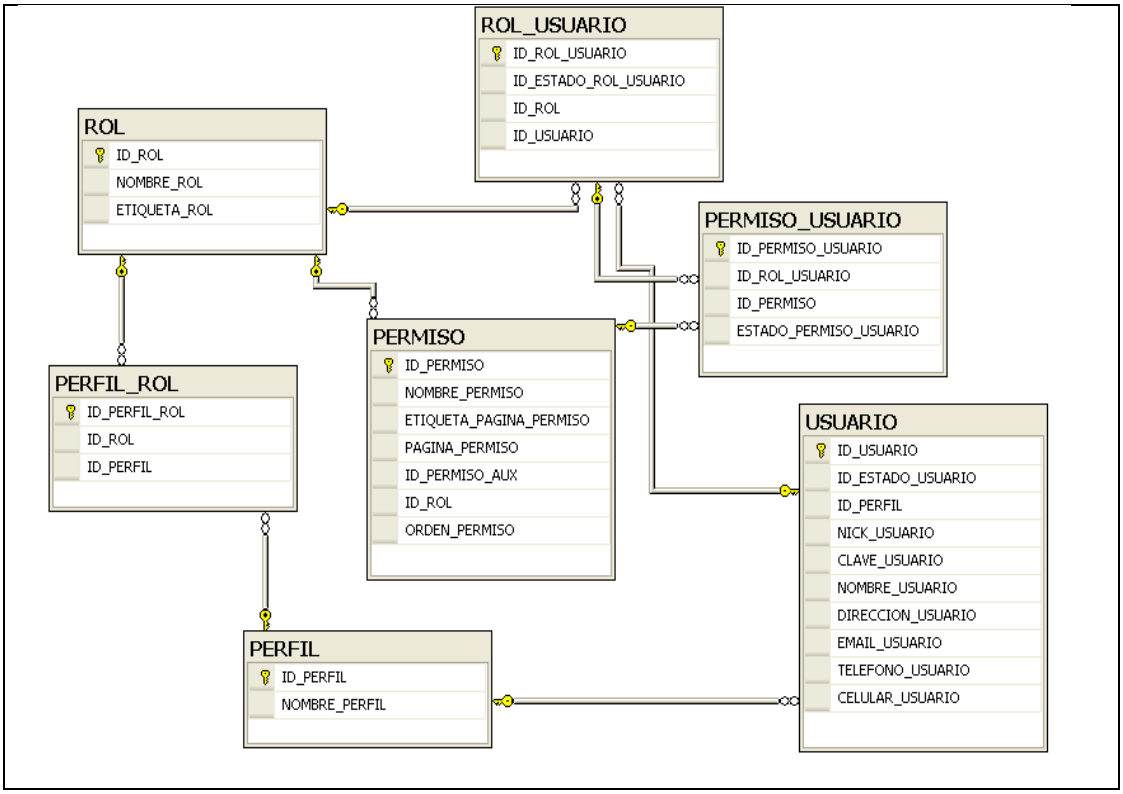


Figura 23: Modelo de datos

Fuente: Tesistas

A los usuarios se les asigna roles, y dentro de estos roles se encuentran las tareas que puede realizar. De igual manera a un usuario se le asigna un perfil de acuerdo a su posición dentro de la compañía.

3.6 DICCIONARIO DE DATOS

Nombre de Archivo: CUENTA		**Fecha de creación:** 07/02/2012
Descripción: Archivo principal de cuentas, almacenara información sobre las cuentas de los socios		
Campo	**Tipo**	**Descripción**
ID_CUENTA	int	Número de identificación de la cuenta
ID_ESTADO_CUENTA	int	Almacena el estado de la cuenta
ID_SOCIO	int	Identificador del socio
ID_USUARIO	int	Identificador del usuario
NUMERO_CUENTA	varchar	Identifica el número de cuenta creada
FECHA_APERTURA_CUENTA	datetime	Fecha en que se crea la cuenta
FECHA_CIERRE_CUENTA	datetime	Fecha en que se cierra la cuenta
SALDO_ACTUAL_CUENTA	numeric	Saldo que el socio a abonado hasta la fecha
Campos clave: ID_CUENTA, ID_ESTADO_CUENTA, ID_SOCIO, ID_USUARIO		

Tabla 1: Detalles de la tabla CUENTA

Fuente: Tesistas

Nombre de Archivo: TRANSACCION_BIEN **Fecha de creación:** 07/02/2012 **Descripción**: Archivo que describe las transacciones de de bienes de los socios		
Campo	**Tipo**	**Descripción**
ID_TRANSACCION_BIEN	int	Describe un identificador la transacción
ID_ESTADO_TRANSACCION_BIEN	int	Estado de la transacción de los socios
ID_BIEN	int	Identificador del bien
ID_USUARIO	int	Usuario que realiza la transacción
FECHA_TRANSACCION_BIEN	datetime	Fecha de la transacción
MONTO_TRANSACCION_BIEN	numeric	Monto de la transacción
SALDO_ANTERIOS_TRANSACCION_BIEN	numeric	Saldo anterior de la trasferencia
SALDO_ACTUAL_TRANSACCION_BIEN	numeric	Saldo actual de la transacción
Campos clave: ID_TRANSACCION_BIEN		

Tabla 2: Detalles tabla TRANSACCION_BIEN

Fuente: Tesistas

Nombre de Archivo: USUARIO		**Fecha de creación:** 07/02/2012
Descripción: Almacena la información referente los usuarios del sistema		
Campo	**Tipo**	**Descripción**
ID_USUARIO	int	Número de identificación del usuario
ID_ESTADO_USUARIO	int	Describe si es un usuario activo o no
ID_PERFIL	int	Describe el perfil del usuario
NICK_USUARIO	varchar	Almacena un nombre que identifica al usuario
CLAVE_USUARIO	varchar	Almacena la contraseña del usuario
NOMBRE_USUARIO	varchar	Identifica el nombre real del usuario
DIRECCION_USUARIO	varchar	Almacena la dirección del domicilio del usuario
EMAIL_USUARIO	varchar	Almacena el email
TELEFONO_USUARIO	varchar	Almacena el teléfono fijo del usuario
CELULAR_USARIO	varchar	Almacena el numero de celular del usuario
Campos clave: ID_USUARIO		

Tabla 3: Detalles tabla USUARIO

Fuente: Tesistas

Nombre de Archivo: LOTES		**Fecha de creación:** 07/02/2012
Descripción: Archivo que describe los detalles de los bienes		
Campo	**Tipo**	**Descripción**
ID_LOTE	int	Describe un identificador del lote
ID_BIEN	int	Almacena el identificador del tipo del bien
NUMERO_LOTE	varchar	Almacena el identificador del lote
AREA_LOTE	varchar	Extensión del lote
UBICACIÓN_LOTE	varchar	Ubicación física del lote
VALOR_LOTE	numeric	Costo total del lote
Campos clave: ID_LOTE		

Tabla 4: Detalles tabla LOTES

Fuente: Tesistas

Nombre de Archivo: SOLICITUD_TRANSFERENCIA **Fecha de creación:** 07/02/2012
Descripción: Archivo que describe los detalles de las solicitudes de transferencia

Campo	Tipo	Descripción
ID_SOLICITUD_TRANSFERENCIA	int	Describe un identificador de la solicitud
ID_USUARIO	int	Almacena el id del usuario
ID_CUENTA	int	Id de la cuenta de la transferencia
ID_ESTADO_SOLICITUD_TRANSFERENCIA	int	Estado de la solicitud
FECHA_SOLICITUD_TRANSFERENCIA	datetime	Fecha de elaboración de la solicitud
MOTIVO_SOLICITUD_TRANSFERENCIA	text	Motivo de la solicitud
OBSERVACION_SOLICITUD_TRANSFERENCIA	varchar	Observación de la solicitud
FECHA_APROBACION_SOLICITUD_TRANSFERENCIA	datetime	Fecha en que se aprobó la solicitud
ID_CUENTA_DESTINO	int	Cuenta de destino de la transferencia
ID_USUARIO_APROBACION	int	Usuario que gestionó la solicitud

Campos clave: ID_SOLICITUD_TRANSFERECIA

Tabla 5: Detalles tabla SOLICITUD_TRANSFERENCIA
Fuente: Tesistas

Nombre de Archivo: BIEN		**Fecha de creación:** 07/02/2012
Descripción: Archivo que describe los detalles del bien inmueble		
Campo	**Tipo**	**Descripción**
ID_BIEN	int	Describe un identificador del bien
ID_TIPO_BIEN	int	Almacena el identificador del tipo del bien
ID_ESTADO_BIEN	int	Almacena el identificador del bien
DIRECCION_BIEN	varchar	Dirección física donde se encuentra el bien
AREA_BIEN	varchar	Almacena el área física del bien
SECTOR_BIEN	datetime	Lugar en donde se encuentra el bien inmueble
VALOR_COMPRA_BIEN	datetime	Valor total del bien
NUMERO_LOTES_BIEN	int	Cantidad del lotes que posee el bien
MONTO_CANCELADO_BIEN	decimal	Monto abonado al bien hasta el momento
NOMBRE_BIEN	varchar	Descripción del bien
Campos clave: ID_BIEN		

Tabla 6: Detalles tabla BIEN

Fuente: Tesistas

Nombre de Archivo: SOCIO **Fecha de creación:** 07/02/2012
Descripción: Archivo que almacena la información referente a los socios de la compañía

Campo	Tipo	Descripción
ID_SOCIO	int	Número que identifica al socio
ID_ESTADO_SOCIO	int	Almacena información de del socio
CODIGO_SOCIO	varchar	Valor único que identifica al socio
CEDULA_SOCIO	varchar	Almacena el número de cedula del socio
NOMBRES_SOCIO	varchar	Almacena el nombre real del socio
APELLIDOS_SOCIO	varchar	Apellidos completos del socio
FECHA_NAC_SOCIO	datetime	Almacena fecha de nacimiento del socio
DIRECCION_SOCIO	varchar	Dirección del domicilio del socio
TELEFONO_SOCIO	varchar	Teléfono fijo del socio
CELULAR_SOCIO	varchar	Numero del celular del socio
LUGAR_TRABAJO_SOCIO	varchar	Empresa o compañía donde trabaja el socio
DIRECCION_TRABAJO_SOCIO	varchar	Dirección física del lugar de trabajo del socio
TELEFONO_TRABAJO_SOCIO	varchar	Teléfono fijo del socio
EMAIL_SOCIO	varchar	Dirección de correo electrónico
SEXO_SOCIO	varchar	Almacena el genero del socio
FECHA_INGRESO_SOCIO	datetime	Fecha en que se convirtió en socio del compañía
Campos clave: ID_SOCIO		

Tabla 7: Detalles tabla SOCIO
Fuente: Tesistas

Nombre de Archivo: ROL_USUARIO **Fecha de creación:** 07/02/2012 **Descripción:** Archivo que describe los roles de los usuarios		
Campo	**Tipo**	**Descripción**
ID_ROL_USUARIO	int	Describe un identificador del lote
ID_ESTADO_ROL_USUARIO	int	Almacena el identificador del tipo del bien
ID_ROL	int	Almacena el identificador del lote
ID_USUARIO	int	Extensión del lote
Campos clave: ID_ROL_USUARIO		

Tabla 8: Detalles tabla ROL_USUARIO

Fuente: Tesistas

Nombre de Archivo: SOLICITUD_RETIRO **Fecha de creación:** 07/02/2012 **Descripción:** Archivo que describe los roles de los usuarios		
Campo	**Tipo**	**Descripción**
ID_SOLICITUD_RETIRO	int	Describe un identificador del lote
ID_CUENTA	int	Almacena el identificador del tipo del bien
VALOR_SOLICITUD_RETIRO	decimal	Almacena el identificador del lote
ID_ESTADO_SOLICITUD_RETIRO	int	Extensión del lote
ID_USUARIO	int	Identificación del usuario que ingresó la solicitud
ID_USUARIO_APROBACION	int	Identificación del usuario que aprobó la solicitud
FECHA_SOLICITUD_RETIRO	datetime	Fecha de ingreso de la solicitud
FECHA_APROBACION_SOLICITUD_RETIRO	datetime	Fecha que se aprobó la solicitud
Campo clave: ID_SOLICITUD_RETIRO		

Tabla 9: Detalles tabla SOLICITUD_RETIRO

Fuente: Tesistas

CAPITULO IV

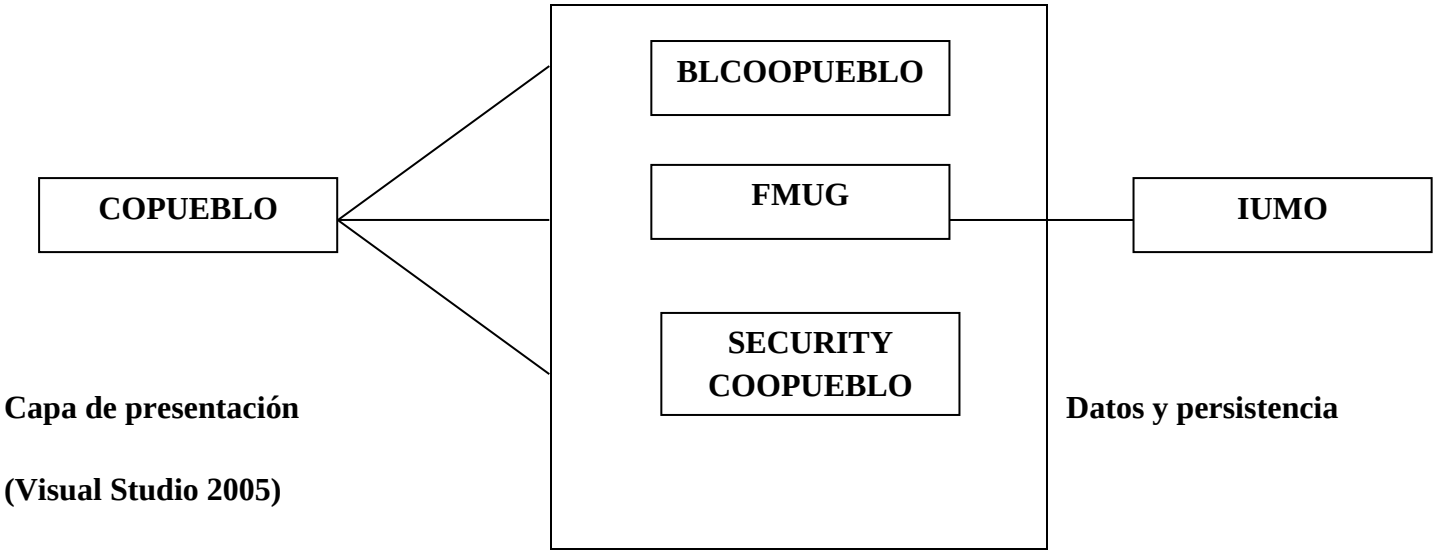
DESARROLLO E IMPLEMENTACION

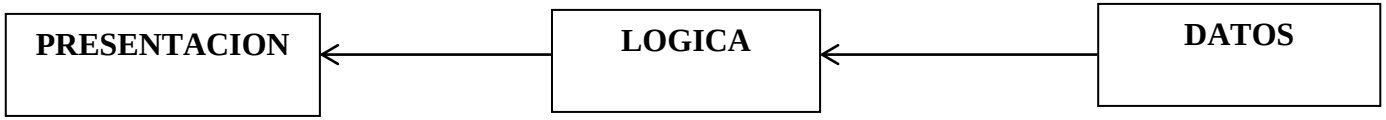
Para el desarrollo del sistema se considerará lo propuesto por la metodología XP, con la finalidad de obtener diseños simples y sencillos y que cumplan con las expectativas del usuario del sistema

4.1 COMPONENTES DEL SISTEMA

En el sistema informático para la Compañía de Promoción y Desarrollo Humano y Social Coopueblo, se presentan tres librerías o componentes principales, estas librerías contienen a su vez todo el proceso de software implementado. Debido a que el sistema está basado en la arquitectura web en capas, tal y como se menciona en el capítulo II (Marco teórico), las capas de presentación y de negocio están embebidas dentro de estas librerías (capa de aplicación, capa de negocio), la tercera capa corresponde a la capa de datos, la misma que está desarrollada en SQL Server 2005 tal y como se menciona en el capítulo dos.

En el diagrama siguiente se demuestra cómo están estructuradas estas capas:


BLCOOPUEBLO
COPUEBLO
FMUG
IUMO
SECURITY
COOPUEBLO
Capa de presentación
(Visual Studio 2005)
Datos y persistencia
Lógica y seguridad


PRESENTACION
LOGICA
DATOS

4.2 CAPA DE PRESENTACIÓN

La capa de presentación corresponde a las interfaces que se crearon para el sistema. Como se observa en el diagrama anterior, la capa de presentación está compuesta por todas las pantallas que contiene el sistema.

4.3 CAPA DE LÓGICA

La capa de lógica corresponde a toda la programación del aplicativo en sí. Para ello se han creado tres componentes o librerías o componentes las cuales se detallan a continuación.

