UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA EN SISTEMAS Y COMPUTACIÓN

"Trabajo de grado previo a la obtención del Título de Ingeniera en Sistemas y Computación."

TRABAJO DE GRADUACIÓN

Título del proyecto

ESTUDIO DE TECNOLOGÍAS OPEN SOURCE Y SU INCIDENCIA EN EL COSTO DEL DESARROLLO DE APLICACIONES WEB. CASO APLICATIVO: SISTEMA DE FACTURACIÓN EN LA EMPRESA NACHOS SPORT.

Autor(es): Raúl Alonso Calderón Alvares.

Martha Cecilia Tierra Macas.

Directora: Ing. Lady Espinoza.

Riobamba – Ecuador

2011

UNIVERSIDAD NACIONAL DE CHIMBORAZO
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA EN SISTEMAS Y COMPUTACIÓN

Título del proyecto

ESTUDIO DE TECNOLOGÍAS OPEN SOURCE Y SU INCIDENCIA EN EL COSTO DEL DESARROLLO DE APLICACIONES WEB. CASO APLICATIVO: SISTEMA DE FACTURACIÓN EN LA EMPRESA NACHOS SPORT.

Los miembros del Tribunal de Graduación del proyecto de investigación de título: ESTUDIO DE TECNOLOGÍAS OPEN SOURCE Y SU INCIDENCIA EN EL COSTO DEL DESARROLLO DE APLICACIONES WEB. CASO APLICATIVO: SISTEMA DE FACTURACIÓN EN LA EMPRESA NACHOS SPORT presentado por: Cecilia Tierra, Raúl Calderón y dirigida por: Ing. Lady Espinoza.

Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en la cual se ha constatado el cumplimiento de las observaciones realizadas, remite la presente para uso y custodia en la biblioteca de la Facultad de Ingeniería de la UNACH.

Para constancia de lo expuesto firman:

----------	**----------**
Presidente del Tribunal (nombre)	**Firma**
----------	**----------**
Miembro del Tribunal (nombre)	**Firma**
----------	**----------**
Miembro del Tribunal (nombre)	**Firma**

ESTUDIO DE TECNOLOGÍAS OPEN SOURCE Y SU INCIDENCIA EN EL COSTO DEL DESARROLLO DE APLICACIONES WEB. CASO APLICATIVO: SISTEMA DE FACTURACIÓN EN LA EMPRESA NACHOS SPORT

AUTORÍA DE LA INVESTIGACIÓN

"La responsabilidad del contenido de este Proyecto de Graduación, nos corresponde exclusivamente a: Cecilia Tierra, Raúl Calderón (autores) y del Ing. Lady Espinoza (director); y el patrimonio intelectual de la misma a la Universidad Nacional de Chimborazo.

AGRADECIMIENTO

A Dios, por ser nuestro creador, amparo y fortaleza, cuando más lo necesite y por hacer palpable su amor a través de cada uno de los que me rodean.
A mis padres, quienes han velado por mi bienestar y educación siendo mi apoyo primordial en todos los momentos de mi vida.
A mis profesores, que sin esperar nada a cambio, han sido pilares en nuestro camino y así, forman parte de este logro que nos abre puertas inimaginables en nuestro desarrollo profesional.

Martha Cecilia Tierra Macas.

AGRADECIMIENTO

A todos los miembros de mi familia ya que cada uno y a su manera incentivaron en mi la decisión y fortaleza de seguir adelante venciendo problemas y mirando al futuro.
A una persona en especial que siempre supo llegar a mi haciéndome ver que en la vida nada es fácil pero todo es posible

Raúl Alonso Calderón Alvares.

DEDICATORIA

A Dios, mis Padres, Hermanos y Amigos, que durante este largo camino me han brindado su apoyo incondicional demostrándome su amor, cariño y respeto en las labores que realizaba a diario. Mil gracias a todos.

Martha Cecilia Tierra Macas.

DEDICATORIA

A mi madre que con su sacrificio y esfuerzo
A inculcado en mi los valores necesarios que
me harán una persona de bien para la sociedad.
A mí querida abuelita que aunque no pudo ver
realizada mi meta desde el cielo siempre me
enviara sus bendiciones.

Raúl Alonso Calderón Alvares.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN.

Este proyecto consiste en el estudio de herramientas de software libre y su incidencia en el costo de desarrollo de aplicaciones web, estudio que se verá plasmado en la implementación de un sistema de facturación en la Empresa Nachos Sport, dedicada a la confección y venta de ropa deportiva dentro y fuera del país.

Las nuevas tecnologías de la información están brindando facilidades en la creación de software para Pymes ya que la utilización de software libre permite que las empresas puedan adquirir software necesarios para su desempeño en menor costo evitando pagos de licencias, y facilidad en la actualización o incremento del sistema, más allá de lo mencionado el uso de Open Source permitirá que la empresa se recupere en menor tiempo de inversión hecha para un software.

Para cumplir con la implementación del sistema, luego de realizar el estudio sobre las herramientas se escogieron las siguiente PHP como lenguaje de programación y MySQL como gestor de bases de datos, además herramientas de java para autentificación de usuarios y control de escritura.

La implantación del sistema mejorara el desempeño de las actividades de la empresa y brindara una mejor atención a los clientes, además generara reportes que ayudaran a tener un seguimiento de los ingresos de empresa en determinados periodos de tiempo, y así de esta manera la información estará segura y disponible en el momento que se necesite.

SUMMARY

This project involves the study of free software tools and their impact on the cost of web application development, a study that will be reflected in the implementation of a billing system Nachos Sport Company, engaged in the manufacture and sale of clothing sport within and outside the country.

New information technologies are providing facilities to the creation of software for SMEs and that the use of free software allows companies to buy software for less cost performance in avoiding licensing fees, and ease of upgrade or increase system, beyond that mentioned the use of open source will allow the company to recover in less time investment made to software.

To comply with the implementation of the system after the study on the following tools were selected as the programming language PHP and MySQL as database manager also java tools for user authentication and control of writing.

The implementation of the system will improve the performance of business activities and provide better customer service, as well as generate reports that help you keep track of business income for certain periods of time, and so this way the information will be safe and available when needed.

1. INTRODUCCIÓN.

Hoy en día cualquier empresa necesita un sistema informático capaz de facilitar la gestión administrativa y de producción, basada en procesos de calidad y eficiencia. Todo esto se puede conseguir mediante un sistema informático que interconecte las diferentes partes de la Empresa.

El presente proyecto de Tesis describe el estudio de tecnologías Open Source en el desarrollo de Aplicaciones Web de bajo costo aplicada en el Desarrollo del Sistema de Facturación en la Empresa Nachos Sport, el cual servirá como una herramienta para la planificación de la misma, ya que permitirá a la Empresa coordinar y planificar sus tareas.

A la fecha actual la Empresa Nachos Sport está conformada por un grupo de personas, que aspiran realizar un trabajo de mayor eficiencia inmerso en el mundo de la tecnología, y de este modo llegar de una mejor manera a sus clientes y proveedores, para lograr lo dicho, quienes conforman la Empresa desean implementar un Sistema que automatice las labores realizadas dentro de la misma.

Con el desarrollo e implementación de este sistema se logrará mayor seguridad en la gestión de la información, eficiencia en los procesos y simplificación del tiempo, con el uso del Sistema se irá acumulando datos sobre la historia de las acciones realizadas, es decir una base de datos que analizar, de donde sacar el rendimiento real de la empresa, que ámbitos están más aprovechados y cuales hay que mejorar.

2. ANTECEDENTES.

2.1. Planteamiento del Problema.

La problemática parte de saber que si bien los Open Source no tienen un precio de licenciamiento, no implica que no haya un costo en la fase de desarrollo, al existir gran variedad de herramientas dicho costo variará de una a otra, para lo que hasta la fecha no existe un estimado en precios refiriéndose a cada una de ellas, es por eso que la investigación persigue saber los costos de ciertas herramientas en la fase de desarrollo.

Para realizar esta investigación se deberán tomar en cuenta indicadores que nos darán a conocer cuál será la herramienta más idónea para desarrollo un sistema tomando en cuenta las necesidades de cada empresa.

Los resultados del estudio realizado serán aplicados en el sistema de facturación en la empresa Nachos Sport de este modo se obtendrán beneficios para la empresa ya se automatizaran procesos, la información estará más segura y disponible para cuando se lo requiera en el menor tiempo posible.

2.2. Formulación del Problema.

¿La Empresa Nachos Sport cuenta con un sistema de facturación implementado con Open Source? Definitivamente no, siendo este el proceso que la empresa necesita de forma indispensable no lo tiene, el desconocimiento y la falta de preparación de quienes están a cargo de la empresa es el factor incide en la falta de este recurso.

El uso de herramientas Open Source para el desarrollo de sistemas para microempresas, es en la actualidad un tema de vital importancia ya que al contrario el software comercial este no cubre pagos de licencia y su código puede ser compartido libremente.

2.3. Sistematización del problema.

- Con la implementación del sistema de facturación la empresa ahorrara recursos económicos, tecnológicos y humanos.
- Los procesos se realizara de una forma rápida disminuyendo el tiempo de ejecución de cada tarea.
- El sistema permitirá realizar reportes a diario, con el que se lograra verificar los ingresos que a diario va generar la empresa.
- El trabajar con herramientas Open Source permitirá tener un estimación del costo total de software en su fase de desarrollo.

2.4. Importancia de la Investigación.

El uso de herramientas Open Source para el desarrollo de sistemas para microempresas, es en la actualidad un tema de vital importancia ya que al contrario del software comercial este no cubre pagos de licencia y su código puede ser compartido libremente.

La problemática parte de saber que si bien los Open Source no tienen un precio de licenciamiento, no implica que no haya un costo en la fase de desarrollo, al existir gran variedad de herramientas dicho costo variará de una a otra, para lo que hasta la fecha no existe un estimado en precios refiriéndose a cada una de ellas, es por eso que la investigación persigue saber los costos de ciertas herramientas en la fase de desarrollo.

Para realizar esta investigación se deberán tomar en cuenta indicadores que nos darán a conocer cuál será la herramienta más idónea para desarrollar un sistema tomando en cuenta las necesidades de cada empresa.

Los resultados del estudio realizado serán aplicados en el sistema de facturación en la empresa Nachos Sport de este modo se obtendrán beneficios para la empresa ya se automatizaran procesos, la información estará más segura y disponible para cuando se lo requiera en el menor tiempo posible.

3. Justificación.

La tecnología en la actualidad obliga a Empresas de todo tipo a adaptarse a los cambios tecnológicos de este modo automatizar procesos que realice dicha empresa, es difícil saber qué tan beneficioso será para una Empresa el realizar estos cambios para algunos muy drásticas, es por eso que se debe tener claro qué tipo de Sistema debe ser implementado en cierta empresa dependiendo cada una de las necesidades de la misma.

Tomando en cuenta la situación actual de la Empresa y que la misma no cuenta con un Sistema Informático, se decidió realizar el estudio de tecnologías Open Source en el desarrollo de Aplicaciones Web aplicada en el Desarrollo del Sistema de Facturación en la Empresa Nachos Sport. El desarrollo de la investigación permitirá la automatización de los procesos que actualmente se lo realizan de manera manual en la entidad, y de esta forma se podrá apoyar de manera directa a las personas que desempeñan sus labores dentro de la Empresa, y además apoyar indirectamente a Clientes y Proveedores.

Actualmente la automatización de las operaciones para las microempresas es parte vital para su desarrollo. Pero al mismo tiempo el costo de licenciamiento representa un precio muy alto que a corto plazo las microempresas no podrán asumir. De ahí que como parte de esta investigación se estudiará la influencia en los costos de desarrollo de aplicaciones para microempresas utilizando herramientas de software libre.

4. Objetivos.

4.1. Objetivo General.

Estudiar las Tecnologías Open Source y su incidencia en el costo del desarrollo de aplicaciones web.

4.2. Objetivos Específicos.

- Realizar un estudio de las herramientas de software libre para el desarrollo de Aplicaciones web.
- Investigar y seleccionar indicadores que influyan en el costo de desarrollo de aplicaciones web.
- Analizar, diseñar e implementar el Sistema de Facturación en la Empresa Nachos Sport.

CAPITULO I

1. MARCO TEÓRICO.

1.1. Open Source.

Es un término que empezó a utilizarse en 1998 por algunos usuarios de la comunidad del software libre, usándolo como reemplazo al nombre original, en inglés, del software libre (free software), que no significaba exactamente lo que se pretendía (free significa a la vez "gratis" y "libre").

El significado obvio del término "código abierto" es "se puede mirar el código fuente", lo cual es un significado más exacto que el del software libre. El software de código abierto (OSS por sus siglas en inglés) es software para el que su código fuente está disponible públicamente. Un programa de código abierto puede ser software libre, pero también puede serlo un programa por el que hay que pagar. Los términos de licenciamiento específicos del Código Abierto varían respecto a lo que se puede hacer con ese código fuente. O sea, "abierto" no necesariamente es "gratis", aunque en su gran mayoría lo sea.

La idea que late detrás del Código Abierto (open source) es bien sencilla: cuando los programadores en internet pueden leer, modificar y redistribuir el código fuente de un programa, éste evoluciona, se desarrolla y mejora. Los usuarios lo adaptan a sus necesidades, corrigen sus errores. Y esto puede ocurrir a tal velocidad que el que está acostumbrado al ritmo de desarrollo de los programas comerciales no lo puede concebir.

A diferencia del Código Cerrado, el Código Abierto permite que varios programadores puedan leer, modificar y redistribuir el código fuente de un programa, por lo que ese programa evoluciona constantemente. La gente lo mejora, lo adapta y corrige sus errores a una velocidad impresionantemente mayor a la aplicada en el desarrollo de software convencional o cerrado, dando como resultado la producción de un mejor software.

Todo el mundo sabe que Internet es una de las más poderosas e importantes herramientas de la actualidad, pero pocos se dan cuenta de que la mayoría de los sitios web que existen, así como los correos electrónicos que a diario son intercambiados en la Red, son servidos por programas de Código Abierto, incluso, el importantísimo sistema de dominios, el cual nos permite referirnos a un servidor de Internet con palabras y no números, está basado completamente en Código Abierto. Así, y sin duda alguna, se puede afirmar que la Internet y su actual repercusión económica y social en el mundo existen gracias al Código Abierto.

Los orígenes del Código Abierto pueden trazarse hasta hace unos 20 años, en las tierras de la academia, sin embargo este movimiento cobra cada vez más importancia en la actualidad debido a su empleo en empresas de todos los tamaños así como en los gobiernos de varios países, ya que no puede hablarse de soberanía ni de seguridad nacional si un gobierno utiliza Software Cerrado producido por una empresa extranjera.

El Código Abierto está cambiando viejos esquemas y rígidas reglas de una forma pragmática y lógica, amenazando muchas veces a compañías establecidas (por ejemplo, Microsoft), quienes paradójicamente proclaman su superioridad en calidad y soporte, mientras prefieren basar su defensa en el terreno jurídico, en base a patentes y patrañas, en lugar de hacerlo en el plano de la excelencia técnica.

Con la cantidad óptima de recursos, bajos costos y adaptando tecnologías de Código Abierto con tecnologías propietarias, hemos logrado exitosos desarrollos e implantaciones de tecnología sin importar el tamaño del cliente o del proyecto, conózcalos.

1.2. Los Beneficios del Open Source.

Desde el punto de vista monetario, la ventaja obvia es la no existencia de costos de licencia para el producto en sí mismo. Otra diferencia importante es que se puede también disponer del código fuente, lo cual le brinda independencia del proveedor (conocido como "contribuyente original" en el lenguaje del Open Source). Debido a ello, tampoco se encuentra atado a su existencia ni a sus prioridades. No esté prisionero. Adicionalmente, si los honorarios por soporte de su proveedor se vuelven exorbitantes, puede contratar otro proveedor para que le brinde sus servicios.

Toda la información es abierta también; no existen políticas ocultas o censuras de la compañía. Si algo no está funcionando, usted no tendrá problemas en enterarse de ello rápidamente. Como consecuencia, los proyectos de Open Source son muy rápidos en reaccionar cuando existen problemas.

La comunidad de usuarios y desarrolladores hacen también una atractiva diferencia. Debido a la diversidad de usuarios, los productos Open Source son generalmente muy bien testeados y usted podrá obtener ayuda y consejos rápidamente.

El código abierto tiene mayor flexibilidad. Los usuarios de código abierto pueden ajustar el producto tanto como sea necesario para conseguir cubrir sus necesidades en formas que no son posibles sin el código fuente. Los usuarios pueden ajustar los productos ellos mismos, o encontrar quien pueda resolver él problema, que incluso podría ser el desarrollador original del producto.

Algunos han proclamado que esto crea un "peligro de disgregación", es decir, múltiples versiones incompatibles de un mismo producto. Esto sólo es un riesgo para los que creen que la competencia es demoníaca: también se tiene múltiples versiones de coches. Y en la práctica, el alto coste de mantener el programa por uno mismo hace que los cambios se reviertan a la comunidad.

En caso contrario, por ejemplo resuelve un problema particular cuya solución se necesitaba para una situación particular, supone también una ganancia para el usuario, porque le ha resuelto un problema que de otra forma no se hubiese podido.

1.3. Evitan problemas Legales.

La utilización de la mayoría de los software comerciales, implican licencias de software y el seguimiento de copias de software y su uso. Esto demanda mantener un registro y la exposición legal; ambos aumentan los costos. Así, las licencias de software y las copias, son una fuente de costos para los negocios, y un riesgo legal tanto para las empresas como para los individuos.

En muchos, quizás la mayoría o todos los negocios, tal seguimiento es imperfecto; algunas veces intencionalmente, generalmente no. En cualquier caso, esta imperfección expone al culpable a acciones legales (multas, juicios, arresto) por el quebranto de leyes y los derechos de autor.

La mayoría, casi todo, el software Open Source puede ser copiado y utiliza libremente. No existe seguimiento de licencias, ni por lo tanto costos relacionados, como así tampoco riesgos legales.

1.4. Licencias Open Source.

Definen los privilegios y restricciones que un usuario de la licencia debe seguir para utilizar, modificar o redistribuir el software Open Source.

Bajo la Definición Open Source, las licencias deben cumplir diez condiciones para ser consideradas licencias de software abierto:

1. Libre redistribución: el software debe poder ser regalado o vendido libremente.
2. Código fuente: el código fuente debe estar incluido u obtenerse libremente.
3. Trabajos derivados: la redistribución de modificaciones debe estar permitida.
4. Integridad del código fuente del autor: las licencias pueden requerir que las modificaciones sean redistribuidas solo como parches.
5. Sin discriminación de personas o grupos: nadie puede dejarse fuera.
6. Sin discriminación de áreas de iniciativa: los usuarios comerciales no pueden ser excluidos.
7. Distribución de la licencia: deben aplicarse los mismos derechos a todo el que reciba el programa
8. La licencia no debe ser específica de un producto: el programa no puede licenciarse solo como parte de una distribución mayor.
9. La licencia no debe restringir otro software: la licencia no puede obligara que algún otro software que sea distribuido con el software abierto deba también ser de código abierto.
10. La licencia debe ser tecnológicamente neutral: no debe requerirse la aceptación de la licencia por medio de un acceso por clic de ratón o de otra forma específica del medio de soporte del software.

1.5. Tipos de licencias.

Una licencia es aquella autorización formal con carácter contractual que un autor de un software da a un interesado para ejercer "actos de explotación legales". Pueden existir tantas licencias como acuerdos concretos se den entre el autor y el licenciatario. Desde el punto de vista del software libre, existen distintas variantes del concepto o grupos de licencias:

1.5.1. Licencias GPL

Una de las más utilizadas es la Licencia Pública General de GNU (GNUGPL). El autor conserva los derechos de autor (copyright), y permite la redistribución y modificación bajo términos diseñados para asegurarse de que todas las versiones modificadas del software permanecen bajo los términos más restrictivos de la propia GNUGPL. Esto hace que sea imposible crear un producto con partes no licenciadas GPL: el conjunto tiene que ser GPL.

Es decir, la licencia GNUGPL posibilita la modificación y redistribución del software, pero únicamente bajo esa misma licencia. Y añade que si se reutiliza en un mismo programa código "A" licenciado bajo licencia GNUGPL y código "B" licenciado bajo otro tipo de licencia libre, el código final "C", independientemente de la cantidad y calidad de cada uno de los códigos "A" y "B", debe estar bajo la licencia GNUGPL.

En la práctica esto hace que las licencias de software libre se dividan en dos grandes grupos, aquellas que pueden ser mezcladas con código licenciado bajo GNUGPL (y que inevitablemente desaparecerán en el proceso, al ser el código resultante licenciado bajo GNUGPL) y las que no lo permiten al incluir mayores u otros requisitos que no contemplan ni admiten la GNUGPL y que por lo tanto no pueden ser enlazadas ni mezcladas con código gobernado por la licencia GNUGPL.

En el sitio web oficial de GNU hay una lista de licencias que cumplen las condiciones impuestas por la GNUGPL y otras que no.
Aproximadamente el 60% del software licenciado como software libre emplea una licencia GPL.

1.5.2. Licencias AGPL.

La Licencia Pública General de Affero (en inglés Affero General Public License, también Affero GPL o AGPL) es una licencia copyleft derivada de la Licencia Pública General de GNU diseñada específicamente para asegurar la cooperación con la comunidad en el caso de software que corra en servidores de red.

La Affero GPL es íntegramente una GNUGPL con una cláusula nueva que añade la obligación de distribuir el software si éste se ejecuta para ofrecer servicios a través de una red de ordenadores.

La Free Software Foundation recomienda que el uso de la GNU AGPLv3 sea considerado para cualquier software que usualmente corra sobre una red.

1.5.3. Licencias estilo BSD.

Llamadas así porque se utilizan en gran cantidad de software distribuido junto a los sistemas operativos BSD. El autor, bajo tales licencias, mantiene la protección de copyright únicamente para la renuncia de garantía y para requerir la adecuada atribución de la autoría en trabajos derivados, pero permite la libre redistribución y modificación, incluso si dichos trabajos tienen propietario.

Son muy permisivas, tanto que son fácilmente absorbidas al ser mezcladas con la licencia GNUGPL con quienes son compatibles. Puede argumentarse que esta licencia asegura “verdadero” software libre, en el sentido que el usuario tiene libertad

ilimitada con respecto al software, y que puede decidir incluso redistribuirlo como no libre. Otras opiniones están orientadas a destacar que este tipo de licencia no contribuye al desarrollo de más software libre (normalmente utilizando la siguiente analogía: "una licencia BSD es más libre que una GPL si y sólo si se opina también que un país que permita la esclavitud es más libre que otro que no la permite").

1.5.4. Licencias estilo MPL y derivadas.

Esta licencia es de Software Libre y tiene un gran valor porque fue el instrumento que empleó Netscape Communications Corp. para liberar su Netscape Communicator 4.0 y empezar ese proyecto tan importante para el mundo del Software Libre: Mozilla.