4.3.1 BLCOOPUEBLO

En esta librería, se encuentra la capa de la lógica del negocio del sistema informático, es decir aquí se encuentran las clases que forman este sistema, en las cuales está el código que ha sido escrito para que la aplicación funcione de la forma que se esperaba.

Además dentro de esta librería, se encuentra otra librería llamada. IUMO en la cual se encuentra el acceso a la base de datos mediante ADO.NET. Descrito en el capítulo II (Marco teórico).

La conexión con la base de datos, se lo realiza en la clase *webconfig.vb*en donde se ingresa la cadena de conexión. Esto se observa en la siguiente línea de código.

```
<add connectionString="Data Source=GONZALO-C7F8854\SA;Initial
Catalog=copueblo;User ID=sa;Password=sa.1" name="DataBase"
providerName="System.Data.SqlClient"/>
```

En el bloque de código anterior se observa la cadena de conexión con la base de datos, en donde se especifica el nombre de equipo en donde reside esta base de datos "`GONZALO-C7F8854`", el nombre de la base de datos "`copueblo`", el nombre de inicio de sesión "`sa`", y la contraseña para esta sesión "`sa.1`".

De esta manera se establece la conexión con la base de datos, y se puede realizar cualquier acción sobre ella. También se observa cómo se invoca a la librería

system.dataen la conexión, ya que como se explica, esta es la librería que contiene todas las aplicaciones necesarias para utilizar una base de datos.

Dado que esta librería pertenece a la capa de la lógica del negocio, tal como se observa en el diagrama anterior. Aquí se encuentran todas las clases y funciones que hacen posible que el software funcione.

Estas clases se observan en la siguiente figura:

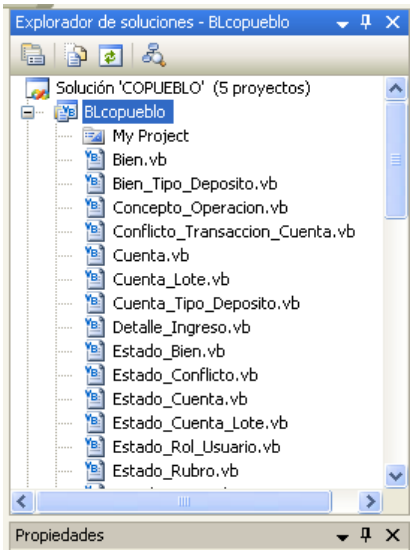


Figura 24: Clases del sistema

Fuente: Tesistas

Tal y como se observa dentro de la librería BLcopueblo, están todas las clases que componen el sistema informático.

4.3.2 FMUG

Dentro de la librería FMUG, se encuentra todo lo que corresponde a la capa de presentación del sistema, como se explicó en el capítulo II, toda la parte visual del sistema, está desarrollada con la tecnología AJAX.

Cuando el usuario ya se ha registrado en el sistema, pasará inmediatamente a la pantalla principal de administración. Aquí se encontrará con varios ítems como:

1. En la parte superior izquierda está el nombre de la compañía.
2. Debajo del nombre de la compañía en la parte izquierda se encuentra todas las acciones que el usuario del sistema puede realizar.
3. En la parte superior a centro se encuentra la fecha y hora en la que nos encontramos, así como, un ícono que permite cerrar la sesión y otro que permite salir del sistema.
4. Ya dentro de la pantalla principal se encuentra el nombre del usuario que se registró en el sistema.

La pantalla a la cual nos referimos es la siguiente:

Figura 25: Pantalla principal del sistema

Fuente: Tesistas

Para que el usuario pueda visualizar esta pantalla existe un código que hace posible que este tenga una interfaz gráfica amigable y se sienta cómodo a la hora de realizar su trabajo.

A continuación presentamos el código que hace posible que el usuario puede ver todas estas acciones posibles en su pantalla. Presentaremos un pequeño bloque de código que hace posible que se visualice la fecha y hora en la pantalla.

La declaración se lo hace en una clase del tipo JavaScript, como la mayor parte del código del resto del sistema.

Esta declaración se encuentra dentro de la clase de JavaScript llamada: _GScript.js como se puede observar en el gráfico siguiente:

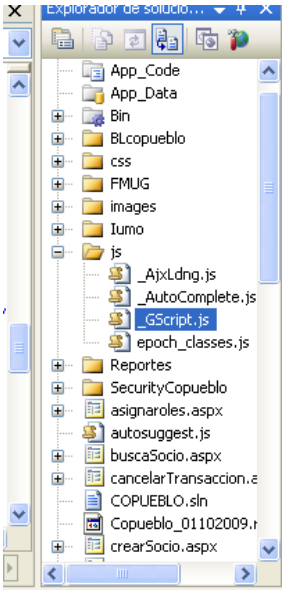


Figura 26: Clases de JavaScript

Fuente: Tesistas

Dentro de esta clase de tipo JavaScript se encuentran declaradas muchas funciones que luego serán usadas por clases declaradas en la parte principal del sistema.

La función que hace posible que se visualice la fecha y hora en el sistema es la siguiente:

```
function FechaTiempo()
    {
        var AR = new Array();
        var t = new Date();
var sec;
        var min;
        var hor;
        var dias = new Array("Lunes", "Martes", "Miercoles",
"Jueves", "Viernes", "Sabado", "Domingo");
        var meses = new Array("Enero", "Febrero", "Marzo", "Abril",
"Mayo", "Junio", "Julio", "Agosto", "Septiembre", "Octubre",
"Noviembre", "Diciembre");
        var NombreDia;
        var NombreMes;
        var cadenaFecha;
        NombreDia = dias[t.getDay()-1];
        NombreMes = meses[t.getMonth()];

if(t.getHours() < 10){hor="0"+t.getHours();}else{hor=t.getHours();}
```

```
        if(t.getMinutes() <
10){min="0"+t.getMinutes();}else{min=t.getMinutes();}
        if(t.getSeconds() <
10){sec="0"+t.getSeconds();}else{sec=t.getSeconds();}

cadenaFecha = NombreDia + ", " + t.getDate() + " de " + NombreMes +
" de " + t.getFullYear();
        cadenaFecha = cadenaFecha + " <b>" + hor + "</b>" + "h" +
"<b>" + min + "</b>" +  ":" + "<span style='font-size:smaller'>" +
sec + "</span>";
        return cadenaFecha;
    }
```

Esta función luego es llamada en el bloque de código HTML de la pantalla a la cual se desea que haga uso de esta función.

Para realizar una llamada a esta función desde el bloque de código HTML de cualquier pantalla que se desee se ejecuta el siguiente comando:

```
<script type="text/javascript" language="javascript"
src="js/_GScript.js"></script>
```

De esta manera se realiza el llamado a la clase _DScript.js y se puede usar todas las funciones declaradas en esta clase.

En la pantalla en mención se ha creado otra función llamada `Init()` que ejecuta la función `FechaTiempo()` declarada en la clase `_GScript.js`. esta funcion se puede observar a continuación:

```
function Init()
    {
        ajustaTamanio();
        LeeSesiones();
        $("tdHora").innerHTML = FechaTiempo();
setTimeout("Init();", 1000);
        window.onresize = maximizaVentana;
}
    function LeeSesiones()
    {
        var resp = principal.LeeSessionesActivas().value;
        if(!(resp==null))
        {
            $("tdSesiones").innerHTML = resp;
        }
    }
```

De esta manera se ejecuta la función y se puede visualizar la fecha y hora en la parte superior de la pantalla.

La mayor parte de las funciones que realizan el trabajo son declaradas en clases externas y llamadas en las clases principales de las pantallas que se está construyendo.

Así mismo para darle un aspecto mucho más profesional a las páginas que el usuario puede visualizar, el diseño en cuanto a colores y estética de las pantallas se realiza en clases externas en este caso se las realiza en las llamadas "hojas de estilo". En estas clases se realiza toda la parte estética que le da una apariencia de diseño sencillo pero a la vez complejo.

A continuación se muestra una imagen en la que se observa la construcción de estas hojas de estilo:

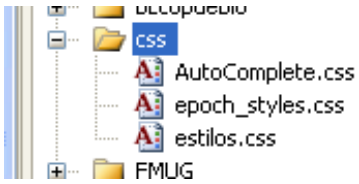


Figura 27: Construcción de las hojas de estilos

Fuente: Tesistas

Se pueden observar que se construyeron tres hojas de estilo en las cuales se puede realizar la apariencia que tendrán las páginas.

Para ilustrar esta sección, a continuación se presenta una pequeña parte de la hoja de estilo llama "estilos.css" la cual se puede observar en el grafico anterior.

```
.txtGrisTool
{
	color:#555555;
	font-family:Trebuchet MS;
	font-size:medium;
	text-decoration:none;
```

En el bloque de código se observa:

1. La llamada al componente de la pantalla que este caso se llama

   "`txtGrisTool`".

2. Luego se define el color que tendrá este componente, así como el tipo de letra y la decoración del texto.

Para que este pequeño bloque de código pueda ser ejecutado debe ser llamado de la misma forma que se llama a una clase en JavaScript, es decir desde el bloque HTML en la pantalla en construcción. Como se puede observar a continuación.

```
<link href="css/estilos.css" rel="stylesheet" type="text/css" />
```

De esta manera se ejecuta el código en la pantalla en construcción, y especialmente el componente tomará la forma y estructura que se le dio en la hoja de estilos.

En la imagen anterior se observa que existen tres clases de tipo hojas de estilos creadas. En estas hojas de estilos se realiza el diseño de todos los componentes que contienen todas las pantallas del sistema.

4.3.3 SECURITY COOPUEBLO

En la capa de SECURIY COOPUEBLO, se encuentra lo correspondiente a la seguridad del sistema informático. A continuación se presenta una breve descripción de lo que es MD5.

4.3.3.1 MD5

MD5 es un método de encriptación ampliamente utilizado, el cual consiste en obtener un resumen de un archivo o un documento, mediante el uso de ciertas funciones matemáticas.

El algoritmo genera una cadena de 32 caracteres, y generalmente recibe los nombres de. “resumen” o “huella”.

El algoritmo MD5 tiene varias características que lo hacen particularmente interesante. En primer lugar, su aplicación a una misma entrada obtiene siempre la misma salida. Por otro lado, pequeñas diferencias en los documentos de entrada generan normalmente grandes diferencias en la salida, lo que hace que cualquier modificación en el documento original se vea reflejada en el resumen. Además, la probabilidad de que dos documentos distintos generen la misma huella es muy baja.

Para terminar, y aunque su utilidad a efectos de verificación es escasa, es imposible averiguar el contenido del documento original a partir de su resumen.

MD5 es un algoritmo público y estandarizado por los organismos que acuerdan las tecnologías utilizadas en Internet, existiendo multitud de herramientas disponibles públicamente para calcular el resumen de cualquier documento.

Para verificar que un documento no ha sido alterado, utilizando MD5, hay que calcular su resumen MD5 aplicándole el algoritmo y comprobar que ese resumen coincide con un valor previamente generado (normalmente por el autor del documento) y hecho público por algún medio.

A continuación se presenta el código que hace posible que la información se guarde encriptada en la base de datos.

```
public string ObtenerMd5(string pass)
        {
            MD5 md5 = MD5CryptoServiceProvider.Create();
            byte[] dataMd5 =
md5.ComputeHash(Encoding.Default.GetBytes(pass));
            StringBuilder sb = new StringBuilder();
            for (int i = 0; i < dataMd5.Length; i++)
                sb.AppendFormat("{0:x2}", dataMd5[i]);
return sb.ToString();
        }


    }
}
```

Este bloque de código permite que la información se guarde de manera encriptada en la base de datos. Especialmente en lo referente a las contraseñas de los usuarios, ya que si alguna persona accede a la base de datos, lo que vera es una serie de caracteres inentendibles.

Como se observa primero se debe declarar las librerías necesarias para que posteriormente se pueda invocar al método de encriptación MD5. Y esto se observa en la librería que se encuentra en la línea número dos del bloque de código anterior.

Luego para usar este bloque de código, y que la información se guarde en la base de datos, hacemos el llamado de esta clase, desde la implementación de la clase (Usuario.vb). Este proceso se muestra a continuación:

```
Sub New(ByVal nick As String, ByVal clave As String)
        Dim where As String

        Dim claveMd5 As String
        Dim seguridadMd5 As New SecurityCopueblo.SecuritywithMd5()
        claveMd5 = seguridadMd5.ObtenerMd5(clave)

        where = String.Format("nick_usuario='{0}' and
clave_usuario='{1}'", nick, claveMd5)
        Iumo.ManagerListDB.GetSqlToObject(where, Me)

End Sub
```

Aquí se observa cómo se realiza la llamada a la clase en donde se implementa el algoritmo de encriptación MD5. Primero se invoca la librería, y luego a la clase que se encuentra dentro de esta librería, tal como se explica en el detalle de las librerías del sistema. Esta llamada lo podemos ver en la línea número cuatro de código del bloque anterior.

```
Dim seguridadMd5 As New SecurityCopueblo.SecuritywithMd5()
```

De esta manera se hace uso del algoritmo implementado, y la información de la clave del usuario se guardará en la base de datos encriptado.

ID_USUARIO	ID_ESTADO_U...	ID_PERFIL	NICK_USUARIO	CLAVE_USUARIO	NOMBRE_USUA...	DIRECC
1	1	1	adm	b09c600fddc573f117449b3723f23d64	Ec. Esther Estrada	Guayaqu
2	1	3	andres	231badb19b93e44f47da1bd64a8147f2	Andres Narvaez	Las casa

Figura 28: Tabla con datos encriptados

Fuente: Tesistas

En el grafico anterior se observa que la clave del usuario se guarda encriptado en la tabla USUARIO. El guardar la clave de esta forma en la base de datos, le confiere una seguridad adicional al sistema, ya que si por alguna razón esta tabla llega a ser abierta por una persona que desee ver las claves, se encontrara con una combinación de número y letras inentendibles, y por supuesto muy difíciles de descifrar, ya que se trata de un algoritmo de encriptación muy seguro.

4.4 CAPA DE DATOS

La capa de datos y persistencia hace referencia en este caso en particular al componente creado como: IUMO es aquí donde se crean los métodos para la conexión con la base de datos, así como también métodos de consultas, inserciones y eliminación de información en las diferentes tablas de la base de datos.

A continuación se presenta un breve ejemplo de un método contenido dentro de este componente:

```
Public Shared Function ExecuteSQL(ByVal sql As String, ByVal con As
IDbConnection) As Boolean
            Dim filas As Integer
            Dim valido As Boolean = False
            Dim cmd As IdbCommand = Nothing

            cmd = con.CreateCommand


            Try
                cmd.Connection = con
                cmd.CommandText = sql
                filas = cmd.ExecuteNonQuery()
                valido = True

            Catch ex As Exception

                valido = False

            End Try

            MyConnection.CloseConnetion(con)

            Return valido

        End Function
```

Como se observa el código anterior ejecuta una instrucción SQL mediante la instrucción `filas = cmd.ExecuteNonQuery()` se capturan los errores que puedan existir y finalmente se cierra la conexión, de esta manera es como en el componente o librería IUMO se mantiene la persistencia a la base de datos.

4.5 CODIGO DEL PROGRAMA.

Clase Transaccion_Cuenta.vb

```
ublic Class Transaccion_Cuenta

#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private
id_transaccion_cuenta As Integer
    Private id_cuenta As Integer
    Private id_usuario As Integer
    Private id_tipo_transaccion_cuenta As Integer
    Private id_tipo_deposito As Integer
    Private id_estado_transaccion_cuenta As Integer
    Private fecha_transaccion_cuenta As DateTime
    Private monto_transaccion_cuenta As Decimal
    Private saldo_anterior_transaccion_cuenta As Decimal
    Private saldo_actual_transaccion_cuenta As Decimal
    Private numero_documento_cuenta As String
    Private id_usuario_cancelacion As Integer
    Private fecha_documento_transaccion_cuenta As DateTime

#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_transaccion_cuenta
        End Get
        Set(ByVal value As Integer)
            Me.id_transaccion_cuenta = value
        End Set
    End Property

    Public Property Cuenta() As Integer
        Get
            Return id_cuenta
        End Get
        Set(ByVal value As Integer)
            Me.id_cuenta = value
        End Set
    End Property

    Public Property Usuario() As Integer
        Get
            Return id_usuario
        End Get
        Set(ByVal value As Integer)
            Me.id_usuario = value
        End Set
    End Property

    Public Property EstadoDeTransaccion() As Integer
        Get
            Return id_estado_transaccion_cuenta
        End Get
        Set(ByVal value As Integer)
            Me.id_estado_transaccion_cuenta = value
        End Set
    End Property
```

```
Public Property Fecha() As DateTime
    Get
        Return fecha_transaccion_cuenta
    End Get
    Set(ByVal value As DateTime)
        Me.fecha_transaccion_cuenta = value
    End Set
End Property

Public Property Monto() As Decimal
    Get
        Return monto_transaccion_cuenta
    End Get
    Set(ByVal value As Decimal)
        Me.monto_transaccion_cuenta = value
    End Set
End Property

Public Property SaldoAnterior() As Decimal
    Get
        Return saldo_anterior_transaccion_cuenta
    End Get
    Set(ByVal value As Decimal)
        Me.saldo_anterior_transaccion_cuenta = value
    End Set
End Property

Public Property SaldoActual() As Decimal
    Get
        Return saldo_actual_transaccion_cuenta
    End Get
    Set(ByVal value As Decimal)
        Me.saldo_actual_transaccion_cuenta = value
    End Set
End Property

Public Property Documento() As String
    Get
        Return numero_documento_cuenta
    End Get
    Set(ByVal value As String)
        Me.numero_documento_cuenta = value
    End Set
End Property

Public Property TipoDeTransaccion() As Integer
    Get
        Return id_tipo_transaccion_cuenta
    End Get
    Set(ByVal value As Integer)
        Me.id_tipo_transaccion_cuenta = value
    End Set
End Property

Public Property TipoDeDeposito() As Integer
    Get
        Return id_tipo_deposito
    End Get
    Set(ByVal value As Integer)
        Me.id_tipo_deposito = value
```

```
        End Set
    End Property
    Public Property UsuarioCancelacion() As Integer
        Get
            Return Me.id_usuario_cancelacion
        End Get
        Set(ByVal value As Integer)
            Me.id_usuario_cancelacion = value
        End Set
    End Property
    Public Property FechaDocumento() As DateTime
        Get
            Return Me.fecha_documento_transaccion_cuenta
        End Get
        Set(ByVal value As DateTime)
            Me.fecha_documento_transaccion_cuenta = value
        End Set
    End Property
#End Region

#Region "Constructores"
    Sub New()

    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub
    Sub New(ByVal documento As String, ByVal estadoDeTransaccion As
Integer)
        Dim where As String

        where = String.Format("numero_documento_cuenta='{0}' and
id_estado_transaccion_cuenta={1}", documento, estadoDeTransaccion)
        Iumo.ManagerListDB.GetSqlToObject(where, Me)
    End Sub
#End Region
#Region "Metodos"
    Function Guardar() As Boolean
        Dim retorna As Boolean = False
        Dim Itrans As IdbTransaction =
Iumo.IumoTransacction.GetIumoTransaction()

        Try
            If Me.ID = Nothing Then
                Iumo.ManagerDB.SaveWithIDUsingTransaction(Me,
Itrans, True)
            Else
                Iumo.ManagerDB.UpdateUsingTransaction(Me, Itrans)
            End If
            Itrans.Commit()
            retorna = True
        Catch ex As Exception
            Itrans.Rollback()
            retorna = False

        End Try
        Return retorna
    End Function
    Function CancelarTransaccion() As Boolean
        Dim retorna As Boolean = False
```

```
        Dim cuenta As Cuenta

        Me.EstadoDeTransaccion =
EstadoDeTransaccionDeCuenta.Cancelada
        If Me.Guardar = True Then
            cuenta = New Cuenta(Me.Cuenta)
            If cuenta.Debitar(Me.UsuarioCancelacion, Me.Monto,
TipoDeTransaccionCuenta.RetiroPorCancelacionDeTransaccion,
Me.TipoDeDeposito) = True Then
                Return True
            Else
                Return False
            End If

        End If
    End Function
    Function EliminarTransaccion()
        Dim retorna As Boolean = False

        Me.EstadoDeTransaccion =
EstadoDeTransaccionDeCuenta.Eliminada

        Return Me.Guardar
    End Function

#End Region

End Class
```

Clase Socio.vb

```
Public Class Socio
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private id_socio
As Integer
    Private id_estado_socio As Integer
    Private codigo_socio As String
    Private cedula_socio As String
    Private nombres_socio As String
    Private apellidos_socio As String
    Private fecha_nac_socio As DateTime
    Private direccion_socio As String
    Private telefono_socio As String
    Private celular_socio As String
    Private lugar_trabajo_socio As String
    Private direccion_trabajo_socio As String
    Private telefono_trabajo_socio As String
    Private email_socio As String
    Private sexo_socio As String
    Private fecha_ingreso_socio As DateTime

#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_socio
        End Get
        Set(ByVal value As Integer)
```

```
            Me.id_socio = value
        End Set
    End Property

    Public Property EstadoSocio() As Integer
        Get
            Return id_estado_socio
        End Get
        Set(ByVal value As Integer)
            Me.id_estado_socio = value
        End Set
    End Property

    Public Property Codigo() As String
        Get
            Return codigo_socio
        End Get
        Set(ByVal value As String)
            Me.codigo_socio = value
        End Set
    End Property

    Public Property Cedula() As String
        Get
            Return cedula_socio
        End Get
        Set(ByVal value As String)
            Me.cedula_socio = value
        End Set
    End Property

    Public Property Nombres() As String
        Get
            Return nombres_socio
        End Get
        Set(ByVal value As String)
            Me.nombres_socio = value
        End Set
    End Property

    Public Property Apellidos() As String
        Get
            Return apellidos_socio
        End Get
        Set(ByVal value As String)
            Me.apellidos_socio = value
        End Set
    End Property

    Public Property FechaNacimiento() As DateTime
        Get
            Return fecha_nac_socio
        End Get
        Set(ByVal value As DateTime)
            Me.fecha_nac_socio = value
        End Set
    End Property

    Public Property Direccion() As String
        Get
            Return direccion_socio
```

```
    End Get
    Set(ByVal value As String)
        Me.direccion_socio = value
    End Set
End Property

Public Property Telefono() As String
    Get
        Return telefono_socio
    End Get
    Set(ByVal value As String)
        Me.telefono_socio = value
    End Set
End Property

Public Property Celular() As String
    Get
        Return celular_socio
    End Get
    Set(ByVal value As String)
        Me.celular_socio = value
    End Set
End Property

Public Property LugarTrabajo() As String
    Get
        Return lugar_trabajo_socio
    End Get
    Set(ByVal value As String)
        Me.lugar_trabajo_socio = value
    End Set
End Property

Public Property DireccionTrabajo() As String
    Get
        Return direccion_trabajo_socio
    End Get
    Set(ByVal value As String)
        Me.direccion_trabajo_socio = value
    End Set
End Property

Public Property TelefonoTrabajo() As String
    Get
        Return telefono_trabajo_socio
    End Get
    Set(ByVal value As String)
        Me.telefono_trabajo_socio = value
    End Set
End Property

Public Property Email() As String
    Get
        Return email_socio
    End Get
    Set(ByVal value As String)
        Me.email_socio = value
    End Set
End Property

Public Property Sexo() As String
```

```
        Get
            Return sexo_socio
        End Get
        Set(ByVal value As String)
            Me.sexo_socio = value
        End Set
    End Property

    Public Property FechaIngreso() As DateTime
        Get
            Return fecha_ingreso_socio
        End Get
        Set(ByVal value As DateTime)
            Me.fecha_ingreso_socio = value
        End Set
    End Property

#End Region

#Region "Constructores"

    Sub New()
    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub
    Sub New(ByVal cedula As String)
        Dim where As String
        where = String.Format("cedula_socio='{0}'", cedula)
        Iumo.ManagerListDB.GetSqlToObject(where, Me)
    End Sub

#End Region
#Region "Metodos"

    Function Guardar() As Boolean
        Dim retorna As Boolean = False
        Dim Itrans As IdbTransaction =
Iumo.IumoTransacction.GetIumoTransaction()

        Try
            If Me.ID = Nothing Then
                Iumo.ManagerDB.SaveWithIDUsingTransaction(Me,
Itrans, True)

            Else
                Iumo.ManagerDB.UpdateUsingTransaction(Me, Itrans)

            End If

            Itrans.Commit()
            retorna = True
        Catch ex As Exception
            Itrans.Rollback()
            retorna = False

        End Try
        Return retorna
    End Function
    Public Sub CrearCodigo()
```

```
        Dim parametro As New Parametro("sufijoSocio")
        Me.Codigo = parametro.Valor + Me.ID.ToString("000000")
        Me.Guardar()
    End Sub
#End Region
#Region "Funciones"

    Public Shared Function TraeSocios(ByVal estadoDelSocio As
Integer) As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from SOCIO Where ")
        sql.AppendFormat("  id_estado_socio = {0} order by
apellidos_socio,nombres_socio", estadoDelSocio)

        Return Iumo.ManagerListDB.GetList(sql.ToString, New Socio)

    End Function
    Public Shared Function TraeSocios() As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from SOCIO order by
apellidos_socio,nombres_socio")

        Return Iumo.ManagerListDB.GetList(sql.ToString, New Socio)

    End Function
    Public Shared Function TraeSocios(ByVal nombreSocio As String,
ByVal apellidoSocio As String) As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from SOCIO Where ")
        sql.AppendFormat("  nombres_socio like '%{0}%' ",
nombreSocio)
        sql.AppendFormat(" Or apellidos_socio like '%{0}%' order by
apellidos_socio,nombres_socio", apellidoSocio)

        Return Iumo.ManagerListDB.GetList(sql.ToString, New Socio)

    End Function
#End Region

End Class
```

Clase Lotes.vb

```
Public Class Lotes
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private id_lote As
Integer
    Private id_bien As Integer
    Private numero_lote As String
    Private area_lote As String
    Private ubicacion_lote As String
    Private valor_lote As Decimal

#End Region

#Region "Propiedades"
```

```
    Public Property ID() As Integer
        Get
            Return id_lote
        End Get
        Set(ByVal value As Integer)
            Me.id_lote = value
        End Set
    End Property

    Public Property Bien() As Integer
        Get
            Return id_bien
        End Get
        Set(ByVal value As Integer)
            Me.id_bien = value
        End Set
    End Property

    Public Property Numero() As String
        Get
            Return numero_lote
        End Get
        Set(ByVal value As String)
            Me.numero_lote = value
        End Set
    End Property

    Public Property Area() As String
        Get
            Return area_lote
        End Get
        Set(ByVal value As String)
            Me.area_lote = value
        End Set
    End Property

    Public Property Ubicacion() As String
        Get
            Return ubicacion_lote
        End Get
        Set(ByVal value As String)
            Me.ubicacion_lote = value
        End Set
    End Property

    Public Property Valor() As Decimal
        Get
            Return valor_lote
        End Get
        Set(ByVal value As Decimal)
            Me.valor_lote = value
        End Set
    End Property

#End Region

#Region "Constructores"

    Sub New()
    End Sub
```

```
    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub

#End Region
#Region "Metodos"

    Function Guardar() As Boolean
        Dim retorna As Boolean = False
        Dim Itrans As IdbTransaction =
Iumo.IumoTransacction.GetIumoTransaction()

        Try
            If Me.ID = Nothing Then
                Iumo.ManagerDB.SaveWithIDUsingTransaction(Me,
Itrans, True)
            Else
                Iumo.ManagerDB.UpdateUsingTransaction(Me, Itrans)

            End If

            Itrans.Commit()
            retorna = True
        Catch ex As Exception
            Itrans.Rollback()
            retorna = False

        End Try
        Return retorna
    End Function

#End Region
#Region "Funciones"
    Public Shared Function TraeTipoDepositoPorLote(ByVal lote As
Integer) As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from LOTES_TIPO_DEPOSITO Where
")
        sql.AppendFormat("  id_lote = {0} ", lote)

        Return Iumo.ManagerListDB.GetList(sql.ToString, New
Lotes_Tipo_Deposito)

    End Function
#End Region

End Class
```

Cuenta_Lote.vb

```
Public Class Cuenta_Lote
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private
id_cuenta_lote As Integer
    Private id_estado_cuenta_lote As Integer
    Private id_lote As Integer
    Private id_cuenta As Integer
    Private observacion_cuenta_lote As String

#End Region
```

```
#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_cuenta_lote
        End Get
        Set(ByVal value As Integer)
            Me.id_cuenta_lote = value
        End Set
    End Property

    Public Property EstadoCuentaLote() As Integer
        Get
            Return id_estado_cuenta_lote
        End Get
        Set(ByVal value As Integer)
            Me.id_estado_cuenta_lote = value
        End Set
    End Property

    Public Property Lote() As Integer
        Get
            Return id_lote
        End Get
        Set(ByVal value As Integer)
            Me.id_lote = value
        End Set
    End Property

    Public Property Cuenta() As Integer
        Get
            Return id_cuenta
        End Get
        Set(ByVal value As Integer)
            Me.id_cuenta = value
        End Set
    End Property

    Public Property Observacion() As String
        Get
            Return observacion_cuenta_lote
        End Get
        Set(ByVal value As String)
            Me.observacion_cuenta_lote = value
        End Set
    End Property

#End Region

#Region "Constructores"
    Sub New()

    End Sub

    Sub New(ByVal cuenta As Integer)
        Dim where As String
        where = String.Format("id_cuenta={0}", cuenta)
        Iumo.ManagerListDB.GetSqlToObject(where, Me)
    End Sub
#End Region
#Region "Metodos"
    Function Guardar() As Boolean
```

```
        Dim retorna As Boolean = False
        Dim Itrans As IdbTransaction =
Iumo.IumoTransacction.GetIumoTransaction()

        Try
            If Me.ID = Nothing Then
                Iumo.ManagerDB.SaveWithIDUsingTransaction(Me,
Itrans, True)
            Else
                Iumo.ManagerDB.UpdateUsingTransaction(Me, Itrans)

            End If

            Itrans.Commit()
            retorna = True
        Catch ex As Exception
            Itrans.Rollback()
            retorna = False

        End Try
        Return retorna

    End Function

#End Region


End Class
```

Clase Estado_Bien.vb

```
Public Class Estado_Bien
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private
id_estado_bien As Integer
    Private nombre_estado_bien As String

#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_estado_bien
        End Get
        Set(ByVal value As Integer)
            Me.id_estado_bien = value
        End Set
    End Property

    Public Property Nombre() As String
        Get
            Return nombre_estado_bien
        End Get
        Set(ByVal value As String)
            Me.nombre_estado_bien = value
        End Set
    End Property

#End Region
```

```
#Region "Constructores"

    Sub New()
    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub

#End Region
#Region "Metodos"

    Sub Guardar()
        If Me.ID = Nothing Then
            Iumo.ManagerDB.SaveWithID(Me, True)
        Else
            Iumo.ManagerDB.Update(Me)
        End If
    End Sub

#End Region
#Region "Funciones"
    Public Shared Function TraeEstadosDelBien() As ArrayList
        Dim sql As String


sql = "Select * from Estado_Bien "

        Return Iumo.ManagerListDB.GetList(sql, New Estado_Bien)
    End Function
#End Region

End Class
```

Clase Estado_Cuenta.vb

```
Public Class Estado_Cuenta
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private
id_estado_cuenta As Integer
    Private nombre_estado_cuenta As String

#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_estado_cuenta
        End Get
        Set(ByVal value As Integer)
            Me.id_estado_cuenta = value
        End Set
    End Property

    Public Property Nombre() As String
        Get
            Return nombre_estado_cuenta
        End Get
```

```
        Set(ByVal value As String)
            Me.nombre_estado_cuenta = value
        End Set
    End Property

#End Region

#Region "Constructores"

    Sub New()
    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub

#End Region
#Region "Metodos"

    Sub Guardar()
        If Me.ID = Nothing Then
            Iumo.ManagerDB.SaveWithID(Me, True)
        Else
            Iumo.ManagerDB.Update(Me)
        End If
    End Sub

#End Region
#Region "Funciones"

#End Region
    Public Shared Function TraeEstadosDeCuenta() As ArrayList
        Dim sql As String

        sql = "Select * from Estado_Cuenta"
        Return Iumo.ManagerListDB.GetList(sql, New Estado_Cuenta)

    End Function
End Class
```

Clase Estado_Socio.vb

```
Public Class Estado_Socio
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private
id_estado_socio As Integer
    Private nombre_estado_socio As String

#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_estado_socio
        End Get
        Set(ByVal value As Integer)
```

```
            Me.id_estado_socio = value
        End Set
    End Property

    Public Property Nombre() As String
        Get
            Return nombre_estado_socio
        End Get
        Set(ByVal value As String)
            Me.nombre_estado_socio = value
        End Set
    End Property

#End Region

#Region "Constructores"

    Sub New()
    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub

#End Region
#Region "Metodos"

    Sub Guardar()
        If Me.ID = Nothing Then
            Iumo.ManagerDB.SaveWithID(Me, True)
        Else
            Iumo.ManagerDB.Update(Me)
        End If
    End Sub

#End Region
#Region "Funciones"
    Public Shared Function TraeEstadosDelSocio() As ArrayList
        Dim sql As String

        sql = "Select * From Estado_Socio"

        Return Iumo.ManagerListDB.GetList(sql, New Estado_Socio)
    End Function
#End Region

End Class
```