Se utilizan en gran cantidad de productos de software libre de uso cotidiano en todo tipo de sistemas operativos. La MPL es Software Libre y promueve eficazmente la colaboración evitando el efecto "viral" de la GPL (si usas código licenciado GPL, tu desarrollo final tiene que estar licenciado GPL).

Desde un punto de vista del desarrollador la GPL presenta un inconveniente en este punto, y lamentablemente mucha gente se cierra en banda ante el uso de dicho código. No obstante la MPL no es tan excesivamente permisiva como las licencias tipo BSD.

1.5.5. Copyleft.

Hay que hacer constar que el titular de los derechos de autor (copyright) de un software bajo licencia copyleft puede también realizar una versión modificada bajo su copyright original, y venderla bajo cualquier licencia que desee, además de distribuir la versión original como software libre. Esta técnica ha sido usada como un modelo de negocio por una serie de empresas que realizan software libre (por ejemplo

MySQL); esta práctica *no* restringe ninguno de los derechos otorgados a los usuarios de la versión copyleft.

También podría retirar todas las licencias de software libre anteriormente otorgadas, pero esto obligaría a una indemnización a los titulares de las licencias en uso. En España, toda obra derivada está tan protegida como una original, siempre que la obra derivada parta de una autorización contractual con el autor.

1.6. Diferencias software libre y software propietario.

Esta es una relación básica de diferencias prácticas entre el software propietario ya que está liberado bajo la licencia GPL.

Tabla 1. Diferencia entre software libre y comercial.

Aspecto a considerar	Software propietario	Software libre (GPL)
Acceso al código fuente	Prohibido por licencia	Si, Garantizado
Corrección de errores en el programa por el cliente/usuario	No	Si
Duplicación del software	Prohibido	Posible y Recomendada
Libertad de Competencia para el Mantenimiento.	No, depende del fabricante	Si imposible limitarla
Posibilidad de examinar el código del producto.	Prohibido, salvo permiso fabricante	Si
Venta de segunda mano	Prohibido	N/A
Respeto a estándares globales	En función del fabricante	En la mayoría de los casos
Adaptaciones al cliente (P.e.Idioma)	En función del fabricante	Disponible
Virus, gusanos	Frecuentes	Muy Infrecuentes

1.7. Ejemplos de Software libre vs Software propietario.

Hoy en día existen multitud de alternativas para casi el 100 % de las tareas necesarias .Aquí se incluyen una tabla con soluciones propietarias y sus alternativas en entornos libres.

Tabla 2. Ejemplos de software libre versus propietario.

Ejemplos de Software libre vs Software propietario		
Nombre	**Software libre**	**Software propietario**
Navegador	Internet Explorer, Netscape /Mozilla for Windows, Opera, ...	1) Netscape / Mozilla. 2) Galeon. 3) Konqueror. 4) Nautilus.
Suite ofimática	MS Office, StarOffice / OpenOffice	1) Openoffice. 2) Koffice.
Procesador de textos	Word, StarOffice / OpenOffice Writer, 602Text	1) Abiword. 2) StarOffice 3) OpenOffice 4) Kword.
Hoja de cálculo	Excel, StarOffice / OpenOffice Calc	1) Gnumeric. 2) / OpenOffice Calc. 3) StarOffice 4) Kspread.
Gráficos y dibujo	Excel	1) Kivio. 2) Dia. 3) KChart. 4) Gnuplot.
Creación presentaciones	MS PowerPoint, StarOffice Presentation, OpenOffice Impress	1) StarOffice Presentation. 2) OpenOffice Impress. 3) Kpresenter.

		4) MagicPoint.
Base de datos Local .	Access	1) KNoda. 2) Gnome DB Manager. 3) OpenOffice + MySQL
Gestor de finanzas	Personales MS Money, Quicken	1) GNUcash. 2) GnoFin. 3) Kmymoney. 4) Grisbi.
Gestión de Proyectos	MS Project, Project Expert 7	Mr Project.
Cliente correo electrónico como MS Outlook	Outlook	1) Evolution.
Base de Datos	MS SQL, MySQL for Windows	1) PostgreSQL. . 2) MySQL. 3) mSQL. 4) SAP DB.
Servidor web	Internet Information Server, Apache para Windows, roxen	1) Apache.

CAPITULO II

2. TECNOLOGÍAS OPEN SOURCE PARA EL DESARROLLO DE APLICACIONES WEB.

Desde los inicios de Internet, fueron surgiendo diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. A medida que paso el tiempo, las tecnologías fueron desarrollándose y surgieron nuevos problemas a dar solución. Esto dio lugar a desarrollar lenguajes de programación para las web dinámicas, que permitieran interactuar con los usuarios y utilizaran sistemas de Bases de Datos. A continuación se dará una introducción a los diferentes lenguajes de programación para la web.

2.1. DESCRIPCION DE LAS DISTINTAS HERRAMIENTAS.

2.1.1. Lenguaje Html.

Desde el surgimiento de internet se han publicado sitios web gracias al lenguaje HTML. Es un lenguaje estático para el desarrollo de sitios web (acrónimo en inglés de Hyper Text Markup Language, en español Lenguaje de Marcas Hipert extuales). Desarrollado por el World Wide Web Consortium (W3C). Los archivos pueden tener las extensiones (htm, html).

Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento.

Sintaxis:

<html> (Inicio del documento HTML)

<head> (Cabecera)

</head>

<body> (Cuerpo)

</body>

</html>

 Negrita

 Definir párrafo

<etiqueta> Apertura de la etiqueta

</etiqueta> Cierre de la etiqueta

Ventajas:

- Sencillo que permite describir hipertexto.
- Texto presentado de forma estructurada y agradable.
- No necesita de grandes conocimientos cuando se cuenta con un editor de páginas web o WYSIWYG.
- Archivos pequeños.
- Despliegue rápido.
- Lenguaje de fácil aprendizaje.

- Lo admiten todos los exploradores.

Desventajas:

- Lenguaje estático.
- La interpretación de cada navegador puede ser diferente.
- Guarda muchas etiquetas que pueden convertirse en “basura” y dificultan la corrección.
- El diseño es más lento.
- Las etiquetas son muy limitadas.

2.1.2. Lenguaje Javascript.

Este es un lenguaje interpretado, no requiere compilación. Fue creado por BrendanEich en la empresa Netscape Communications. Utilizado principalmente en páginas web. Es similar a Java, aunque no es un lenguaje orientado a objetos, el mismo no dispone de herencias. La mayoría de los navegadores en sus últimas versiones interpretan código Javascript.

El código Javascript puede ser integrado dentro de nuestras páginas web. Para evitar incompatibilidades el World Wide Web Consortium (W3C) diseño un estándar denominado DOM (en inglés DocumentObjectModel, en su traducción al español Modelo de Objetos del Documento).

Sintaxis: <script type="text/javascript"> ... </script>

Ventajas:

- Lenguaje de scripting seguro y fiable.
- Los script tienen capacidades limitadas, por razones de seguridad.
- El código Javascript se ejecuta en el cliente.

Desventajas:

- Código visible por cualquier usuario.
- El código debe descargarse completamente.
- Puede poner en riesgo la seguridad del sitio, con el actual problema llamado XSS (significa en inglés Cross Site Scripting renombrado a XSS por su similitud con las hojas de estilo CSS).

2.1.3. Lenguaje Php.

Es un lenguaje de programación utilizado para la creación de sitio web. PHP es un acrónimo recursivo que significa "PHP Hypertext Pre-processor", (inicialmente se llamó Personal Home Page). Surgió en 1995, desarrollado por PHP Group.

PHP es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. PHP no necesita ser compilado para ejecutarse. Para su funcionamiento necesita tener instalado Apache o IIS con las librerías de PHP. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas. Los archivos cuentan con la extensión (php).

PHP es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor web en forma de módulo o ejecutado como un binario CGI separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. Estas propiedades hacen que cualquier cosa que sea ejecutada en un servidor web sea insegura por naturaleza.

PHP está diseñado específicamente para ser un lenguaje más seguro para escribir programas CGI que Perl o C, y con la selección correcta de opciones de configuración en tiempos de compilación y ejecución, y siguiendo algunas prácticas correctas de programación.

Sintaxis:

La sintaxis utilizada para incorporar código PHP es la siguiente:

```
<?
$mensaje = “Hola”;
echo $mensaje;
?>
```

También puede usarse:

```
<?php
$mensaje = “Hola”;
echo $mensaje;
?>
```

Ventajas:

- Muy fácil de aprender.
- Se caracteriza por ser un lenguaje muy rápido.
- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- Es un lenguaje multiplataforma: Linux, Windows, entre otros.
- Capacidad de conexión con la mayoría de los manejadores de base de datos: MysSQL, PostgreSQL, Oracle, MS SQL Server, entre otras.
- Capacidad de expandir su potencial utilizando módulos.
- Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Desventajas:

- Se necesita instalar un servidor web.
- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La programación orientada a objetos es aún muy deficiente para aplicaciones grandes.
- Dificulta la modularización.
- Dificulta la organización por capas de la aplicación.

2.1.4. Leguaje Jsp.

Es un lenguaje para la creación de sitios web dinámicos, acrónimo de Java Server Pages. Está orientado a desarrollar páginas web en Java. JSP es un lenguaje multiplataforma. Creado para ejecutarse del lado del servidor.

JSP fue desarrollado por Sun Microsystems. Comparte ventajas similares a las de ASP.NET, desarrollado para la creación de aplicaciones web potentes. Posee un motor de páginas basado en los servlets de Java. Para su funcionamiento se necesita tener instalado un servidor Tomcat.

Sintaxis:

Características:

- Código separado de la lógica del programa.
- Las páginas son compiladas en la primera petición.
- Permite separar la parte dinámica de la estática en las páginas web.
- Los archivos se encuentran con la extensión (jsp).
- El código JSP puede ser incrustado en código HTML.

Elementos de JSP

Los elementos que pueden ser insertados en las páginas JSP son los siguientes:

- **Código:** se puede incrustar código "Java".
- **Directivas:** permite controlar parámetros del servlet.
- **Acciones:** permite alterar el flujo normal de ejecución de una página.

Ventajas:

- Ejecución rápida del servlets.
- Crear páginas del lado del servidor.
- Multiplataforma.
- Código bien estructurado.
- Integridad con los módulos de Java.
- La parte dinámica está escrita en Java.
- Permite la utilización se servlets.

Desventajas:

- Complejidad de aprendizaje.

2.1.5. Lenguaje Python.

Es un lenguaje de programación creado en el año 1990 por Guido van Rossum, es el sucesor del lenguaje de programación ABC. Python es comparado habitualmente con Perl. Los usuarios lo consideran como un lenguaje más limpio para programar. Permite la creación de todo tipo de programas incluyendo los sitios web.

Su código no necesita ser compilado, por lo que se llama que el código es interpretado.

Es un lenguaje de programación multiparadigma, lo cual fuerza a que los programadores adopten por un estilo de programación particular:

- Programación orientada a objetos.
- Programación estructurada.
- Programación funcional.
- Programación orientada a aspectos.

Sintaxis:

Ejemplo de una clase en Phyton:

```
defdibujar_muneco(opcion):
ifopcion == 1:
C.create_line(580, 150, 580, 320, width=4, fill="blue")
C.create_oval(510, 150, 560, 200, width=2, fill='PeachPuff')
```

Ventajas:

- Libre y fuente abierta.
- Lenguaje de propósito general.
- Gran cantidad de funciones y librerías.
- Sencillo y rápido de programar.
- Multiplataforma.
- Licencia de código abierto (Opensource).
- Orientado a Objetos.
- Portable.