Clase Permiso.vb

```
Public Class Permiso
#Region "Variables"
    <Iumo.DeclarationVar(Iumo.TypeAttribute.key)> Private id_permiso
As Integer
    Private nombre_permiso As String
    Private pagina_permiso As String
    Private id_permiso_aux As Integer
    Private id_rol As Integer
    Private etiqueta_pagina_permiso As String
```

```
    Private orden_permiso As String
#End Region

#Region "Propiedades"
    Public Property ID() As Integer
        Get
            Return id_permiso
        End Get
        Set(ByVal value As Integer)
            Me.id_permiso = value
        End Set
    End Property
    Public Property Etiqueta() As String
        Get
            Return Me.etiqueta_pagina_permiso

        End Get
        Set(ByVal value As String)
            Me.etiqueta_pagina_permiso = value

        End Set
    End Property

    Public Property Nombre() As String
        Get
            Return nombre_permiso
        End Get
        Set(ByVal value As String)
            Me.nombre_permiso = value
        End Set
    End Property

    Public Property Pagina() As String
        Get
            Return pagina_permiso
        End Get
        Set(ByVal value As String)
            Me.pagina_permiso = value
        End Set
    End Property

    Public Property PermisoAux() As Integer
        Get
            Return id_permiso_aux
        End Get
        Set(ByVal value As Integer)
            Me.id_permiso_aux = value
        End Set
    End Property

    Public Property Rol() As Integer
        Get
            Return id_rol
        End Get
        Set(ByVal value As Integer)
            Me.id_rol = value
        End Set
    End Property

    Public Property Orden() As String
        Get
```

```
            Return Me.orden_permiso
        End Get
        Set(ByVal value As String)
            Me.orden_permiso = value
        End Set
    End Property

#End Region

#Region "Constructores"

    Sub New()
    End Sub

    Sub New(ByVal key As Integer)
        Iumo.ManagerListDB.GetSqlToObject(key, Me)
    End Sub

#End Region
#Region "Metodos"

    Sub Guardar()
        If Me.ID = Nothing Then
            Iumo.ManagerDB.SaveWithID(Me, True)
        Else
            Iumo.ManagerDB.Update(Me)
        End If
    End Sub

#End Region
#Region "Funciones"

    Public Shared Function TraePermisosPorRol(ByVal rol As Integer)
As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from PERMISO Where ")
        sql.AppendFormat("  id_rol = {0} order by orden_permiso",
rol)

        Return Iumo.ManagerListDB.GetList(sql.ToString, New Permiso)

    End Function
    Public Shared Function TraePermisosPorRol(ByVal rol As Integer,
ByVal id_aux_permiso As Integer) As ArrayList
        Dim sql As New StringBuilder

        sql.AppendFormat(" Select * from PERMISO Where ")
        sql.AppendFormat("  id_rol = {0} ", rol)
        sql.AppendFormat(" And id_permiso_aux = {0} order by
orden_permiso", id_aux_permiso)

        Return Iumo.ManagerListDB.GetList(sql.ToString, New Permiso)

    End Function
#End Region

End Class
```

4.6 IMPLEMENTACIÓN

4.6.1 Requisitos de hardware.

El trabajo que se pretende desarrollar para la "Compañía de Promoción y Desarrollo Humano y Social Coopueblo". Será realizado utilizando las siguientes herramientas de desarrollo:

Creación de la base de datos.- la base de datos que se pretende crear para la compañía será desarrollada utilizando la herramienta de SQL Server 2005.

Desarrollo de la aplicación.- la aplicación o software será desarrollado utilizando la herramienta visual Estudio 2005.

Las herramientas mencionadas han sido propuestas con anterioridad y han sido aprobadas por parte del consejo de carrera de la universidad. Por lo que el desarrollo de la aplicación web será desarrollado en su totalidad utilizando estas herramientas.

Se menciona estas herramientas porque, en base a ellas se realizará una documentación de las características y capacidades que debe tener el hardware, para que pueda soportar las aplicaciones mencionadas.

El equipo de hardware y software es necesario para que la aplicación web pueda funcionar es el siguiente:

Visual Estudio 2005[8]

Requisito	Professional
Procesador	Procesador de 600 MHz Se recomienda: procesador de 1 gigahercio (GHz)
RAM	192 MB Se recomiendan: 256 MB
Espacio disponible en el disco duro	Sin MSDN: • Se requiere 1 GB de espacio disponible en la unidad del sistema • Se requiere 2 GB de espacio disponible en la unidad de instalación Con MSDN: • Se requiere 1 GB de espacio disponible en la unidad del sistema • Se requieren 3,8 GB de espacio disponible en la unidad de instalación con una instalación completa de MSDN
Sistema operativo	Windows 2000 Service Pack 4, Windows XP Service Pack 2 o Windows Server 2003 Service Pack 1 Para un equipo de 64 bits, los requisitos son: • Ediciones de x64 de Windows Server 2003 Service Pack 1 • Edición de x64 para Windows XP Professional

Tabla 10: Requisitos para la instalación de Visual Studio 2005

[8]http://msdn.microsoft.com/es-es/library/4c26cc39(v=vs.80).aspx

SQL SERVER 2005[9]

Esta tabla muestra los requisitos de hardware para instalar y ejecutar SQL Server 2005 en la plataforma de 32 bits.

SQL Server 2005 (32 bits)	Tipo de procesador	Velocidad de procesador	Memoria (RAM)
SQL Server 2005 Enterprise Edition SQL Server 2005 Developer Edition SQL Server 2005 Standard Edition	Se requiere un procesador compatible con Pentium III o superior	Mínimo: 600 MHz Recomendado: 1 GHz o superior	Mínimo: 512 MB Recomendado: 1 GB o superior Máximo: sistema operativo máximo
SQL Server 2005 WorkgroupEdition	Se requiere un procesador compatible con Pentium III o superior	Mínimo: 600 MHz Recomendado: 1 GHz o superior	Mínimo: 512 MB Recomendado: 1 GB o superior Máximo: 3 GB
SQL Server 2005 Express Edition	Se requiere un procesador compatible con Pentium III o superior	Mínimo: 600 MHz Recomendado: 1 GHz o superior	Mínimo: 192 MB Recomendado: 512 MB o superior Máximo: 1 GB

Tabla 11: Requisitos para la instalación de SQL Server 2005

1.6.1 Requisitos de espacio en disco duro

Durante la instalación de SQL Server 2005, Windows Installer crea archivos temporales en la unidad del sistema. Antes de ejecutar el programa de instalación

[9]http://download.microsoft.com/download/8/4/0/840afd36-a779-487/RequirementsSQL2005.htm

para instalar o actualizar SQL Server 2005, se debe verificar que se dispone de 1,6 GB de espacio en disco en la unidad del sistema para estos archivos. Este requisito es aplicable incluso si se instala componentes de SQL Server en una unidad distinta de la predeterminada.

Los requisitos de espacio en disco duro reales para los componentes instalados dependen de la configuración del sistema y de las aplicaciones y características que se haya decidido instalar. En la siguiente tabla se muestran los requisitos de espacio en disco de los componentes de SQL Server 2005.

Característica	**Requisito de espacio en disco**
Motor de base de datos y archivos de datos, Réplica y Búsqueda de texto	150 MB
AnalysisServices y archivos de datos	35 MB
ReportingServices y Administrador de informes	40 MB
Componentes del motor de NotificationServices, componentes de cliente y componentes de reglas	5 MB
IntegrationServices	9 MB
Componentes de cliente	12 MB
Herramientas de administración	70 MB
Herramientas de desarrollo	20 MB
Libros en pantalla de SQL Server y Libros en pantalla de SQL Server Mobile	15 MB

Tabla 12: Requisitos de espacio en disco

Fuente: http://download.microsoft.com/download/RequirementsSQL2005.htm

CAPÍTULO V

PRUEBAS

5.1 PRUEBAS DE CAJA NEGRA

Antes de empezar a describir lo que se refiere a pruebas de caja negra, veremos cómo se debe realizar un buen código.

Un Buen código es aquel que funciona sin bugs, (un **bug** es un error o un defecto en el software o hardware que hace que un programa funcione incorrectamente).

Además el código debe ser legible, se debe ajustar a los estándares dela organización para que todos los desarrolladores del sistema manejen y entiendan las mismas herramientas y mecanismos en la codificación.

También se consideran ciertas reglas que se detallan a continuación para evitar problemas en la mayoría de los lenguajes de programación más usados.

- Las descripciones deben ser cortas y claras para no confundir la lectura del código.
- Organización de los métodos, una buena disposición del código hará que futuros cambios sean posibles.
- Uso generoso de espacios en blanco.
- Es fundamental que el grupo de desarrolladores respete el mismo estilo de codificación.
- Toda aplicación debe ser documentada no importa lo pequeña que sea dicha aplicación.

Para las pruebas de caja negra se usará la herramienta RECORD-PLAYBACK, al ejecutar dicha aplicación antes de hacer correr el programa a probar, se activa la opción récord, el tester navega por las diferentes aplicaciones del software, menús, cuadros de dialogo, plantillas, tablas, botones, y los resultados son grabados en forma de texto para un reporte, y en el lenguaje de la herramienta en un archivo de grabado, cuando nuevas aplicaciones son agregadas al software o son modificadas las aplicaciones actuales, la opción playback de la herramienta navega automáticamente

por el programa y luego emite un reporte de resultados y operaciones aun no testeadas.

La prueba de caja negra implica una variada selección de los datos de prueba así, como una buena interpretación de los resultados para determinar el nivel de optimización de la funcionalidad del sistema.

El aspecto humano es esencial en la prueba de caja negra aplicando factibles sucesos de la vida real a la prueba, errores de tipeo, trabajar en aplicaciones equivocadas creyendo trabajar en la aplicación deseada, etc.

El principal objetivo es determinar la funcionalidad del software. La prueba de caja negra tiene otras metas, determinar la eficiencia del programa desde el desempeño en el equipo, el tiempo de retardo de las salidas hasta el nivel de recuperación del sistema luego de fallas o caídas sean estas producidas por manejo incorrecto de datos, equipo o producidas externamente como cortes de energía.

La prueba con intervención del usuario es un pequeño periodo de tiempo en el cual el Usuario, bajo el asesoramiento de testers, se desenvuelve en el software y examina la operatividad de los datos que el maneja. El usuario dará la opinión final en la cuestión de los datos de prueba.

Cabe recalcar que las pruebas se realizaron con la Ec. Esther Estrada que es la persona que maneja el sistema en su totalidad, es decir tiene acceso a todos los roles y perfiles del software.

A continuación las pruebas realizadas:

5.1.1 PRUEBA UNO

Ingreso al sistema

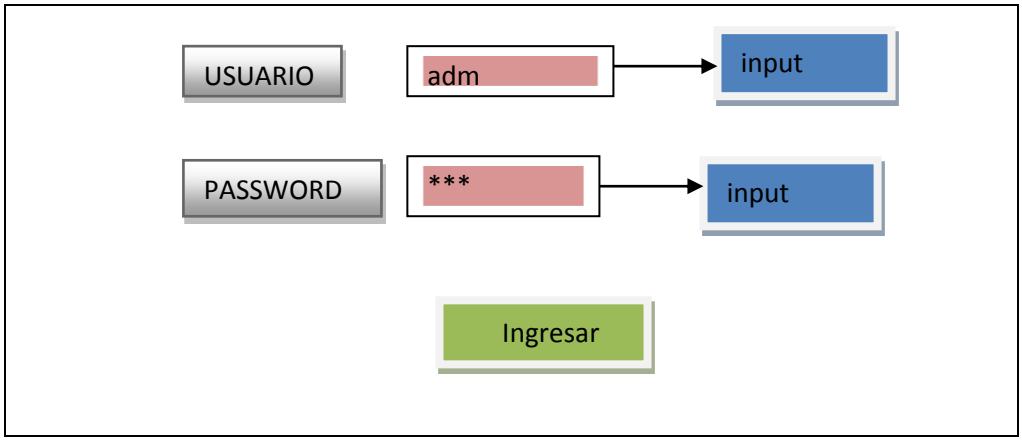


Figura 29: Ingreso al sistema

Fuente: Tesistas

Esta prueba se realizó el día martes en horas de la mañana. Como se visualiza para ingresar al sistema se debe colocar el usuario y contraseña, se presiona en botón ingresar y si estos datos son correctos inmediatamente pasa a la venta principal del sistema, caso contrario se despliega un mensaje de error.

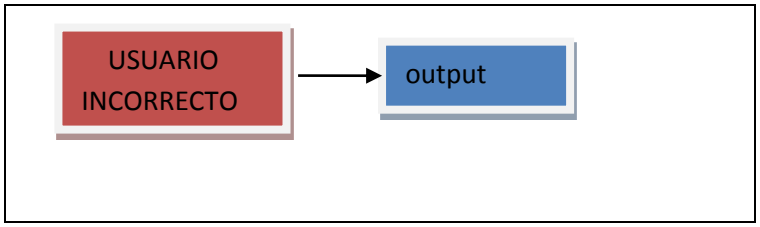


Figura 30: Mensaje de ingreso al sistema

Fuente: Tesistas

El tiempo de devolución de los resultados está acorde a los requerimientos de usuario.

5.1.2 PRUEBA DOS

Ingreso de datos de un nuevo socio

De igual manera que la prueba anterior se realizó el día martes en horas de la mañana.

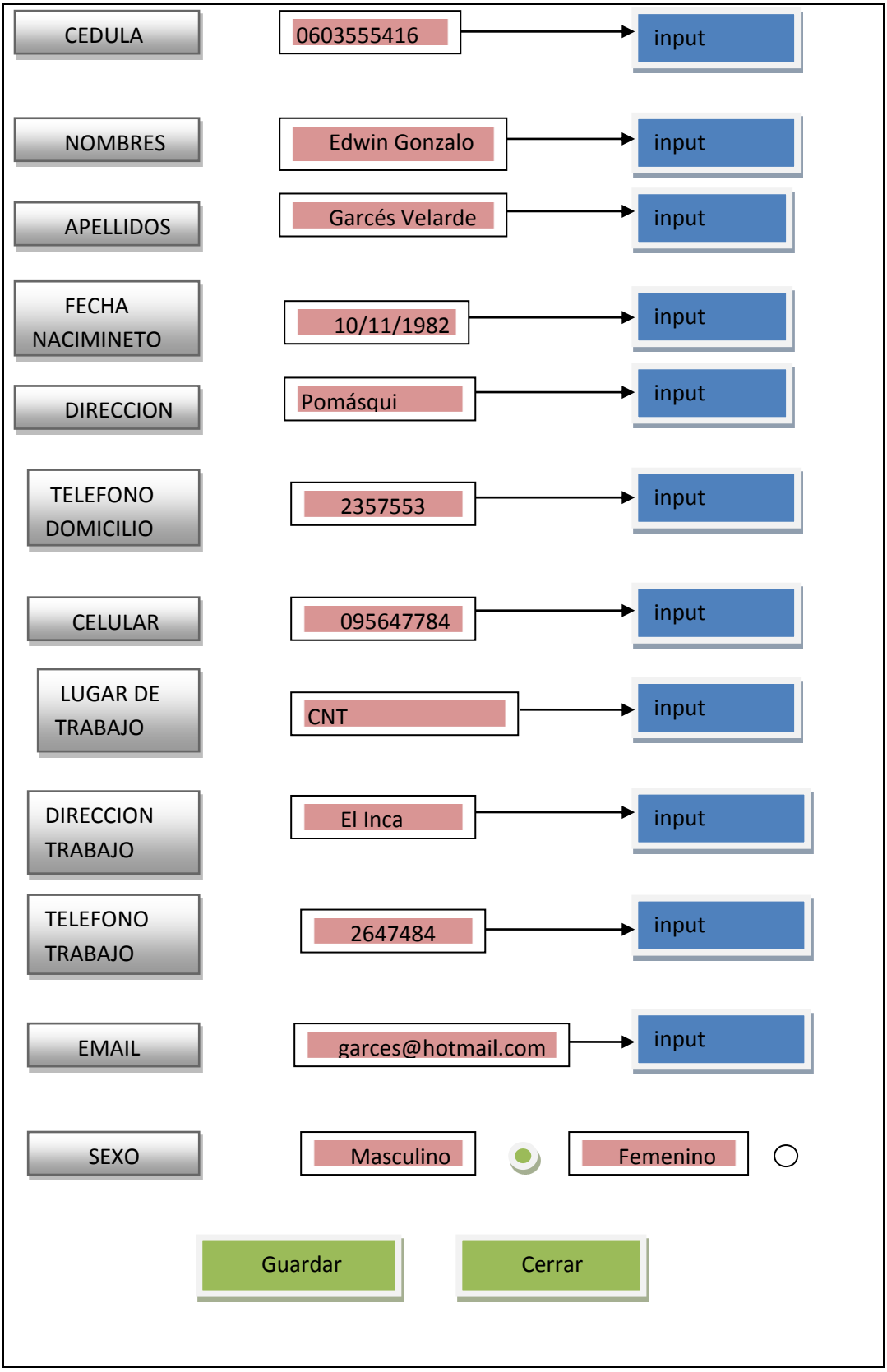


Figura 31: Ingreso de nuevo socio

Fuente: Tesistas

Como se muestra en el esquema anterior se debe ingresar todos los datos en los campos que se requieren, todos los campos son obligatorios de llenar, si alguno de ellos está incompleto o está mal ingresado el momento de proceder a guardar los datos se despliega un mensaje de error según sea el caso:

Si está incompleto un campo se despliega el mensaje:

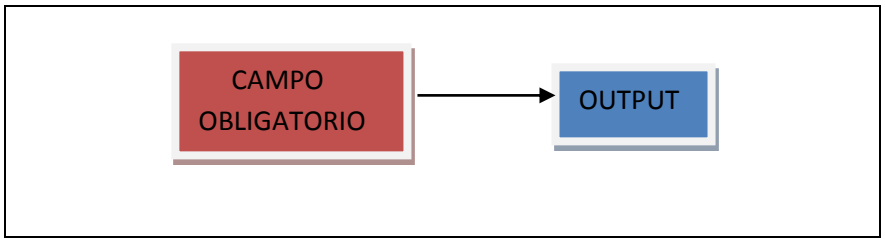


Figura 32: Mensaje de campo obligatorio

Fuente: Tesistas

Si no cumple con el formato establecido del campo se despliega el mensaje:

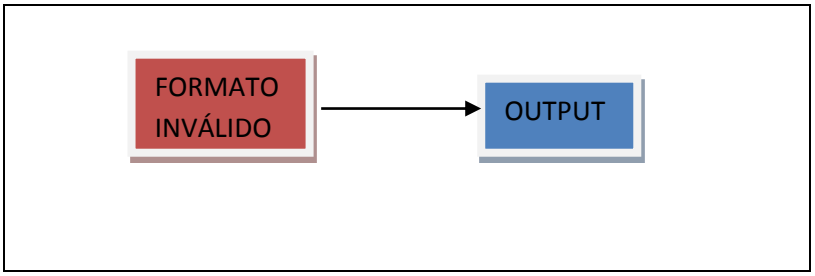


Figura 33: Mensaje de formato inválido

Fuente: Tesistas

Si todos los datos están completos e ingresados con el formato que requiere el campo, se despliega el mensaje

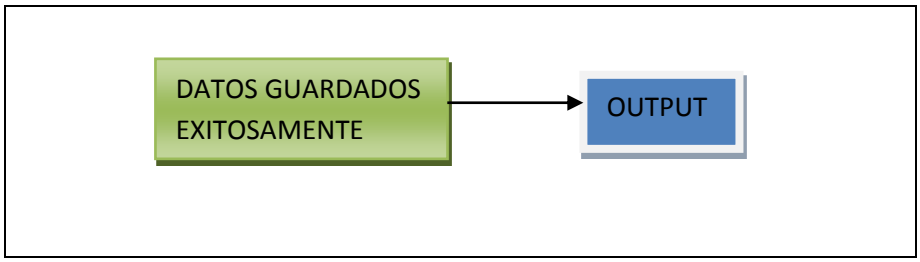


Figura 34: Mensaje de datos guardados

Fuente: Tesistas

Los tiempos de respuesta están dentro de lo establecido y al requerimiento del usuario.

5.1.3 PRUEBA TRES

Creación de una cuenta nueva para el socio

De la misma manera que las dos pruebas anteriores se realizó el día martes, pero en horas de la tarde.

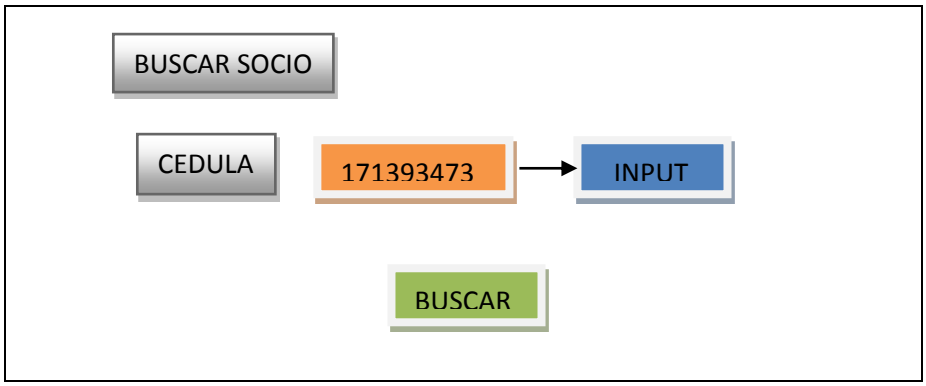


Figura 35: Creación de nueva cuenta

Fuente: Tesistas

El momento que se ingresa la cédula de un socio que ya está ingresado en el sistema se va a desplegar los datos personales del mismo, como se muestra a continuación:

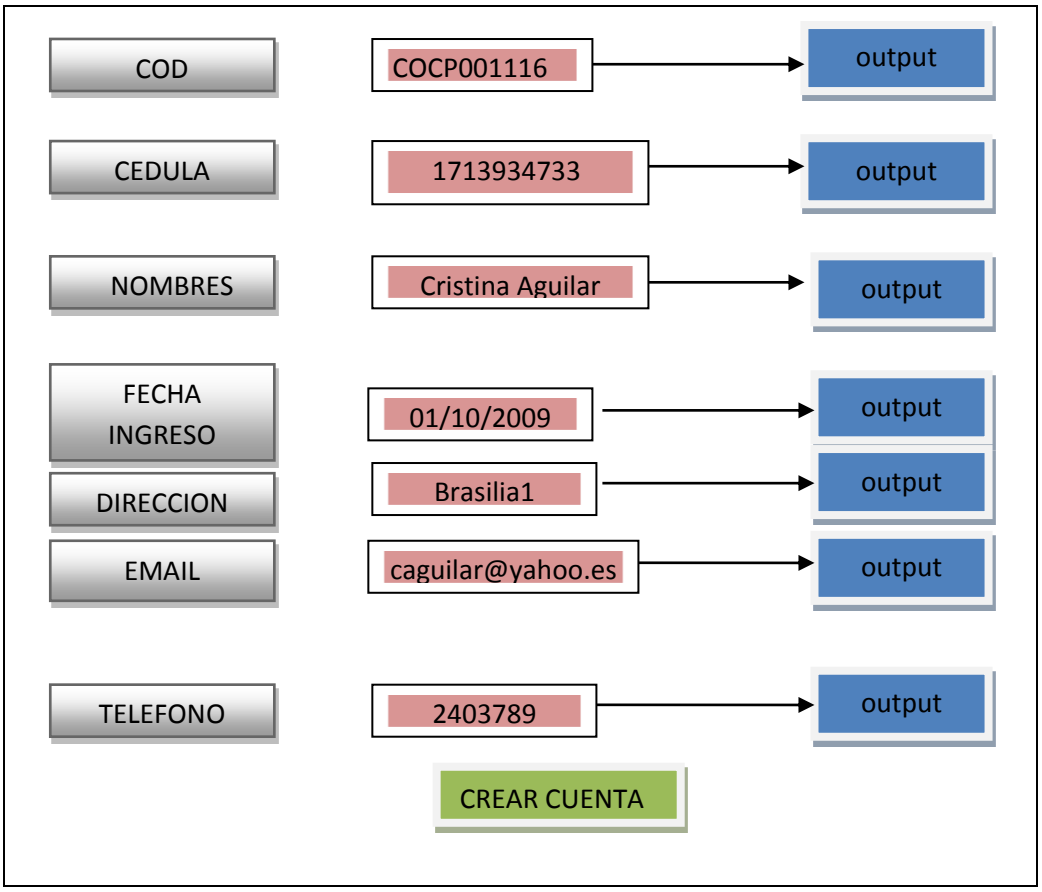


Figura 36: Ingreso de datos de la cuenta

Fuente: Tesistas

Si la cédula que se ingresa no está registrada en la base de datos del sistema aparece el mensaje:

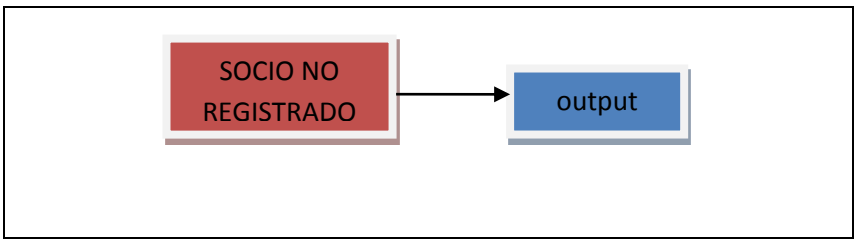


Figura 37: Mensaje de error de creación de cuenta

Fuente: Tesistas

El momento que se despliega los datos del cliente se debe oprimir el botón Crear cuenta y automáticamente se visualiza la cuenta asignada al socio:

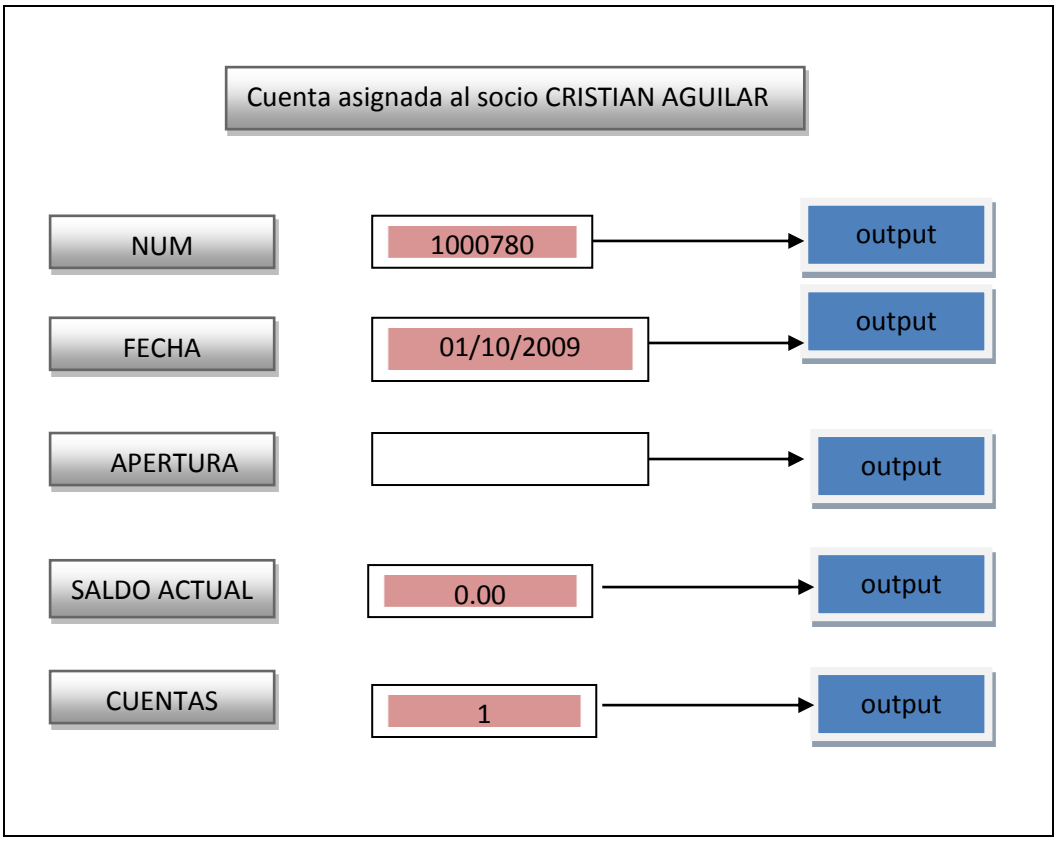


Figura 38: Detalles de la cuenta

Fuente: Tesistas

Se realizaron estas tres pruebas, tanto la del ingreso al sistema, el ingreso de datos del socio y la creación de la cuenta, ya que son las más relevantes y en la cuales se pueden medir los tiempos de respuesta adecuadamente y observar el desempeño bueno o malo del sistema.

Las pruebas fueron exitosas, ya que en ninguno de los tres escenarios se presentaron problemas de despliegue de información, problemas con los cuadros de diálogos (OUTPUT), y sobre todo los tiempos de respuesta de cada uno de estos fueron acorde al desarrollo de cada interacción desarrollada tanto por la Ec. Esther Estrada y los realizadores de las pruebas.

CAPITULO VI

CONCLUSIONES Y RECOMENDACIONES

6.1 CONCLUSIONES:

En base a lo realizado y a los resultados obtenidos podemos concluir lo siguiente:

- Se ha logrado realizar la implementación de un registro de inscripciones de los socios, para la creación de las cuentas con sus respectivos depósitos, mediante el uso de nuevas tecnologías, que nos ayudaron a construir de forma clara y precisa software de calidad, que garantiza el correcto funcionamiento de cada uno de los módulos presentados en el sistema.

- Se desarrolló un módulo de transferencias internas, el cual se lo utiliza para poder trasladar los depósitos (saldos), realizados por un socio a otro previo, al mutuo acuerdo. Este proceso se lo hizo mediante la obtención de datos de los socios de la compañía, contenidos en la Base de datos.

- El uso de tecnologías adicionales a los tradicionales lenguajes de programación, han sido de mucha ayuda, ya que pueden simplificar líneas de código de manera rápida, y además que son de fácil entendimiento e implementación en varias herramientas de desarrollo.

- Usar metodologías ágiles para el desarrollo de software, pueden resultar de mucha ayuda en proyectos cuyo plazo de entrega es corto, además que resultan fáciles de desarrollar e implementar si se cumplen con las reglas del modelo y las necesidades del usuario.

- Para el desarrollo de las bases de datos se ha tomado en cuenta las necesidades de almacenamiento de la información de la compañía, para lo cual se escogió la herramienta de gestión de bases de datos que posea una interfaz amigable y sea de consistencia robusta.

- El desarrollo de un sistema utilizando metodologías agiles, implica la presencia del cliente en todo momento del proceso de desarrollo, ya sea para realizar pruebas funcionales o para aportar con nuevos requerimientos, por lo que antes de elegir una de estas metodologías, se debe tener en consideración la disponibilidad del cliente.

- El uso de la metodología XP ha sido de gran ayuda para el desarrollo de este sistema, debido a que es una metodología no burocrática y bastante práctica, y además se ajusta perfectamente a procesos de desarrollo rápidos.

- El almacenar la información de las claves de los usuarios encriptado en la base de datos, le confiere una seguridad extra al sistema debido a que si alguien ingresa a la base para ver esta información, se encontrará con una serie de números y letras inentendibles.

6.2 RECOMENDACIONES:

Con la finalidad de obtener mejores resultados con respecto al proyecto se recomienda:

- Para garantizar un manejo adecuado de la aplicación informática, se recomienda la capacitación del personal que va a estar encargado del manejo de la misma.

- De acuerdo al incremento de los socios, y si la situación lo amerita, se recomienda la instalación de un servidor centralizado, en el cual se aloje el sistema informático y sirva como centro de almacenamiento de la información, ya que por su mayor robustez soporta alto tráfico de datos.

- De igual manera se recomienda que la base de datos y el sistema informático, no estén alojados en el mismo equipo, ya que de producirse un daño en este, se perdería tanto la información como el sistema informático.

- Debido al incremento constante de socios se recomienda que se obtengan respaldos periódicos del sistema, sobre todo de la base de datos, ya que es en esta donde se almacena la información de la compañía.

- El sistema instalado cumple con todas las expectativas actuales de los requerimientos de la compañía, sin embargo se debe tener en cuenta que: existen puntos que aún se hacen de forma no automatizada, por lo que se recomienda que un futuro cercano, se implementen todos los procesos necesarios para que se disponga de un sistema que cumpla 100% con las necesidades cambiantes de los usuarios y la compañía.

- Es muy importante que, cuando se culmine la evaluación de este sistema, se tome en consideración la implementación de una metodología del uso del sistema, como herramienta de apoyo para la implementación de este servicio, evaluando primeramente la infraestructura de la compañía.

GLOSARIO

DEFINICIONES, ACRÓNIMOS Y ABREVIATURAS

En esta sección se muestran todos los términos, acrónimos y abreviaturas que se han utilizado en el presente documento.

ADO.NET.- Conjunto de componentes usados para acceder a datos y a conjuntos de datos, es una parte de la biblioteca de clases base que están incluidas en el Microsoft .net framework.

AJAX.- Técnica de desarrollo web para desarrollar aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir en el navegador de los usuarios.

ASPX.- Extensión con la que se guardan los formularios web construidos en asp.net estos archivos típicamente contienen etiquetas HTML o XHTML estático, y también etiquetas definiendo Controles Web que se procesan del lado del servidor.

ASP.NET.- Es un framework para aplicaciones web desarrollado y comercializado por Microsoft. Es usado para construir sitios web dinámicos.

AWE.- Extensión de dirección física, permite a un sistema de 32 bits utilizar hasta 64 Gigabytes de memoria física.

COM.- Tecnología que permite crear componentes de software reutilizables.

COOPUEBLO.- Nombre de la compañía para la cual se ha desarrollado el presente sistema informático.

CSS.- Tecnología utilizada para la presentación de las diferentes pantallas del sistema. Ayuda a separar la presentación del documento de su estructura.