Desventajas:

- Lentitud por ser un lenguaje interpretado.

2.1.6. Lenguaje Ruby.

Ruby es un lenguaje dinámico para una programación orientada a objetos rápida y sencilla. Es un lenguaje interpretado de muy alto nivel y orientado a objetos. Desarrollado en el 1993 por el programador japonés Yukihiro "Matz" Matsumoto. Su sintaxis está inspirada en Phyton, Perl. Es distribuido bajo licencia de software libre (Opensource).

Sintaxis:

puts "hola"

Características:

- Existe diferencia entre mayúsculas y minúsculas.
- Múltiples expresiones por líneas, separadas por punto y coma ";".
- Dispone de manejo de excepciones.
- Ruby puede cargar librerías de extensiones dinámicamente si el (Sistema Operativo) lo permite.
- Portátil.

Ventajas:

- Permite desarrollar soluciones a bajo Costo.
- Software libre.
- Multiplataforma.

Los invitamos a conocer nuestras categorías sobre: Ajax, ASP, Bases de Datos, CSS, Javascript, Perl/CGI, PHP, RubyonRails, XHTML y XML para aprender más sobre los diferentes lenguajes de programación para la web.

2.2. Selección de herramientas para el desarrollo de la Aplicación.

Una vez escogidas las herramientas más opcionadas para la realización del trabajo de investigación se detalla un análisis en profundidad de estas dos tecnologías ampliamente aceptadas por la comunidad web, como son PHP y Java.

Se analizaran los aspectos importantes y distinguibles de ambas tecnologías y que se debe tener en cuenta a la hora de seleccionar uno u otro ante un nuevo proyecto, concretamente, un portal web. Se prestará especial atención a los puntos fuertes y débiles de cada lenguaje y cómo su rival se comporta ante eso.

PHP y Java son dos tecnologías que desde su lanzamiento siempre han venido precedidas de debates acerca de las ventajas y desventajas. Moviéndonos por el mundo de los desarrolladores nos damos cuenta que como en la mayoría de los temas, no existe una opinión general acerca de cuál es mejor. La conclusión final nunca es blanca o negra, sino que siempre cada una tendrá sus seguidores y detractores.

2.3. Comparación entre tecnologías Open Source.

Mediante la comparación realizada en la siguiente tabla presentamos los resultados obtenidos Php como la mejor opción. Debido a que nuestra investigación está ligada al costo de desarrollo y por el conocimiento previo que se tiene sobre la herramienta seleccionamos este Open Source para el desarrollo de la aplicación.

Tabla 3. Comparación entre tecnologías Open Source.

Comparación Tecnologías	PHP	JAVA
Modularización		✓
Mantenibilidad	✓	✓
Coste de desarrollo.	✓	
Integración externa.		✓
Seguridad		✓
Rendimiento	✓	
Escalabilidad	✓	✓

Java	Php
6 puntos	5 puntos
55%	45%

Tabla 4. Resultados de la comparación.

2.4. SISTEMAS GESTORES DE BASES DE DATOS.

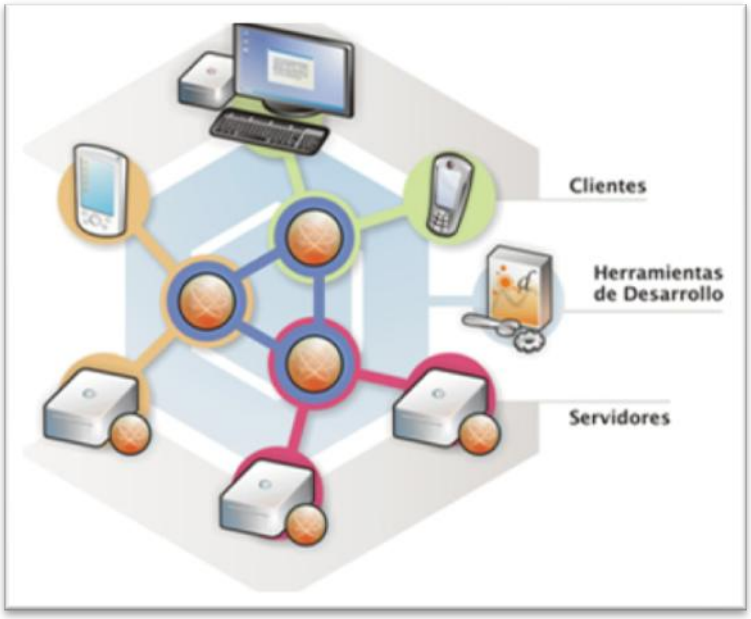


Figura 1. Sistemas Gestores de Bases de Datos.

La información, sea de la naturaleza que sea, y la posibilidad de obtener el máximo control sobre ella ha sido uno de los principales objetivos del ser humano desde hace ya siglos.

De hecho, el poder de gestionar grandes cantidades de datos se ha constituido, especialmente a través de las últimas décadas, en uno de los factores más significativos en lo que respecta al nivel de desarrollo del hombre.

Hoy en día es difícil encontrar un solo lugar en el mundo civilizado en el que no exista un completo control sobre todo lo que sea "registrable", ya sea en horarios de trenes, datos personales de la población, información estadística de cualquier índole o, incluso, datos antes tan difíciles de registrar de forma exhaustiva como son material literario u obras de arte de un museo.

Ya no existe casi nada que pueda escapar del control humano y, como tal, se debe acostumbrar a cohabitar por los sistemas encargados de realizar este tipo de gestión automática de la información, sin tener por ello que temer en ningún momento a las consecuencias derivadas de dicho control.

Los ordenadores han favorecido en gran medida la consumación de este objetivo, puesto que fueron ideados para encargarse de realizar todas aquellas operaciones que al hombre le suponían un gran esfuerzo y cantidad de tiempo.

Operaciones de carácter repetitivo en las que antes el ser humano debía emplear horas, como puede ser la ordenación de archivos o rellenar sobres con direcciones para remitir correspondencia a multitud de clientes, en nuestros días a un ordenador no le lleva más de unos segundos.

De hecho, conforme mejora la calidad y prestaciones de los equipos informáticos, mayor capacidad de cálculo son capaces de ofrecer éstos, por lo que, incluso la ordenación o búsqueda en ficheros constituidos por millones de fichas ya no representan un obstáculo para cualquier ordenador PC de la gama alta.

2.4.1. Sistema Gestor PostGreSQL.

2.4.1.1. Qué es PostGreSQL.

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

El siguiente gráfico ilustra de manera general los componentes más importantes en un sistema PostgreSQL.

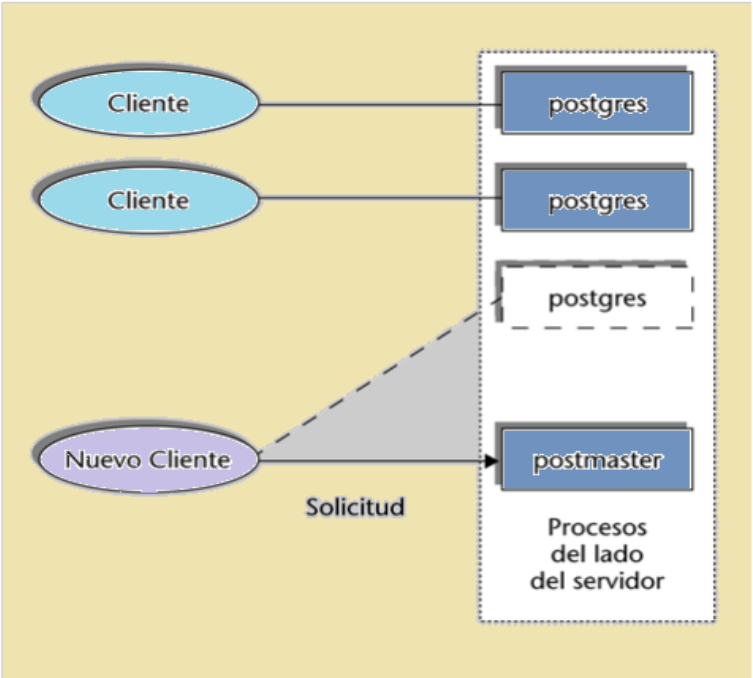


Figura 2. Arquitectura de Postgre.

- **Aplicación cliente:** Esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir víaTCP/IP ó sockets locales.
- **Demonio postmaster:** Este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes.

- **Ficheros de configuración:** Los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg_hba.conf y pg_ident.conf.
- **Procesos hijos postgres:** Procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes.
- **PostgreSQL share buffer cache:** Memoria compartida usada por POstgreSQL para almacenar datos en caché.
- **Write-Ahead Log (WAL):** Componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO).
- **Kernel disk buffer cache**: Caché de disco del sistema operativo.
- **Disco:** Disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

2.4.1.2. Características.

La última serie de producción es la 9.0. Sus características técnicas la hacen una de las bases de datos más potentes y robustos del mercado. Su desarrollo comenzó hace más de 15 años, y durante este tiempo, estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

Características más importantes y soportadas por PostgreSQL:

a) Generales.

- Es una base de datos 100% ACID
- Integridad referencial
- Tablespaces
- Nestedtransactions (savepoints)
- Replicaciónasincrona / Streaming replication - Hot Standby

- Two-phasecommit
- PITR - point in time recovery
- Copias de seguridad en caliente (Online/hotbackups)
- Unicode
- Juegos de caracteres internacionales
- Multi-VersionConcurrency Control (MVCC)
- Múltiples métodos de autentificación
- Acceso encriptado vía SSL
- Actualización in-situ integrada (pg_upgrade)
- Completa documentación
- Licencia BSD
- Disponible para Linux y UNIX en todas sus variantes (AIX, BSD, HP-UX, SGIIRIX, Mac OS X, Solaris, Tru64) y Windows 32/64bit.

b) Programación / Desarrollo.

- Funciones/procedimientos almacenados (storedprocedures) en numerosos lenguajes de programación, entre otros PL/pgSQL (similar al PL/SQL de oracle), PL/Perl, PL/Python y PL/Tcl
- Bloques anónimos de código de procedimientos (sentencias DO)
- Numerosos tipos de datos y posibilidad de definir nuevos tipos. Además de los tipos estándares en cualquier base de datos, tenemos disponibles, entre otros, tipos geométricos, de direcciones de red, de cadenas binarias, UUID, XML, matrices, etc.
- Soporta el almacenamiento de objetos binarios grandes (gráficos, videos, sonido)
- APIS para programar en C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, PHP, Lisp, Scheme, Qt y muchos otros.
- SQL
- Llaves primarias (primarykeys) y foráneas (foreignkeys)

- Check, Unique y Not null constraints
- Restricciones de unicidad postergables (deferrableconstraints)
- Columnas auto-incrementales
- Índices compuestos, únicos, parciales y funcionales en cualquiera de los métodos de almacenamiento disponibles, B-tree, R-tree, hash ó GiST
- Sub-selects
- Consultas recursivas
- Joins
- Vistas (views)
- Disparadores (triggers) comunes, por columna, condicionales.
- Reglas (Rules)
- Herencia de tablas (Inheritance)
- Eventos LISTEN/NOTIFY

2.4.1.3. Límites de Postgresql.

Tabla 5. Limitaciones de Postgre.

Límite	Valor
Máximo tamaño base de dato	Ilimitado (Depende de tu sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

2.5. Sistema Gestor MySQL.

2.5.1. Qué es MySQL.

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca.