DOM.- Modelo de objetos del documento, proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

HTML.- Lenguaje de marcado de Hipertexto, lenguaje utilizado para la elaboración de Páginas web.

HttpRequest.- Permite a ASP.NET leer los valores HTTP enviados por un cliente durante una solicitud Web.

IIS.- Es un servidor web y un conjunto de servicios para el sistema operativo Microsoft Windows.

MSSQL.- Sistema para la gestión de bases de datos utilizado para la creación de bases de datos relacionales.

OLE DB.- Tecnología desarrollada por Microsoft usada para tener acceso a diferentes fuentes de información o bases de datos.

PHP.- Es un lenguaje de programación de uso general de script del lado del servidor.

RDBMS.- Sistema de gestión de base de datos relacional. Hace referencia a bases de datos relacionales como SQL Server.

SQL.- Lenguaje de consulta estructurado, es un lenguaje declarativo de acceso a bases de datos relacionales que permite realizar diferentes tipos de operaciones en ellas.

Transact/SQL (T-SQL).- Es una extensión al SQL de Microsoft y Sybase. SQL, que frecuentemente se dice ser un Lenguaje de Búsquedas Estructurado.

XHTML.- Es básicamente HTML expresado como XML válido. Es más estricto a nivel técnico, pero esto permite que posteriormente sea más fácil al hacer cambios o buscar errores entre otros.

XML.- Lenguaje de marcas extensible, permite definir la gramática de lenguajes específicos para estructurar documentos grandes. A diferencia de otros lenguajes, da soporte a bases de datos.

XP.-, Metodología para desarrollo de software.

XSLT.- Presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML.

.NET FRAMEWORK.- Componente de software incluido en los sistemas operativos Microsoft. Gestiona la ejecución de programas escritos específicamente para este framework.

ACRÓNIMOS Y ABREVIATURAS

AJAX: Asynchronous JavaScript And XML

AWE: Address widowing Extention.

COM: Component Object Model

CSS: Cascade style sheet

DOM: Document Object Model

HTML: HyperText Markup Language

IIS: Internet Information Services

MSSQL: Microsoft SQL

PHP: Hypertext Preprocessor

RDBMS: Relational Database Management System

SQL: Structured Language Query

XHTML: eXtensible HyperText Markup Language

XML: eXtensible Markup Language

XP: Extreme Programming

XSLT: Extensible Stylesheet Language Transformations

BIBLIOGRAFÍA.

- http://es.kioskea.net/contents/genie-logiciel/methodes-agiles.php3,Autor: "Desconocido", Titulo: "Métodos rápidos"
- http://www.buenastareas.com/ensayos/Modelo-Xp/1623491.html, Autor: "Indira Larraz", Titulo: "Modelo XP"
- http://elezeta.net/2004/08/27/extreme-programming-xp/, Autor: "Lucas Zallio", Titulo: "Extreme Programming"
- http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html, Autor: "Maria A. Mendoza sanchez", Titulo: "Metodologías de desarrollo de software"
- http://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema, Autor: "Desconocido",Titulo: "Extreme programming"
- http://programacionextrema.tripod.com/fases.htm, Autor: "Desconocido", Titulo: "Fases de la programación extrema"
- http://www.monografias.com/trabajos51/programacion-extrema/programacion-extrema2.shtml, Autor: "Adrian Anaya Villegas", titulo: "Extreme Programming"
- http://translate.google.com.ec/translate?hl=es&langpair=en%7Ces&u=http://blog.james-carr.org/2006/06/09/we-dont-use-uml-in-xp/, Autor "James Carr", Titulo: "Nosotros no usamos UML en XP"
- http://translate.google.com.ec/translate?hl=es&langpair=en%7Ces&u=http://www.trireme.com/whitepapers/process/xp-uml/paper.htm, Autor: "Alan Cameron Wills", Titulo: "UML ser reúne XP"
- http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf, Autor: "joseJoskowicz", Titulo: "Reglas y practicas de XP"

- http://www.willydev.net/descargas/masyxp.pdf, Autor: “Patricio Letelier y Maria Carmen Penadés”, Titulo: “Metodologias agiles para el desarrollo de software”
- http://www.info-abuclm.es/asignaturas/42551/trabajosAnteriores/Presentacion-XP.pdf
- http://eisc.univalle.edu.co/materias/WWW/material/lecturas/xp.pdf
- http://www.slideshare.net/guest51797f/uml-xp-02, Autor. “Armando de Andreis”, Titulo: “Vista estructural”
- http://www.hipertexto.info/documentos/b_datos.htm, Autor: “María Jesús Lamarca Lapuente”, Titulo: “Bases de datos”
- http://www.degerencia.com/idea/las_bases_de_datos_mas_grandes_del_mundo, Autor: “Desconocido”, Titulo: “Las bases de datos mas grandes del mundo”
- http://www.entrebits.cl/foros/temas-generales/13922-lista-de-bases-de-datos-populares-con-servidor.html, Autor: “Desconocido”, Titulo: “MySQL”
- http://es.wikipedia.org/wiki/Anexo:Comparaci%C3%B3n_de_sistemas_administradores_de_bases_de_datos_relacionales, Autor: “Desconocido”, Titulo: “Motores de bases de datos”
- http://www.mitecnologico.com/Main/ComparacionEntreBasesDeDatosOrientadasAObjetosYLasBasesDeDatosRelacionalesOrientadasAObjetos, Autor: “Lauro soto”, Titulo: “Comparacion de las bases de datos”
- http://html.rincondelvago.com/base-de-datos_3.html, Autor: “Desconocido”, Titulo: “Bases de datos”
- http://www.tufuncion.com/tendencias-lenguajes-programacion, Autor: “Desconocido”, Titulo: “Los lenguajes de programación mas populares”

- http://www.tufuncion.com/diferentes-lenguajes-programacion, Autor: “Desconocido”, Titulo: “8 lenguajes de programación que deberías conocer”
- http://www.webtaller.com/maletin/articulos/lenguajes-programacion-web-2.php, Autor: “Carlos Castillo”, Titulo: “Lenguajes de programación para la web”
- http://www.larevistainformatica.com/lenguajes-programacion-web.htm, Autor: “Desconocido”, Titulo: “Lenguaje de programación web”
- http://www.tejedoresdelweb.com/w/Lenguajes_programaci%C3%B3n_web, Autor: “Desconocido”, Titulo: “Lenguajes programación web”

ANEXOS

ANEXO 1

MANUAL DE USUARIO

MANUAL DE USUARIO

A continuación se describe un manual fácil de entender y ponerlo en práctica en caso que se lo requiera, como el usuario final debe manejar el sistema AVANTANSIA.

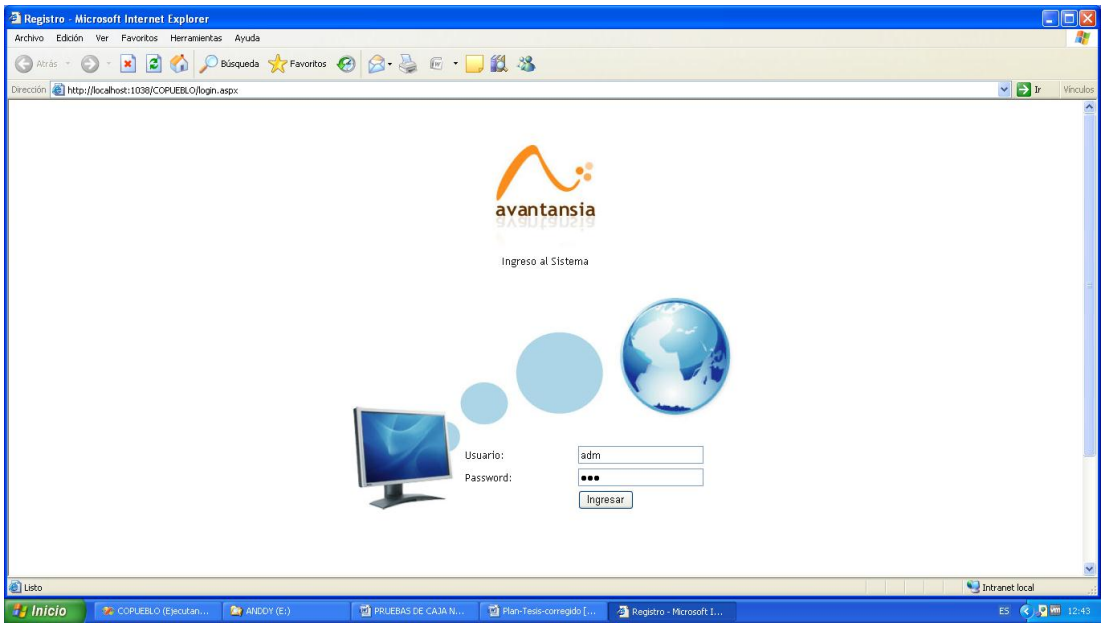


Figura 1: Pantalla de ingreso del sistema

La imagen anterior muestra el ingreso al sistema en donde se ingresa el usuario y el password.

Una vez que el usuario ha ingresado sus datos correctamente, podrá ingresar al sistema y realizar las acciones pertinentes.

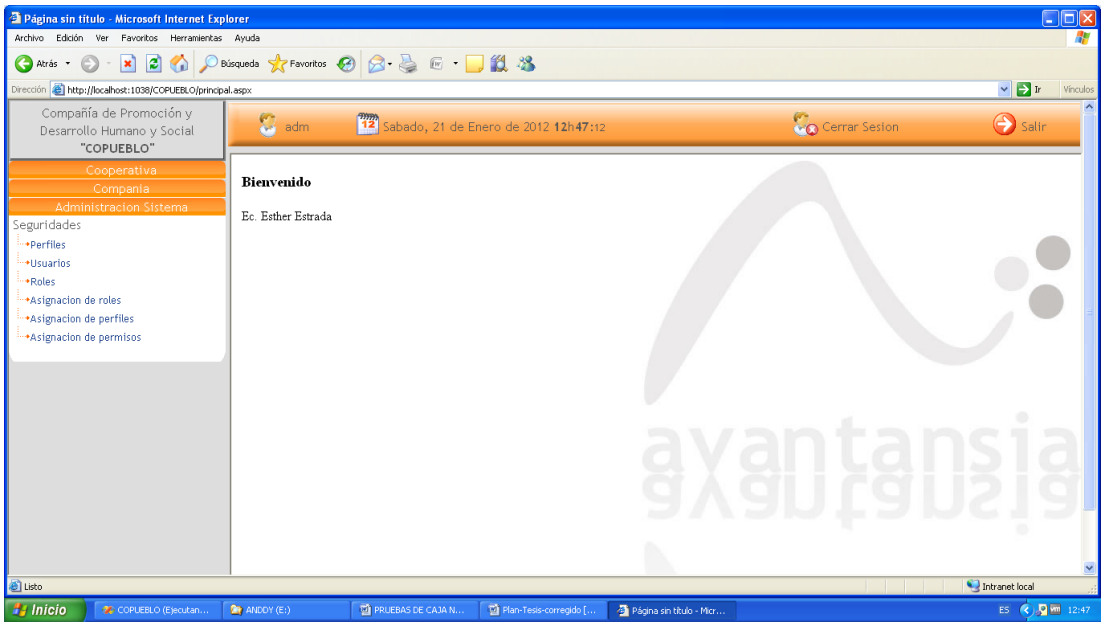


Figura 2: Pantalla principal del sistema

Aquí se visualiza la pantalla principal del sistema la que contiene los 3 roles que son: Cooperativa, compañía y administración del sistema.

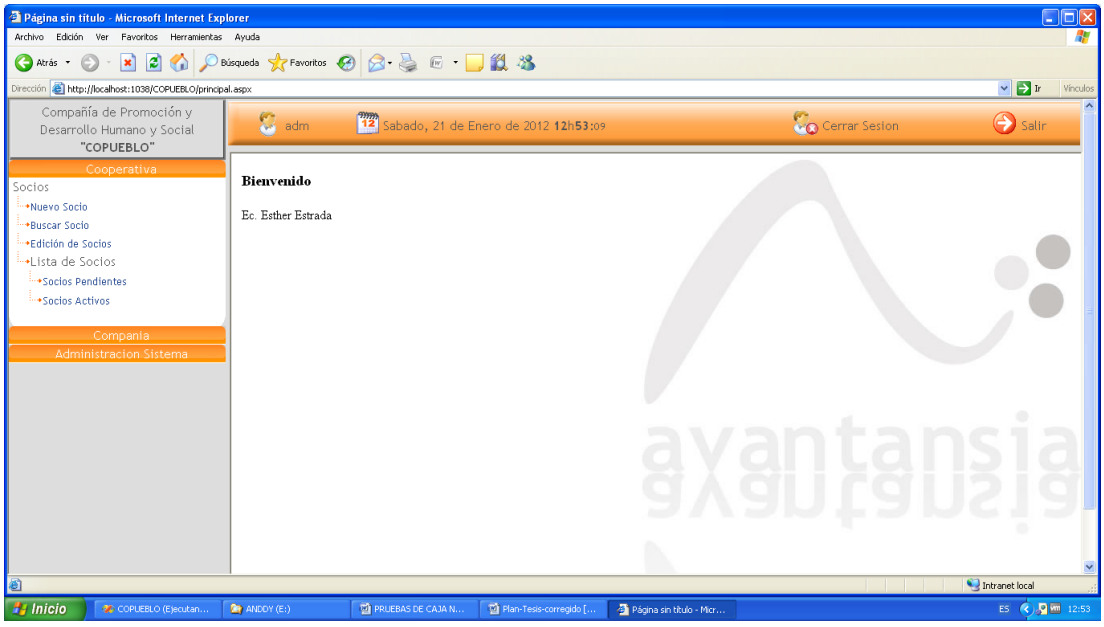


Figura 3: Pantalla de roles con los sub-menú

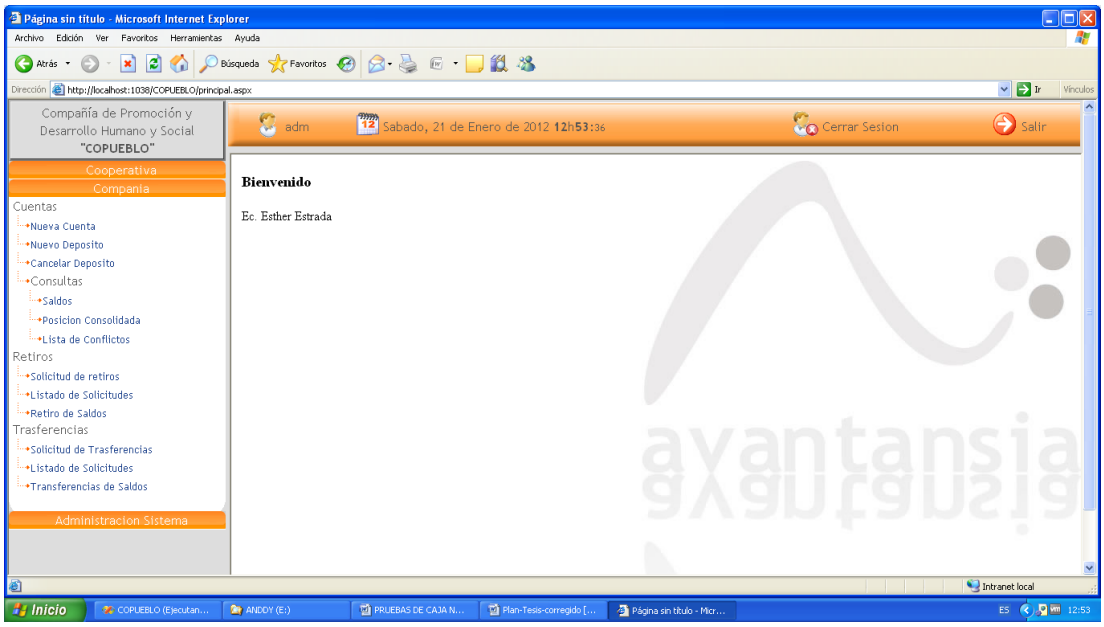


Figura 4: Pantalla de roles con los sub-menú

Como se visualiza cada rol contiene un sub-menú en los cuales tiene las funciones que se pueden realizar.

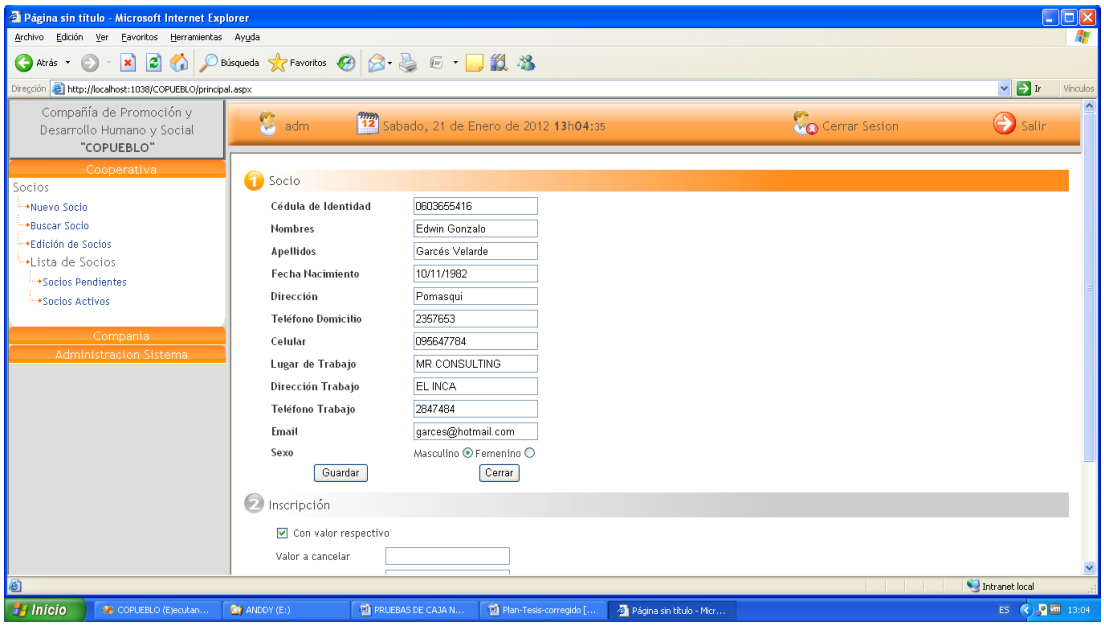


Figura 5: Pantalla de ingreso de los datos

Se puede visualizar el ingreso de los datos del nuevo socio.

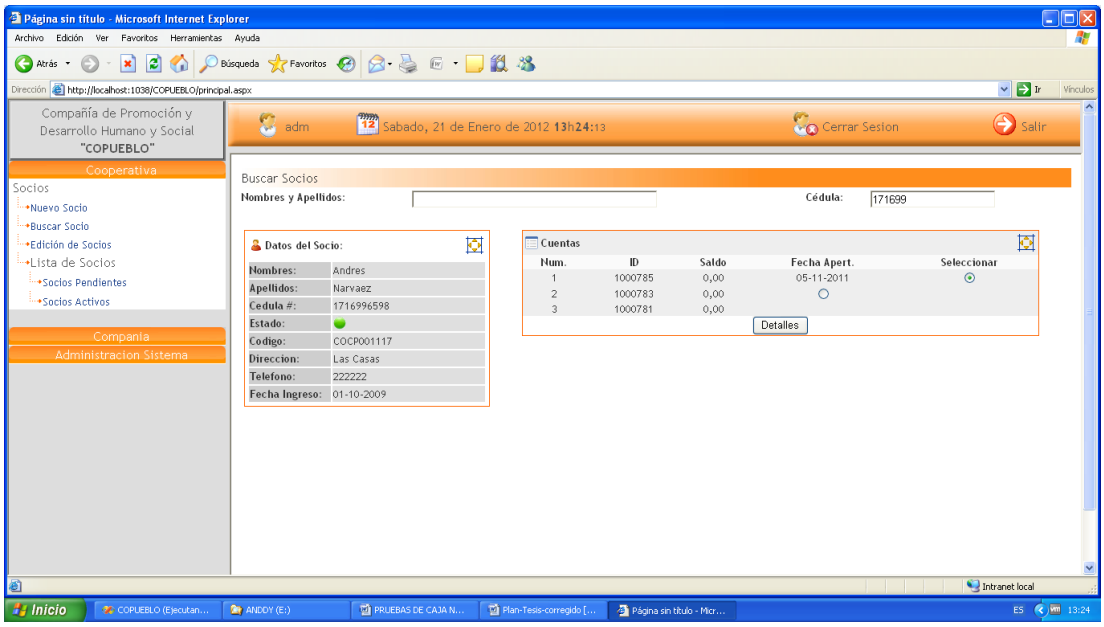


Figura 6: Pantalla de búsqueda del socio

En esta pantalla se muestra la búsqueda de un socio ya ingresado

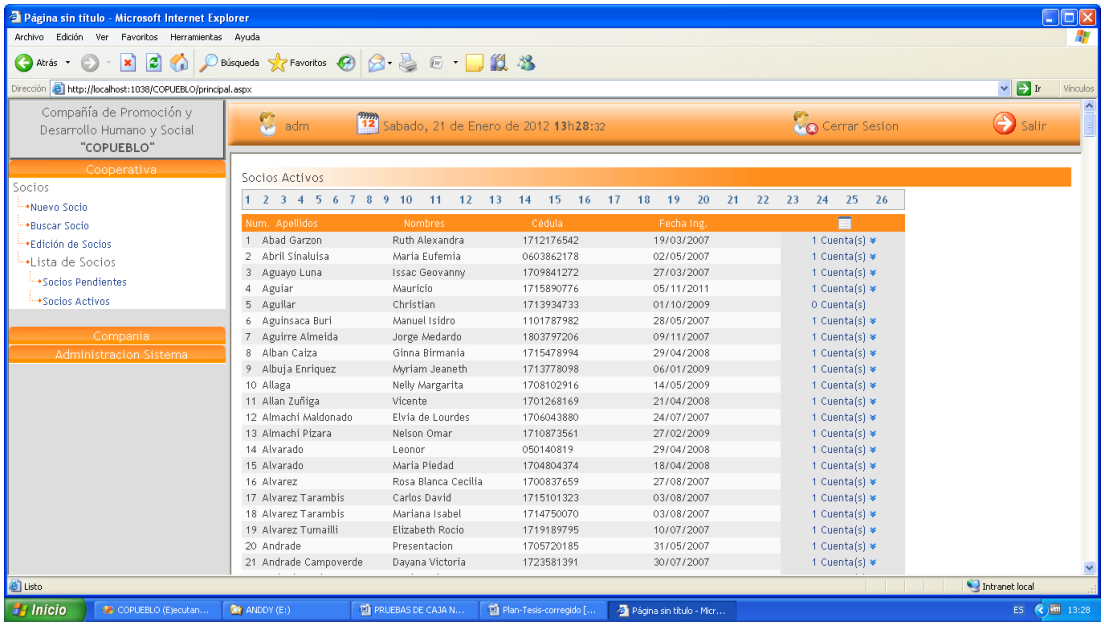


Figura 7: Lista de los socios activos

Se muestra la lista de los socios activos en el sistema.

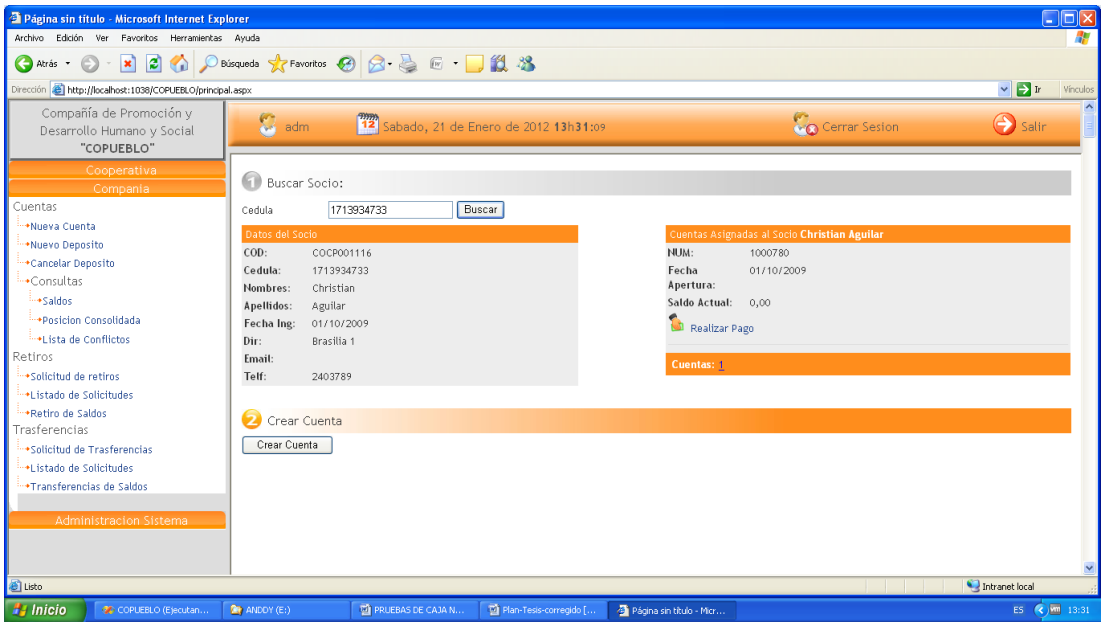


Figura 8: Pantalla de creación de una nueva cuenta

En esta pantalla se busca al cliente por el número de cedula para crear una nueva cuenta.

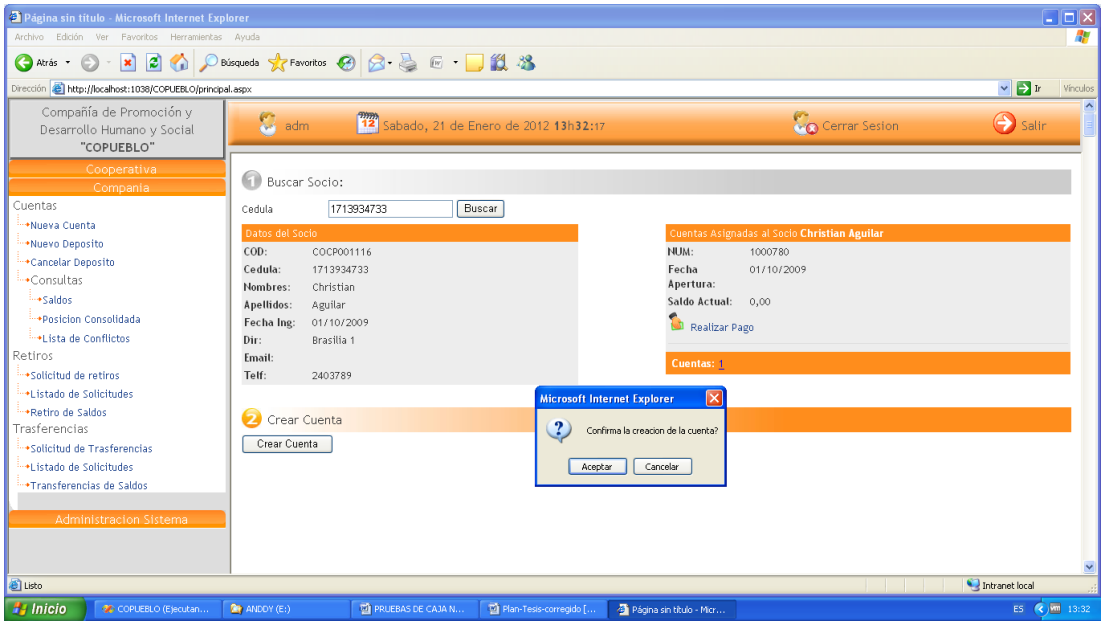


Figura 9: Mensaje de confirmación

Se despliega un mensaje para la confirmación de la nueva cuenta.

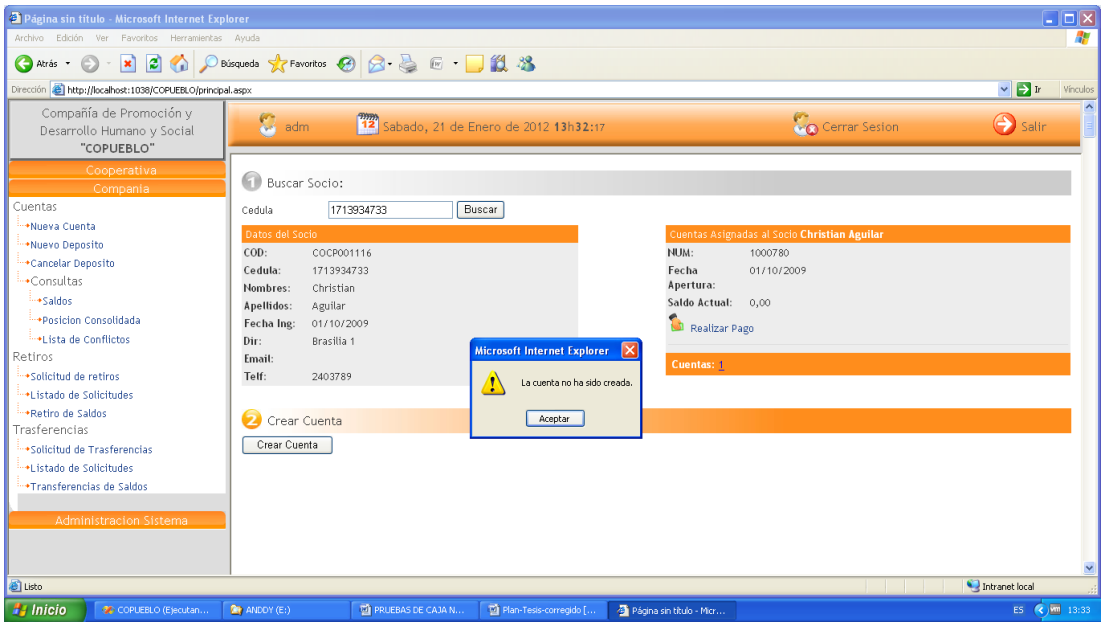


Figura 10: Pantalla de cuenta creada

Luego despliega el mensaje que la cuenta ha sido creada, automáticamente se le asigna un número de cuenta.

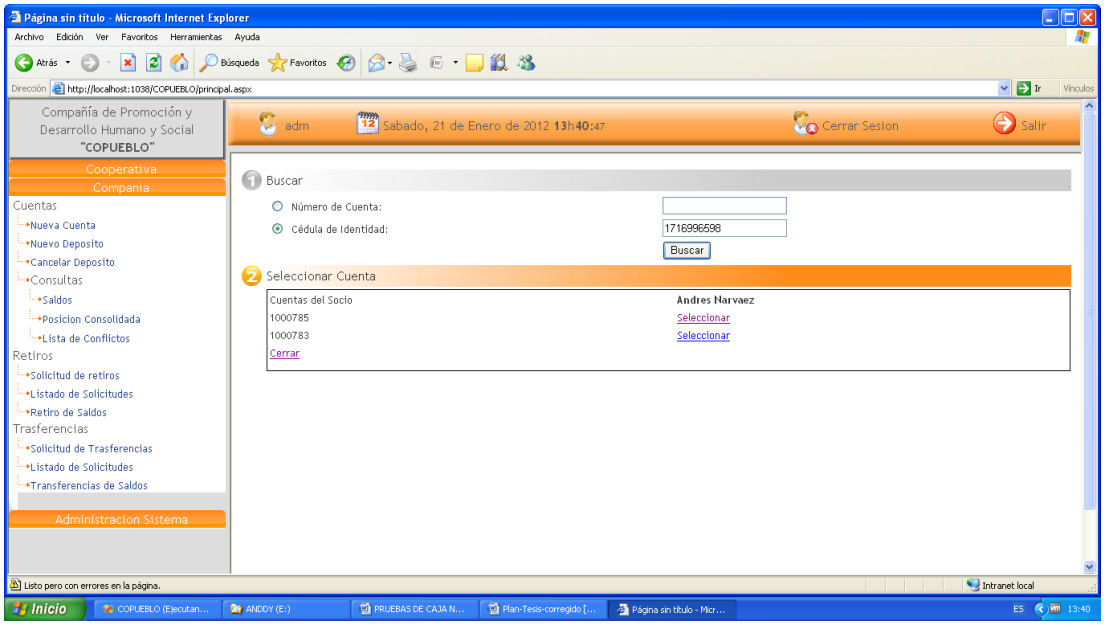


Figura 11: Pantalla de ingreso de un depósito nuevo

Para el ingreso de un nuevo depósito primero se busca a través de 2 opciones: por el número de cuenta o por la cedula y despliega la cuenta o las cuentas que posee el socio.