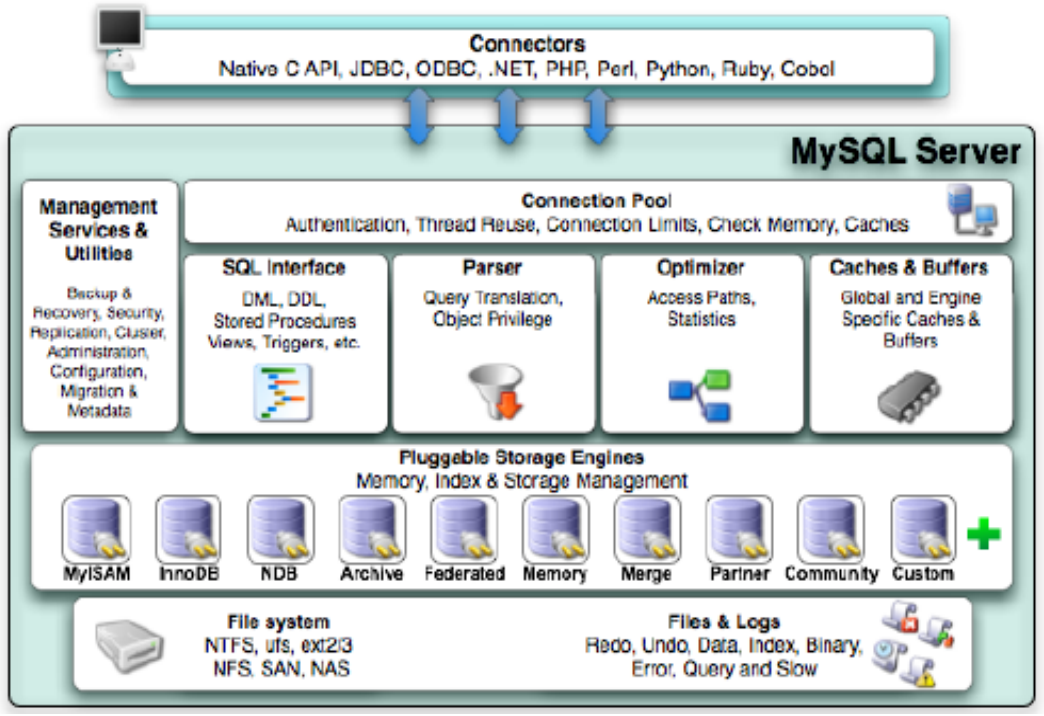


Figura 3. Arquitectura de MySql

Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

2.5.1.1. Características de MySQL.

Las principales características de este gestor de bases de datos son las siguientes:

- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc).
- Gran portabilidad entre sistemas.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.

2.6. Perspectiva histórica.

La comunidad de PostgreSQL es la más antigua, la comunidad más grande y de más rápido crecimiento de este tipo. PostgreSQL se inició en 1985 en la Universidad de California, Berkley como una evolución del proyecto Ingres. Después de varios años en el mundo académico, el proyecto PostgreSQL fue puesto en libertad en el mundo del código abierto. La comunidad comenzó a florecer hasta el punto que, hoy en día, hay más de 1.000 colaboradores y más de 30.000 miembros.

La principal ventaja de la gran comunidad es la interacción entre usuarios y desarrolladores, que permite a los usuarios a involucrarse directamente en el diseño

de nuevas características. Esta comunidad diversa es el modelo que muchas otras comunidades de código abierto.

Para evaluar adecuadamente las ventajas de PostgreSQL y MySQL, los dos principales bases de datos de código abierto, uno debe mirar a la historia y la genealogía de cada uno, y su funcionalidad de características y rendimiento. Al comparar y contrastar PostgreSQL y MySQL aquí, Centro de Conocimiento colaborador JimMlodgenski le guía a una decisión más informada acerca de que la base de datos de código abierto es adecuado para su implementación en la empresa.

PostgreSQL y MySQL: Cómo seleccionar el derecho de fuente abierta de bases de datos - Características de PostgreSQL y funcionalidad.

2.7. PostgreSQL y su funcionalidad.

PostgreSQL está considerado como el más avanzado de bases de datos de código abierto en el mercado-una reputación que ha construido a través de décadas de desarrollo. Como con todas las funciones y de código abierto DBMS relacional (RDBMS), PostgreSQL cuenta con muchas características diseñadas para admitir un gran transacción, aplicaciones de misión crítica.

La fuerza de la base de PostgreSQL es llevar a cabo de manera segura los datos que maneja. Esto comienza con el acceso, control de conexiones a la base de datos mediante el uso de mecanismos de autenticación de la empresa, tales como LDAP o Kerberos. Una vez autenticado, todas las comunicaciones con la base de datos puede ser más de una capa de sockets seguros (SSL) para entornos de alta seguridad.

Al añadir o modificar datos, PostgreSQL cumplir una serie de restricciones definidas por el usuario para garantizar la calidad de los datos de acuerdo a las reglas de negocio. Esto va desde las restricciones de comprobación sencilla de los controles clave más compleja extranjeros. Una vez que los datos se almacenan en el disco, la capacidad de copia de seguridad y, más importante aún, recuperar de un desastre es crucial. PostgreSQL tiene una facilidad simple copia de seguridad en línea que funciona en combinación con un fuerte punto en el tiempo de recuperación (pitr) mecanismo, proporcionando a los administradores la flexibilidad necesaria para recuperarse rápidamente de una pérdida.

Por ejemplo, la arquitectura básica de PostgreSQL permite que otros grupos comunitarios para desarrollar las funciones más avanzadas en PostgreSQL a través de módulos adicionales. Un ejemplo perfecto de esto es el apoyo geoespacial de PostgreSQL. Esta funcionalidad viene de un módulo llamado PostGIS, una simple extensión de PostgreSQL que sin duda hace que sea más fuerte la base de datos espacial y de código abierto o comercial.

Otra extensión de PostgreSQL es la posibilidad de tener diferentes tipos de lenguajes de procedimiento almacenado. Esto permite a los desarrolladores crear código en el servidor usando el mejor lenguaje para sus necesidades. Por ejemplo, un disparador que necesita para realizar el procesamiento de textos complejos que pueden ser escritos en Perl para poder utilizar su funcionalidad fuerte expresión regular.

2.8. MySQL y su funcionalidad.

MySQL tiene la reputación de ser el más popular de código abierto base de datos-una reputación derivada de su legado de rendimiento y simplicidad. Desde el comienzo, MySQL fue diseñado para ser un método rápido de acceso secuencial indizado (ISAM) almacenar los datos de sitios web. Este tipo de carga de trabajo, que se

caracteriza por ser una carga de lectura-sobre todo con muchas pequeñas consultas-ha dado lugar a características tales como una caché de consultas que mejora el rendimiento de MySQL aún más. Esta concentración en el rendimiento ha inspirado a las características tales como MySQLCluster, que permite que la base de datos a escala más allá de un único servidor físico.

PostgreSQL no es la única base de datos de código abierto que permite extensiones externas para aumentar la funcionalidad de la base de datos. Uno de los puntos más fuertes de MySQL es su motor de almacenamiento conectables. MyISAM, el motor de almacenamiento por defecto de MySQL, ofrece el rendimiento de lectura sobre todo los ambientes, y el motor de almacenamiento InnoDB proporciona la robustez necesaria para la operación de escribir aplicaciones de uso intensivo.

Además, hay una serie de motores de almacenamiento de terceros como Brighthouse y DB2 que añadir aún más funciones a MySQL. Esta flexibilidad permite a los administradores para sintonizar una instancia de MySQL basados en las necesidades de las tablas individuales. Por ejemplo, una mesa de lectura, en su mayoría como una tabla de códigos de país puede utilizar un motor de almacenamiento MyISAM, mientras que una tabla de transacciones tales como una tabla de órdenes de venta se puede utilizar InnoDB.

A pesar de PostgreSQL y MySQL tienen la reputación de apoyo a distintos tipos de aplicaciones, las bases de datos se utilizan con frecuencia en un amplio espectro de aplicaciones. Por ejemplo, PostgreSQL es conocida por su fortaleza para soportar las aplicaciones transaccionales de la empresa, pero también se utiliza para apoyar muchas aplicaciones Web. Por el contrario, MySQL, la base de datos tradicionalmente fuerte para las aplicaciones Web, también es utilizado por las aplicaciones que requieren soporte transaccional. La clave es que cualquiera de las

opciones tiene la flexibilidad para manejar un amplio rango de usos, pero uno puede ser una mejor opción técnica y de negocio basado en circunstancias individuales.

2.9. Decidir uno de ellos.

Independientemente de las diferencias técnicas que existen entre los diferentes gestores de bases de datos. La diferencia más notable entre PostgreSQL y MySQL la define la comunidad.

PostgreSQL es la más antigua e independiente de código abierto de la comunidad de base de datos de este tipo. El beneficio de una comunidad es la independencia del proveedor real.

Por otro lado, el segundo tipo de comunidad de código abierto puede tener un proveedor que controla el proyecto y el proyecto podría ser "comprado". MySQL es un ejemplo de ello. El proyecto fue financiado inicialmente MySQL y controlada por MySQL AB, una empresa comercial que emplea todos los principales desarrolladores y arquitectos de MySQL. La comunidad de MySQL más recientemente, ha sido controlado por Sun y Oracle ahora.

CAPITULO III

3. INDICADORES QUE INFLUYEN EN EL COSTO DE DESARROLLO DE APLICACIONES WEB.

3.1. Metodologías de Estimación del tamaño del software.

A continuación se presenta una descripción de cada una de las metodologías de estimación del tamaño, consideradas como las más importantes y más usadas en la actualidad. Existen básicamente dos aproximaciones a esta estimación: el conteo de líneas de código y el conteo de puntos de función. A continuación se describe dichas aproximaciones.

3.2. Estimación basada en líneas de código.

Esta estimación se podría catalogar como de tipo tardío, ya que el número total de líneas de código sólo se puede conocer cuando el producto esté terminado, aunque la tarea no es tan sencilla como contar la longitud de cada archivo; se debe acordar un formato, en donde se especifique qué es lo que se va a contar y qué no. Por ejemplo, los comentarios escritos en el código no deberían ser contados, por lo cual sólo se debe contar, lo que se especifique a ser contado.

Dentro de esta categoría existen varias metodologías las cuales usan las líneas de código como base para la realización de su estimación.

3.3. Estimación por conteo de bloques.

Este enfoque se basa en estimar el número de funciones esperadas que tendrá el sistema. Se puede ver como un enfoque de estimación temprana debido a que estima el número de funciones esperadas. Por tanto, se cuenta con poca información acerca del proyecto con lo que las estimaciones no podrían ser muy exactas. De esta manera, a medida que avanza el proyecto es deseable que las estimaciones fueran más coherentes con la realidad.

Es posible que el método pueda ser complementado con funciones estadísticas para encontrar una estimación más precisa. Con este fin es usada la desviación estándar, obtenida a partir de la información de proyectos pasados ya realizados, lo cual mejora en gran parte las estimaciones para la organización.

A continuación se enumeran los pasos empleados en el uso de este modelo:

- **a.** Estimar el número de bloques, o componentes de software esperados.
- **b.** Multiplicar el número de bloques por el tamaño esperado de cada tipo de bloque.
- **c.** Calcular la desviación estándar para dicho proyecto.
- **d.** Aplicar el método repetidamente para los diferentes niveles de detalle, y así obtener una estimación más precisa.