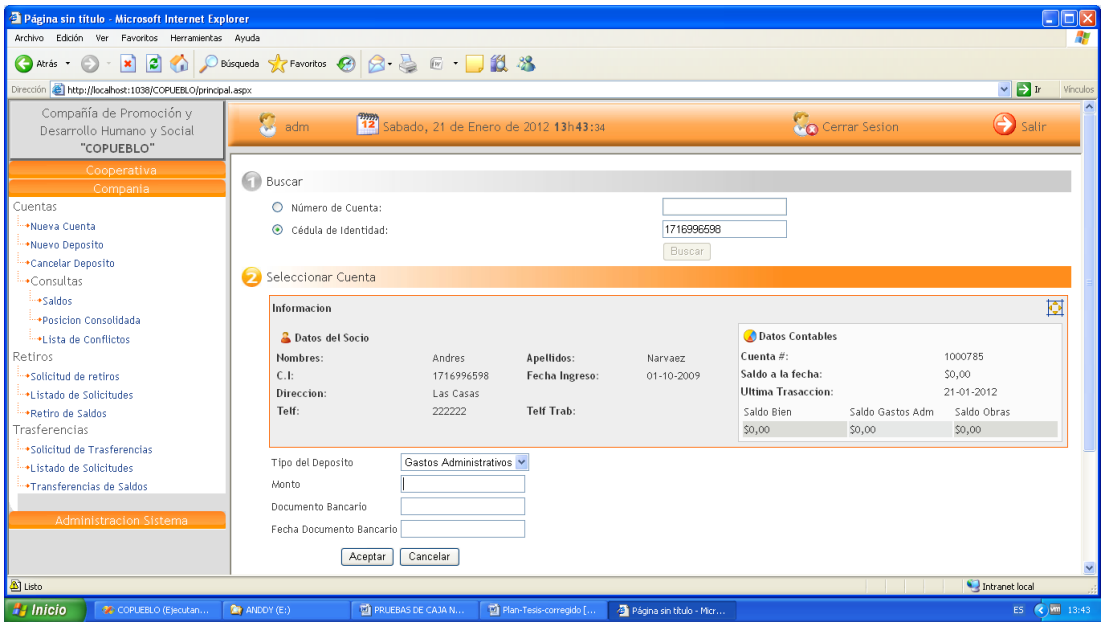


Figura 12: Pantalla de ingreso de tipo de depósito

Luego se selecciona la cuenta y se procede al ingreso del tipo del depósito, monto documento bancario y la fecha del mismo. Los datos del documento bancario son los emitidos por el banco al momento que se realiza el depósito.

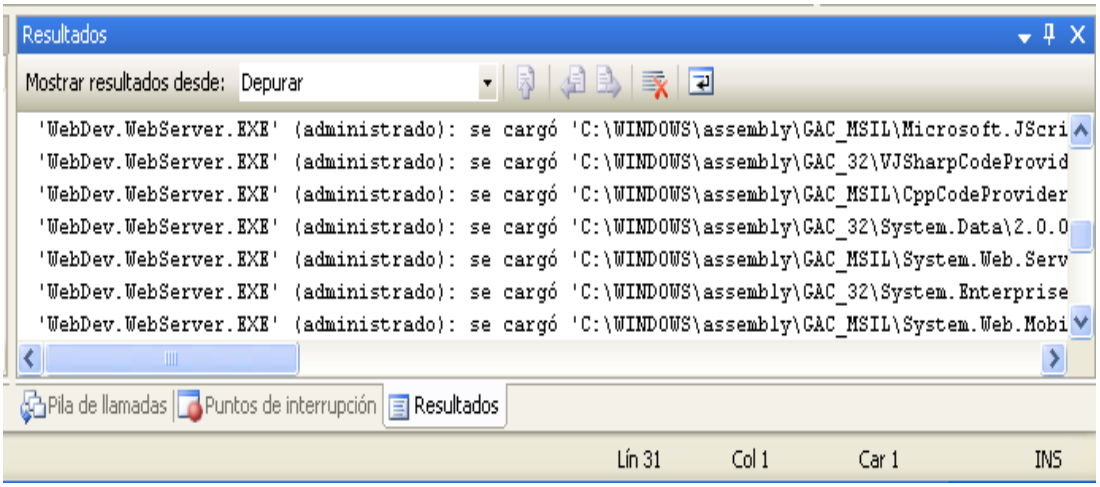


Figura 13: Pantalla de depuración del código

En esta pantalla se muestra la depuración del código el cual se ejecuta sin ningún tipo de bug (error).

Como se muestra en las anteriores pantallas se ingresan los datos sin ningún problema salvo que al momento de que se ingrese un dato no valido el sistema automáticamente devuelve un mensaje de error.

ANEXO 2

MANUAL DE INSTALACIÓN

MANUAL DE INSTALACIÓN

En el presente manual se dará una guía de cómo instalar Internet Information Server (IIS), para alojar una aplicación web, en este caso para nuestro proyecto el software se programó en el lenguaje de ASP.NET.

También se explicará cómo se puede crear un sitio Web local y configurarlo para que se ejecute páginas de ASP.NET.

En primer lugar, debe iniciar el Administrador IIS.

Importante

Debe iniciar una sesión como miembro del grupo Administradores en el equipo local para realizar el procedimiento (o procedimientos) siguientes, o debe haber delegado en la autoridad adecuada.

Figura 14: IIS

El trabajo de desarrollo está basado, entre otros pilares, en sistemas informáticos y herramientas tecnológicas que **requieren profesionales con conocimientos precisos y extensos sobre la instalación**, configuración y soporte del hardware y el software: los compañeros IT.

Pero no siempre tendremos a mano a un Administrador de Sistemas que nos configure el entorno de trabajo o de producción, por lo cual hay labores que son prácticamente imprescindibles de conocer **para poder ser autónomos en nuestro trabajo** sin vernos bloqueados por el entorno de desarrollo.

Para iniciar el Administrador de IIS desde el cuadro de diálogo Ejecutar

1. En el menú Inicio, haga clic en Ejecutar.
2. En el cuadro Abrir, escriba inetmgr y haga clic en Aceptar.

Para iniciar el Administrador de IIS desde la consola Servicios administrativos

1. En el menú Inicio, haga clic en Ejecutar. En la ventana de texto Ejecutar, escriba panel de control y, a continuación, haga clic en Aceptar.
2. En la ventana Panel de control, haga clic en Herramientas administrativas.
3. En la ventana Herramientas administrativas, haga clic en Servicios de Internet Information Server.

1.7 Instalando el Servidor Web

Figura 15: Configuración del servidor web

1 En el Administrador IIS, expanda el equipo local, haga clic con el botón secundario en la carpeta Sitios Web, señale a Nuevo y haga clic en Sitio Web.

2 En el Asistente para crear un sitio Web, haga clic en Siguiente.

3 En el cuadro Descripción, escriba una etiqueta descriptiva para el sitio (esta etiqueta no es lo que escribirá en la barra de direcciones de un explorador) y, a continuación, haga clic en Siguiente.

4 Escriba o seleccione la dirección IP (el valor predeterminado es Ninguna asignada), el puerto TCP y el encabezado de host (por ejemplo, www.microsoft.contoso.com) para el sitio.

Importante

Para asegurarse de que las solicitudes del usuario llegan al sitio Web correcto, debe diferenciar cada sitio del servidor con al menos uno de los tres identificadores únicos: un nombre de encabezado de host, una dirección IP o un número de puerto

TCP. El uso de nombres de encabezado de host únicos es la manera preferida de identificar varios sitios Web en un servidor único. Para llegar al sitio, un cliente debe escribir el par nombre/dirección IP que aparece para el sitio en el servidor DNS o en el archivo local HOSTS.

5 En el cuadro Ruta de acceso, escriba o vaya al directorio que incluye, o incluirá, el contenido del sitio y, a continuación, haga clic en Siguiente.

6 Active las casillas de verificación para los permisos de acceso que desea asignar a sus usuarios y, a continuación, haga clic en Finalizar. De manera predeterminada, las casillas de verificación Lectura y Ejecutar secuencias de comandos aparecen seleccionadas; estos permisos permiten ejecutar páginas ASP.NET para muchos escenarios comunes.

2 Configurar un sitio Web local

Después de crear un nuevo sitio Web local, puede configurarlo para que ejecute páginas ASP.NET y definir su seguridad. La tabla siguiente muestra la configuración de los permisos que están disponibles en todas las versiones de IIS.

Cuenta o grupo	**Permisos**

Administradores	Control total
Sistema	Control total
Cuenta o grupo que se elige a fin de proporcionar acceso para explorar el sitio si se ha deshabilitado la autenticación anónima al crear el directorio virtual.	Leer y Ejecutar
Cuenta configurada para obtener acceso a los recursos del sistema para el contexto de usuario actual de ASP.NET, como la cuenta de Network Service (IIS 6.0) o la cuenta ASPNET (IIS 5.0 y 5.1).	Mostrar contenido de carpeta lectura y escritura

Tabla 1: Configuración de permisos

Una vez configurado el sitio, se pueden agregar páginas Web ASP.NET al directorio del sitio.

2.1 Configurando el Sitio Web

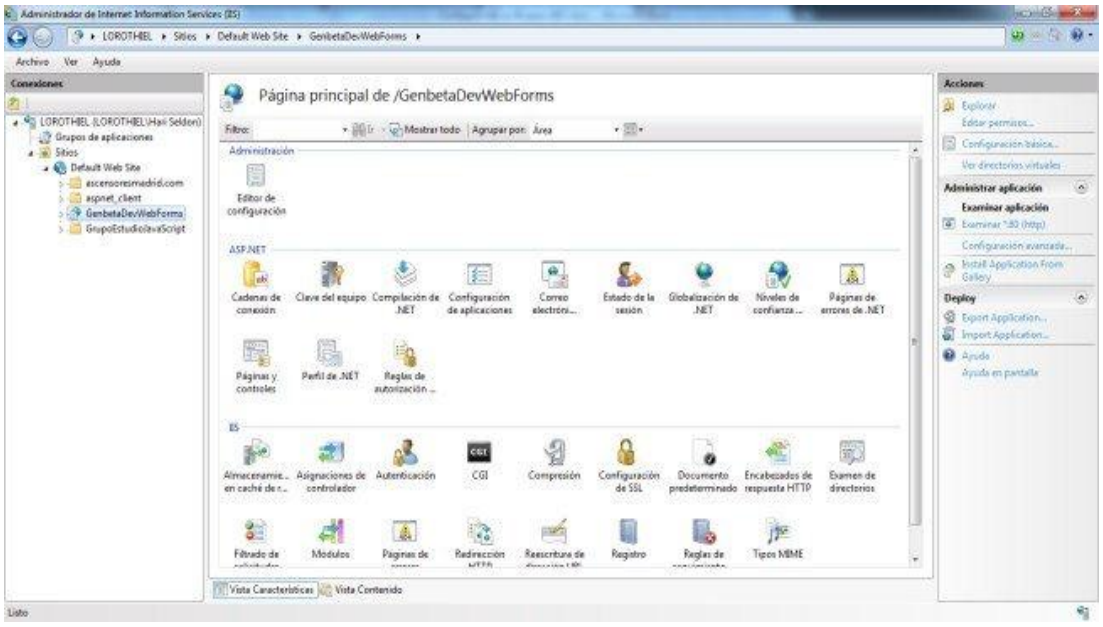


Figura 16: Configuración del sitio web

Visual Studio, por defecto, al crear un nuevo proyecto Web, sitúa la carpeta con el código en la ruta de *Mis Documentos* del usuario, dentro de la carpeta de proyectos (dependiendo de la versión del VS). Esto **es muy cómodo cuando se realizan las pruebas de funcionamiento por medio del "***F5***"**, que lanza la depuración del proyecto y lo despliega en el servidor de desarrollo del propio Visual Studio.

Esta forma de trabajar con la aplicación **se convierte muy lento** y el tener que arrancar la depuración completa cuando, por ejemplo, se está revisando cosas de diseño en el front-end. Para ello lo más inmediato es crear un directorio virtual en el IIS que se ha instalado anteriormente para poder lanzar la aplicación sin necesidad de tener, tan siquiera, el Visual Studio abierto.

Esta operación es sencilla: se abre la consola de Internet Information Server escribiendo "*IIS*" en la casilla de buscar del botón de *Inicio*, a continuación botón derecho al icono del *Web Site por Defecto* y selecciono *Agregar aplicación*. Aquí se escoge el Alias con el que voy a acceder a la Web (por ejemplo si utilizo GenbetaDevWebForms, para acceder a la aplicación pondré en el navegador *localhost\GenbetaDevWebForms*) y el directorio físico del WebSite.

Con estos pasos la aplicación está publicada en el servidor Web. Lo mejor es que esto que hemos realizado **es exactamente lo mismo que se hace para un servidor de producción** pero teniendo cuidado con los Grupos de aplicaciones.

2.1.1 Para configurar la seguridad y la autenticación para un sitio Web local

1. En el Administrador IIS, haga clic con el botón secundario del mouse en el nodo para el sitio que desea configurar y, a continuación, haga clic en Propiedades.
2. Haga clic en la ficha Seguridad de directorios y, a continuación, en la sección Autenticación y control de acceso, haga clic en Editar.
3. Active la casilla de verificación para el método o métodos de autenticación que desea utilizar para el sitio y, a continuación, haga clic en Aceptar. Las

casillas de verificación Habilitar el acceso anónimo y Autenticación integrada de Windows están seleccionadas de manera predeterminada.

4. En el Explorador de Windows, desplácese a la carpeta que va a contener las páginas del sitio. Haga clic con el botón secundario del mouse en la carpeta y, a continuación, haga clic en Compartir y seguridad en el menú contextual.
5. En la ficha Seguridad, configure las cuentas y los permisos adicionales mínimamente necesarios para ejecutar el sitio Web y, a continuación, haga clic en Aceptar. Algunas de las cuentas mostradas, como Administradores y Sistema, ya están configuradas de manera predeterminada.

Nota

Para cambiar los permisos de una cuenta ya existente, selecciónela en la lista Nombres de grupos o usuarios y, a continuación, active las casillas de verificación de los permisos correspondientes. Para agregar una nueva cuenta, haga clic en Agregar y, a continuación, en el botón Ubicaciones. Seleccione el nombre del equipo local en la lista y haga clic en Aceptar. A continuación, escriba en el cuadro de texto el nombre de la cuenta que desee agregar. Después de escribir un nombre, haga clic en Comprobar nombres para comprobar el nombre de cuenta y, por último, haga clic en Aceptar para agregar la cuenta.

2.2 Depuración y web.config

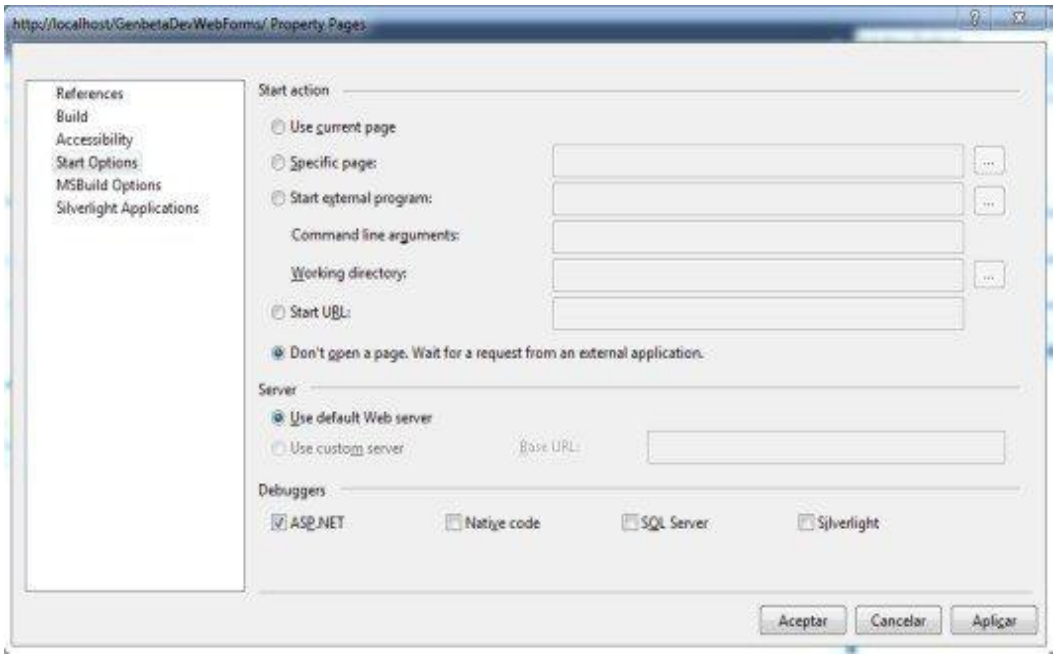


Figura 17: Configuración del sitio web

Una vez que se tiene configurado y funcionando el Site del proyecto Web, llega el momento de eliminar la necesidad de lanzar la aplicación desde el "*debugueador"* del propio Visual Studio para poder trabajar con ella. Para ello se sitúa en la raíz del proyecto web, se pulsa el botón derecho y se selecciona *la página de Opciones*. En la ventana que se abre se elije las opciones de Inicio (Start Options) y se marca la acción de inicio que indica que no abra la página. Que espere la petición desde una aplicación externa.

A partir de entonces se puede decidir cuándo se quiere depurar la aplicación o **cuando simplemente se quiere desarrollar** sin el depurador activo. Es muy cómodo y, sobre todo, se gana mucha velocidad.

Otra ventaja que se obtiene utilizando el IIS como servidor de desarrollo, es darle las máximas facilidades a los compañeros de sistemas o, incluso, al cliente de poder cambiar las configuraciones del web.config **sin necesidad de tener que saber nada sobre .net** y su entorno de desarrollo.

Accediendo al *Editor de Configuración*, **se pueden realizar altas, bajas y modificaciones** de las diferentes secciones que pueden componer el fichero de

configuración del Sitio .NET sin tener acceso físico a la ruta de publicación.