3.4. Estimación del tamaño estadística.

Este método se basa en la estimación del tamaño a partir de la utilización de cálculos estadísticos y dividiendo el sistema en componentes para cada uno de los componentes que integran el sistema posibilitando la estimación del sistema completo tomando como base cada uno de sus componentes por separado. Asimismo, este método se encarga de disminuir la incertidumbre acerca de las estimaciones de los

componentes individuales, lo cual posibilita contar con una estimación mucho más segura del sistema completo.

Con este fin, el método se basa en la estimación por analogía, en la cual se compara el proyecto actual con otros anteriores ya realizados, evidenciando la necesidad de mantener una base de datos con la información acerca de todos estos proyectos anteriores que servirán para la estimación del proyecto en curso.

A continuación se listan los pasos para estimar el tamaño del software con este método:

a. Determinar las funciones que compondrán el nuevo sistema.

b. Buscar información acerca del tamaño de funciones similares ya desarrolladas.

c. Identificar las diferencias entre las funciones similares y las nuevas.

d. Para cada componente o función a estimar, se deben estimar tres parámetros, el menor, el medio y el máximo tamaño de cada uno de los componentes o funciones.

e. Calcular la media estadística y desviación estándar de cada uno de los números obtenidos en el numeral anterior.

f. Tabular cada uno de estos datos obtenidos.

g. Calcular la media total del proyecto, y la desviación estándar del proyecto.

3.5. Estimación por lógica difusa.

Este método se basa en dividir el proyecto en categorías de tamaño. Dependiendo de la cantidad de líneas de código producidas en cada una clasificarlas en grande, mediano y pequeño. Para realizar esta categorización se requiere tener información de proyectos anteriores para generar los grupos antes descritos.

Por consiguiente para realizar la estimación del nuevo proyecto se debe juzgar en qué categoría quedaría éste, lo cual daría un rango de líneas de código que el nuevo proyecto podría producir.

Un problema que presenta este método, es que el cambio tecnológico trae como consecuencia que la magnitud en líneas de código de un proyecto varié, lo cual podría hacer que los grupos ya anteriormente definidos necesariamente tengan que cambiar.

3.6. Estimación basada en puntos de función.

Este método se diferencia a los basados en líneas de código en que, no se basa en la longitud de programa sino en la funcionalidad que presta, lo cual hace a este método independiente del lenguaje.

El Análisis de Punto Función es una técnica que, mediante la descomposición de un sistema en componentes más pequeños, permite que éstos puedan ser mejor comprendidos y analizados en forma individual.

El Análisis de Punto Función se basa en la teoría de que las funciones de una aplicación son la mejor medida del tamaño de un sistema. El Punto Función mide el software mediante la cuantificación de la funcionalidad que el sistema le brinda al usuario basado fundamentalmente en el diseño lógico. Es independiente del lenguaje de computación, de la metodología de desarrollo, de la tecnología utilizada y de la capacidad del equipo de trabajo para desarrollar la aplicación.

El Análisis del Punto Función es un método estándar de medición de desarrollo de software desde el punto de vista del usuario. Su objetivo es medir el software basándose en la cuantificación de la funcionalidad brindada al usuario partiendo fundamentalmente de diseños lógicos. La cuenta de Punto Función para proyectos de

desarrollo mide las funcionalidades que se le proporcionan al usuario conjuntamente con la primera instalación del software producido cuando el proyecto es terminado.

El método realiza su estimación contando el número de componentes de cada clase de punto funcional, luego se estima la complejidad de cada uno de los componentes medidos, alta o baja, según sea el caso, luego se multiplica cada contador de puntos de función por el valor adecuado, y se suma el total de puntos de función.

- ✓ Uso de técnicas y herramientas efectivas para el proceso.
- ✓ Obtención de un espacio físico y ambiente de trabajo óptimo.

3.7. Gestión de Costos.

La gestión de costos es una actividad necesaria para cualquier proyecto debido a que permite conocer qué tanto se va a gastar en su implementación y desarrollo, el destino de los diferentes recursos del proyecto a las actividades planeadas y el control de lo que se está invirtiendo; de esta actividad depende en gran parte que la terminación del proyecto genere ganancias o pérdidas.

3.8. Estimación del costo del proyecto.

Como es de suponerse, el costo de un proyecto se encuentra directamente ligado al tamaño del mismo, ya que el tamaño determina en la mayoría de los casos la duración y la dificultad de realizar dicho sistema. Partiendo de esto, el tamaño constituye uno de los factores que deben ser tenidos en cuenta al momento de realizar una buena estimación del costo de un proyecto. Sin embargo, existen otros tales como: el costo del personal y los recursos necesarios que son claves para el debido desarrollo de esta actividad.

Lo anterior nos deja una clara visión acerca de los múltiples aspectos que deben ser tenidos en cuenta al momento de realizar una estimación apropiada del costo de un

proyecto, así como el método a usar para dicha estimación. En general, existen dos tipos de métodos para la estimación del costo de un proyecto: los métodos algorítmicos y no algorítmicos. Los métodos no algorítmicos se basan por lo general en la experiencia dejada por proyectos anteriores.

3.9. Metodologías de estimación del costo de un proyecto de software.

En general existen dos tipos de métodos para la estimación del costo de un proyecto: los métodos no algorítmicos y algorítmicos A continuación se hace una breve explicación de los métodos más relevantes en esta área.

3.10. Métodos no algorítmicos.

Estos métodos están basados específicamente en las capacidades de juicio de las personas que realizan estas estimaciones, las personas se basan en su experiencia o experiencia de otros para obtener una estimación del proyecto a realizarle, los métodos que pertenecen a esta categoría muchas veces requieren de datos históricos para las estimaciones, lo que muchas veces es algo problemático ya que no todas las organizaciones mantienen información de sus proyectos anteriores. Estos pueden ser:

3.10.1. Costos por Analogía.

Requiere que al menos se tenga información de un proyecto anterior similar, se usan los datos de las anteriores estimaciones y sus resultados para lograr una estimación más precisa.

3.10.2. Juicio Experto.

Se requiere consultar a uno o más expertos en la estimación del tamaño de software, donde cada uno realiza una estimación diferente y luego se llega a un consenso sobre ésta. Los pasos que contiene este método son:

a. Se presenta a cada estimador, se realiza la estimación en la base de los compañeros.
b. Cada experto llena una forma con los resultados obtenidos.
c. El Coordinador prepara un resumen sobre cada una de as estimaciones.
d. Se Repiten los 2 últimos aspectos, hace lograr consenso entre los expertos.

3.10.3. Parkinson.

Se estima sobre el volumen de la producción del cliente, la cual se produce con los recursos que éste puede generar, se ajusta la propuesta para cumplir el presupuesto del cliente. Se obtiene una estimación global a partir de las características de todo el sistema, para luego realizar basado en esto, la estimación de cada parte del sistema.

3.10.4. Precio a Ganar.

Se fija el valor del proyecto para que sea el mejor de todos, sin tener en cuenta el tamaño, toma en cuenta el presupuesto del cliente.

3.10.5. Bottom UP.

Se estima cada uno de los componentes del sistema por separado, y luego se realiza una estimación total que comprende la sumatoria de cada una de las estimaciones pequeñas.

3.10.6. Top – down.

Este método es opuesto al anterior, para este método se realiza la estimación del total del costo para el proyecto, y desde este total se deriva el costo de cada uno de los componentes del software.

3.11. Métodos Algorítmicos.

Estos métodos se basan en la aplicación de una función a las propiedades del sistema para obtener una estimación formal de proyecto a implementar. Los métodos algorítmicos tienen en cuenta cinco factores relacionados con: costos, producto, herramientas computacionales, equipo humano, proyecto.

3.11.1. Modelos Lineales.

Los métodos algorítmicos se basan en la sumatoria de los aspectos que son relevantes al proyecto. Presenta como principal desventaja que la mayoría de veces el desarrollo de un proyecto en cuanto al precio no se comporta de forma lineal.

3.11.2. Modelos Multiplicativos.

Se multiplican los factores importantes del software que determinan el costo total del proyecto.

3.11.3. Modelos basados en funciones de potencia.

Relaciona el tamaño del software con otros factores de costo que influyen en el costo total del desarrollo del proyecto.

3.11.4. Cocomo.

Este modelo fue publicado por Barry Boehm y modificado posteriormente, es una proyección de las prácticas en la construcción de software de la época, evolucionando hasta darle un giro completo a la manera en la que el software era construido, lo cual hizo que el modelo original quedará obsoleto, y entonces se decidiera, reconstruir el modelo para adaptarlo a las nuevas prácticas y hacerlo de nuevo vigente.

Este modelo permite estimar el costo, el esfuerzo y el tiempo de duración de un proyecto de software, y fue creado para su utilización junto a los ciclos de vida modernos en los proyectos de software y sigue las necesidades de los usuarios de la estimación de costos en los proyectos de software, las cuales son apoyo en la planificación de proyectos, previsión del personal del proyecto, preparación del proyecto, replanificación y seguimiento del proyecto.

Para realizar todo esto, COCOMO II proporciona tres modelos de estimación cada vez más detallado, que tienen en cuenta cada sector y tipo de información necesaria en cada etapa del desarrollo de un proyecto de software. Cada uno de estos modelos ofrece mayor fidelidad en las estimaciones a medida que se avanza en la planificación y el diseño del mismo. Los modelos indicados son:

a. Modelo de composición de aplicaciones: Este modelo cubre los proyectos realizados con herramientas modernas de construcción de interfaces gráficas.

b. Modelo de Diseño Anticipado: Este modelo está diseñado para aplicarse en etapas iníciales del desarrollo en las cuales la arquitectura del mismo no haya sido totalmente definida.

c. Modelo de Postarquitectura: Este es el modelo más completo incluido en COCOMO II, y está diseñado para aplicarse luego que se ha terminado la etapa de diseño y luego de que la arquitectura del proyecto se encuentra bien planificada.

3.11.5. Slim.

Se basa en la distribución de poder hombre y en la experiencia y resultados obtenidos en proyectos anteriores. Este método utiliza una ecuación en donde se relaciona el tiempo de entrega y factores ambientales, en los cuales se refleja la capacidad de desarrollo de la compañía.

3.11.6. Modelos discretos.

Estos modelos son del tipo modular en donde se relaciona el esfuerzo, duración y dificultad y otros factores de costo, son fáciles de usar.

3.12. Planteamiento del modelo para la estimación del tamaño.

Esta sección contiene un análisis comparativo sobre las diversas metodologías para la estimación del tamaño del software con el fin de proponer las bases del modelo a desarrollar en el capítulo siguiente.

3.13. Evaluación de métodos para la estimación del tamaño del software.

Con el fin de proponer un modelo para la estimación del tamaño basado en las metodologías ya expuestas para ello.
Se presenta la siguiente tabla en donde se evalúan las virtudes y defectos de cada una permitiendo la escogencia adecuada del método que será utilizado en el modelo propuesto:

Tabla 6. Evaluación de los métodos para estimación del tamaño del software.

METODO	DESCRIPCIÓN	VENTAJA	DESVENTAJA
Conteo de Líneas de Código	Este método toma las líneas de código necesarias para la construcción de un sistema como medida de su tamaño.	-Se basa en el producto del desarrollo de software. -Fácil Conteo	-Dependiente del lenguaje de programación. - Dependiente de los programadores. - Estimación difícil en etapas tempranas del desarrollo.

Estimación basada en la estadística	Este método divide, el sistema en componentes, para así realizar estimaciones sobre cada componente por analogía con otros componentes ya realizados, luego obtienen una estimación pesimista, media y optimista.	-Disminuye la incertidumbre, dividiendo el sistema en componentes. -Se basa en un proceso estadístico, que ofrece un grado de seguridad. -El grado de confiabilidad de las estimaciones se hace mejor a medida que se realicen más estimaciones.	-Si no se cuenta con datos históricos, las estimaciones serán poco confiables. -El método requiere un tiempo para converger en buenas estimaciones.
Estimación Por Lógica Difusa	Este método se basa en estimación por analogía, ya que se toma información histórica del tamaño de diferentes proyectos, y se realizan categorías de tamaño en las que se encasillan los proyectos, luego el nuevo proyecto se encasilla en una de estas categorías, lo cual da un aproximado del tamaño del proyecto.	-Es un método sencillo de aplicar. -Da un rango del tamaño del software, lo que no se compromete del todo con un tamaño exacto	-Depende de la información histórica, de otros proyectos. - El proyecto estimado posiblemente no esté en el rango especificado, lo que podría resultar en una estimación muy alejada de la realidad. Requiere un tiempo de convergencia.
Estimación Por Puntos de Función	Se basa en la funcionalidad del sistema. Para realizar su estimación se deben determinar los componentes de puntos de función para el sistema y clasificarlos según su dificultad.	-Al depender de la funcionalidad del sistema, su aplicación se puede realizar desde la definición de los requerimientos del sistema. -Es Independiente del Lenguaje. -Fácil Utilización.	Es posible que no se encuentren todos los componentes necesarios, lo que daría una estimación equivocada. No es muy adecuado para cuando el software se encuentra construido.

3.14. Método escogido para la estimación del tamaño del software.

De acuerdo con los aspectos expuestos anteriormente y considerando el estado del arte de la estimación de proyectos de software, es posible afirmar que las metodologías son muy variadas y su uso, más que depender de lo que ofrecen, depende del entorno y la organización que quiera implementarlo, así como los procesos de la misma y el método de desarrollo de software que se utilice.

De igual manera se aprecia que las técnicas suelen ser muy sencillas, debido a que solo requieren de una persona para obtener la información del tamaño. Sin embargo, se dejan de lado muchos aspectos importantes que deben ser considerados en la estimación, otros métodos utilizan la funcionalidad del software, por ejemplo, para implantar una debida estimación.

Pensando en la funcionalidad del software y en la facilidad que los puntos de función pueden aportar a las actividades de estimación del tamaño, este último aspecto también importante porque atribuye la sencillez o simplicidad que una pequeña empresa de desarrollo necesita de este tipo de procesos, los puntos de función constituyen el método seleccionado para llevar a cabo la fase de estimación del modelo a proponer.

3.14.1. Por qué se escogió este método.

La característica principal por la que se escogió este método fue la flexibilidad que presenta para ser utilizado en etapas tempranas del desarrollo del software, en donde no es mucho lo que se sabe con respecto a las características del proyecto: esta metodología se basa en la funcionalidad del sistema a implementar y no en el producto a crear.

El método puede ser utilizado en diversas etapas del proyecto, a medida que aumente el conocimiento acerca del proyecto también aumentará la calidad de las estimaciones, haciéndolas cada vez más acercadas a la realidad. Otra característica destacable del análisis de puntos de función, es su independencia del lenguaje que se este usando, esto debido a que se basa en la funcionalidad, lo que hace que esta metodología sea ideal para cualquier tipo de sistema.

3.14.2. Esquema del método de Puntos de función.

Para estimar el tamaño de software por puntos de función, se deben encontrar algunos elementos como las salidas y entradas del sistema y bases de datos/archivos en donde se guarda la información. Posteriormente se debe encontrar la dificultad de cada componente. Finalmente mediante la aplicación de una serie de fórmulas sencillas se obtiene el número total de puntos de función que componen el software.

3.14.3. Evaluación de métodos y modelos para la estimación de costos.

En este campo se encontraron diversas metodologías para la estimación de costos del software a construir. A continuación se muestra una tabla de las fortalezas y debilidades de cada una las metodologías anteriormente descritas.

Tabla 7. Evaluación de los métodos para la estimación de costos

METODOLOGÍA	DESCRIPCIÓN	VENTAJAS	DESVENTAJAS
NO ALGORÍTMICOS			
Costos por Analogía	El costo del proyecto se estima en base al costo de proyectos similares ya realizados.	Si se cuenta con buena información histórica de proyectos pasados, se pueden obtener estimaciones bastante acertadas. Las estimaciones son sencillas de	Se requiere información histórica de proyectos para realizar la estimación. Para que el método sea efectivo se requiere ajustar el método con

		realizar	información de la organización que realiza el proyecto.
Precio a Ganar	Se ajusta el precio del proyecto, para mejorar la propuesta más económica realizada, con el fin de ganar el proyecto.	La estimación se realiza de una manera muy sencilla. Es muy probable que se gane el proyecto.	La estimación muy seguramente está mal, y el costo real estará muy alejado de la realidad. Puede ocasionar que el proyecto lleve a perdidas.
MÉTODOS ALGORÍTMICOS			
COCOMO	Modelo empírico para la estimación del esfuerzo y costo del desarrollo de un sistema de software, se basa en el uso de multiplicadores de esfuerzo, para realizar una estimación del esfuerzo y costo	Involucra varios factores que inciden en el costo del proyecto. No requiere en principio el uso de datos de proyectos anteriores. Permite su utilización a lo largo de todo el proyecto.	Predisposición por parte del equipo de gestión ante la utilización de fórmulas matemáticas. Es un proceso extenso para completar la estimación Algunos factores son algo complicados de determinar.
SLIM	Se basa en la distribución de poder hombre, se usa la ecuación de software, en donde se relaciona, el tiempo de entrega, factores ambientales, en los cuales se refleja la capacidad de desarrollo de la compañía	Usa factores del proyecto y producto, para realizar la estimación, estos factores inciden en el costo del proyecto.	Predisposición por parte del equipo de gestión ante la utilización de fórmulas matemáticas. Es un proceso algo largo, para completar la estimación

De acuerdo con la tabla 7 se evidencia un amplio rango de metodología para la estimación del costo, y cada uno con características diferentes, que los hacen aplicables en diferentes entornos.

Existen metodologías basadas en la experiencia de los gerentes de proyectos, algunas un poco menos elaboradas, lo que intentan es ganar un contrato en cambio de realizar una estimación seria.

De igual manera existen metodologías más complicadas que utilizan modelos empíricos y matemáticos para estimar el costo de un software, evaluando a su vez, una serie de factores concernientes al tamaño del software, a la organización, experiencia en esta clase de proyectos, etc.

3.15. Modelo escogido para la estimación de costos.

En el caso de la estimación de costos se escogió como metodología COCOMO II, aunque esta es un tanto complicada, debido a la utilización de varias fórmulas que estiman el costo de un proyecto.

3.15.1. Esquema del modelo COCOMO II

El modelo se divide en 3 submodelos dependiendo del conocimiento del proyecto, es decir si no se conoce mucho acerca del proyecto, para una estimación inicial se usaría el modelo de composición de aplicaciones , en una etapa más avanzada del proyecto, se podría utilizar el modelo de diseño anticipado, y en el momento que se encuentren diseñada la arquitectura del proyecto, se podría utilizar el modelo de postarquitectura, estos modelos aumentan en complejidad a medida que se avanza las diferentes fases del proyecto, es decir el modelo de composición de aplicaciones es mucho más sencillo que el de diseño anticipado y postarquitectura, de igual manera las calidad de las estimaciones aumenta cuando se usan métodos más complicados.

Para este modelo se usará el modelo de diseño anticipado, ya que es un modelo adecuado para realizar estimaciones en las que se tienen en cuenta varios parámetros que inciden en gran parte en el costo de un proyecto, y a su vez disminuye la dificultad para realizar estas estimaciones, adicionalmente se desarrollarán plantillas para la realización estas estimaciones, con lo que se disminuirá en gran medida la dificultad en la aplicación del método.

3.15.2. Modelo para la estimación y gestión del proyecto.

A continuación se expone cada uno de los pasos planteados por el modelo para llevar a cabo los procesos de estimación y gestión de costos y riesgos, basándose para ello en las metodologías, herramientas y métodos, seleccionados en la propuesta conceptual, tras la evaluación de sus ventajas y desventajas y teniendo en cuenta los criterios de clasificación especificados.

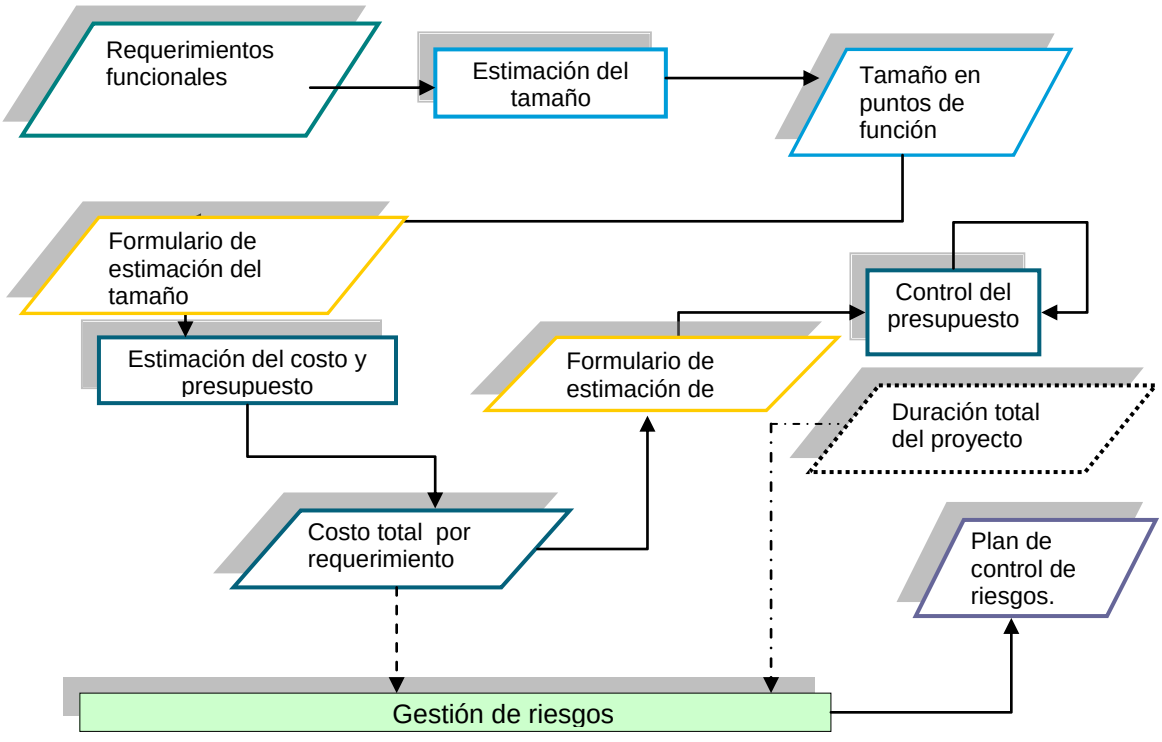


Figura 4. Pasos del modelo propuesto

En la figura 4, los primeros tres pasos (Requerimientos funcionales, estimación de tamaño y tamaño en puntos de función) son secuenciales y cada uno se explica con más detalle a continuación. La gestión de riesgos puede iniciarse de manera paralela en los tres primeros pasos.

Sin embargo, es importante destacar, que utiliza los datos obtenidos de la estimación de costos para llevar a cabo un análisis cuantitativo de riesgo en costo. De igual manera, puede utilizar algunas métricas del proceso de gestión del presupuesto para supervisar los planes de riesgo.

Con el fin de atribuir una mayor simplicidad al modelo, la metodología de gestión de riesgos propuesta se implementará como un paso posterior a la de gestión de costos.

3.15.3. Definir los Requerimientos Funcionales del Proyecto.

Las actividades que comprende la definición de los requerimientos funcionales para el modelo se explican a continuación de manera formal, especificando sus entradas, salidas y el proceso requerido.

3.15.4. Proceso de definición de requerimientos.

Este proceso comprende desde el conocimiento de los requerimientos funcionales del proyecto hasta su especificación.

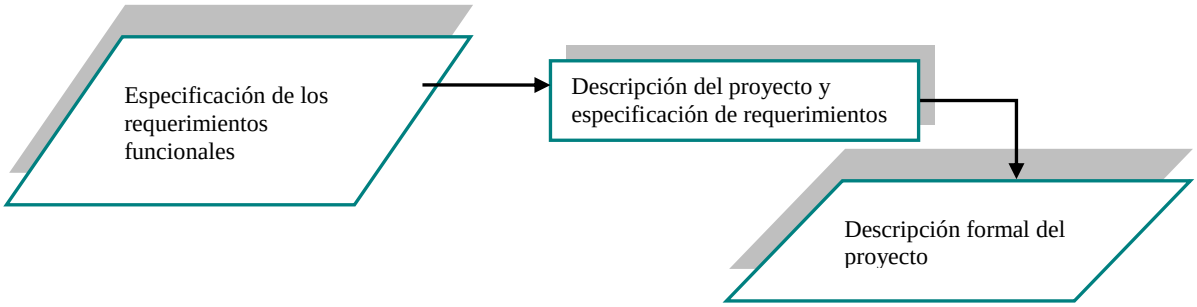


Figura 5. Proceso para la definición de los requerimientos

Tabla 8. Elementos del proceso para la definición de requerimientos

ENTRADAS	SALIDAS	PROCESOS	DESCRIPCIÓN
- Descripción del proyecto. - Especificación de los requerimientos funcionales.	Descripción del proyecto y de sus requerimientos funcionales.	Descripción formal del proyecto.	La especificación de los requerimientos.

3.16. Descripción del proyecto y especificación de los requerimientos.

Es importante contar con una descripción detallada del proyecto y las funcionalidades que debe implementar. Para ello es necesario que el proyecto sea descrito de la manera más clara y completa posible y que se especifiquen los requerimientos que deben ser implementados.

3.16.1. Estimar el Tamaño del Software.

Las actividades que comprende el paso para la estimación del tamaño se explican a continuación de manera formal, especificando sus entradas, salidas y el proceso requerido.

3.16.2. Modelo para la estimación del tamaño.

La figura 6 muestra las entradas y salidas que involucra el proceso para la estimación del tamaño del software.

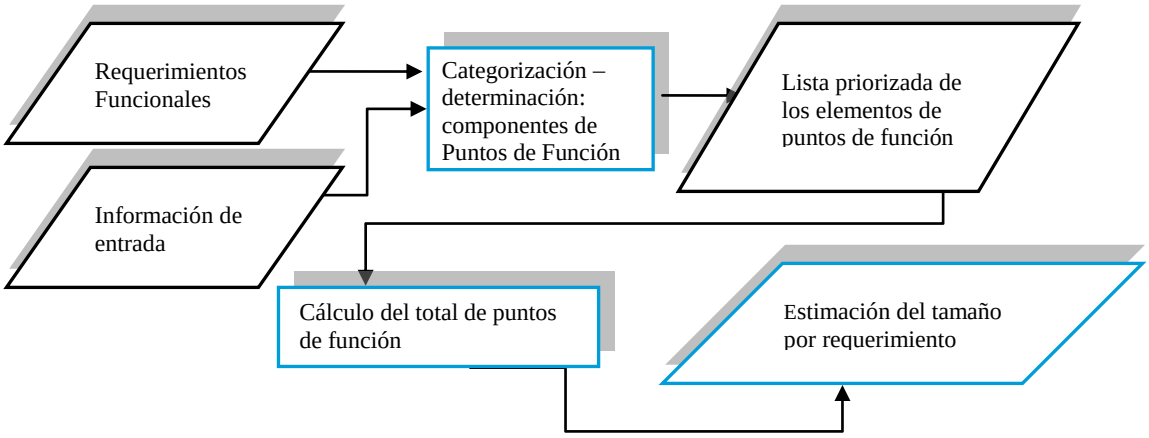


Figura 6. Esquema del Modelo de estimación del Tamaño

A continuación se explica de manera más formal las entradas, salidas y proceso para la estimación del tamaño del software.

Para realizar estas estimaciones se utilizará la metodología del análisis de puntos de función, y mediante ésta se medirá el tamaño del software a implementar a través de los componentes funcionales que deba manejar el sistema en desarrollo.

3.17. Proceso para la estimación del tamaño.

Ahora que ya se han mencionado las entradas y salidas del modelo para la estimación del tamaño, se procederá a describir el proceso con el cual se estimará el tamaño de cada uno de los requerimientos del proyecto.

3.18. Identificar componentes funcionales.

La metodología de puntos de función requiere que se identifiquen una serie de componentes funcionales los cuales proporcionan información acerca de la complejidad de un sistema de software: para este caso la complejidad de implementar un requerimiento, a continuación se explican en más detalle estos componentes.

3.18.1. Archivos Lógicos Internos (ILF)

Estos componentes representan información relacionada lógicamente o reconocida por el usuario que es manejada por la aplicación, es decir un repositorio de datos manejado por la aplicación.

3.18.2. Archivos de Interfaz Externos (ELF).

Este componente representa un grupo de datos relacionados lógicamente o información de control reconocida por el usuario y referenciada pero mantenida por otra aplicación, es decir repositorios de datos manejados por otra aplicación.

3.18.3. Entradas Externas (EI).

Este componente representa un proceso elemental o una acción que procesa datos o información de control que vienen desde afuera de la aplicación hacia adentro, la información pude venir desde una pantalla u otro sistema.

3.18.4. Salidas Externas (EO).

El componente representa un proceso elemental o una acción que envía datos o información de control hacia fuera de la aplicación, en este procedo debe estar involucrado algún tipo de procesamiento lógico sobre los datos de salida, estos elementos pueden ser consultas a bases de datos.

3.18.5. Consultas Externas (EQ).

Este componente es un proceso elemental o una acción que envía datos o información de control hacia afuera la aplicación, el propósito de esta operación es presentar datos tal cual se encuentran en la aplicación, sin ningún procesamiento lógico.

Los componentes identificados por cada uno de los requerimientos son los que deben ser implementados en el proyecto. Dicha identificación debe ser diligenciada en el Formulario para la estimación del tamaño con Puntos de Función, este formulario cuenta con una tabla por cada uno de los componentes de puntos de función anteriormente mencionados, en cada tabla se deben nombrar los componentes identificados de cada tipo.

En estas tablas se relacionan los requerimientos que deben ser implementados, esto con el fin de identificar cada uno de los componentes de puntos de función por requerimiento. Posteriormente en cada una de las tablas se señala los componentes que se relacionan con cada uno de estos.

Luego de relacionar cada componente de puntos de función con uno o más requerimientos se procede a clasificar la dificultad de implementar cada componente.

Esta tarea se refiere a encontrar la dificultad para implementar cada uno de los componentes identificados de puntos de función. La determinación de esta dificultad permite estimar el número de puntos de función para implementar en cada componente.

La dificultad de implementar cada uno de los componentes de puntos de función se determina, encontrando el número de elementos de datos que se maneja en cada uno de estos componentes, los tipos de elementos de datos varían dependiendo del tipo al cual corresponda éste.

3.18.6. Archivos Lógicos Internos y Archivos de Interfaz Externos.

Para estimar la complejidad de estos componentes se deben identificar dos elementos.

- Tipos de elementos de datos (DET): como su nombre lo indica son tipos de datos presentes en el repositorio, un ejemplo de tipos de datos puede ser una columna de una tabla en una base de datos.

- Conjunto de elementos de datos (RET): este elemento representa conjunto de elementos de datos relacionados lógicamente que se encuentran en el repositorio de información, un ejemplo de esto sería el conjunto de DET que representen una dirección.

3.18.7. Entradas Externas, Salidas Externas y Consultas Externas

En el caso de estos componentes se deben identificar los siguientes elementos para determinar la complejidad para implementarlos.

- Tipos de elementos de datos (DET): al igual que para los anteriores componentes este elemento representa los tipo de datos presentes en el componente, para el caso de los elementos del tipo transaccional como los EI, EO, EQ este elemento representa datos que entran o salen de la aplicación, como ejemplo de estos elementos se tiene un campo de entrada de texto presente en una interfaz gráfica.

- Tipo de archivo referenciado (FTR): este elemento representa los repositorios de datos que están involucrados en el desarrollo de la transacción del componente, estos repositorios son lo que se identificaron anteriormente, como un ejemplo se puede tomar una transacción que obtiene datos de una tabla perteneciente a una base de datos, esta tabla sería el FTR a identificar.

Luego de que explicar la manera en que se categorizan cada uno de los elementos de puntos de función, se prosigue a diligenciar el número de elementos identificado para cada componente de puntos de función en el Formulario para la estimación del tamaño con Puntos de Función, esto se realiza diligenciando el número de elementos de clasificación para cada componente luego de la sección de requerimientos.

Los elementos de puntos de función son clasificados dependiendo del número de elementos identificados para cada uno en 3 categorías Alto, Medio, Bajo, para obtener las categorías a partir del número de elementos se deben seguir las siguientes tablas.

3.18.7. Archivos Lógicos Internos y Archivos de Interfaz Externos

Para identificar la complejidad de estos componentes se sigue la tabla 9:

Tabla 9. Determinación de la dificultad de implementación para ILF y ELF.

	1 a19 DETs	20 a50 DETs	51 o más DETs
1RET	Baja	Baja	Media
2 a 5 RETs	Baja	Media	Alta
6 o más RETs	Media	Alta	Alta

3.18.8. Entradas Externas.

En el caso de las Entradas Externas se usa la tabla 10 para determinar la complejidad de implementación de estos componentes.

Tabla 10. Determinación de la dificultad de implementación para EI.

	1a 4 DETs	5 a 15 DETs	16 o más DETs
1 FTR	Baja	Baja	Media
2 FTRs	Baja	Media	Alta
3 o más FTRs	Media	Alta	Alta

Tabla 11. Valores numéricos.

Clasificación	Valores		
	Salidas Externas	**Consultas Externas**	**Entradas Externas**
Baja	4	3	3
Media	5	4	4
Alta	7	6	6

3.18.9. Salidas Externas y Consultas Externas.

Para determinar la dificultad de implementación se utiliza la tabla 11.

Tabla 12. Salidas externas y consultas externas.

	1 a 5 DETs	6 a 19 DETs	20 o más DETs
1 FTR	Baja	Baja	Media
2 a 3 FTRs	Baja	Media	Alta
4 o más FTRs	Media	Alta	Alta

Tabla 13. Valores Numéricos para salidas y consultas.

Clasificación	Valores		
	Salidas Externas	**Consultas Externas**	**Entradas Externas**
Baja	4	3	3
Media	5	4	4
Alta	7	6	6

3.19. Estimación de Líneas de Código (LDC) y Puntos de Función (PF).

Las LDC y los PF se describen como medidas básicas desde donde se calculan métricas de productividad. Los datos de LDC y PF se utilizan de dos formas durante la estimación del proyecto de software:

- Como una variable de estimación que se utiliza para "dimensionar" cada elemento del software.
- Como métricas de línea base recopiladas de proyectos anteriores y utilizados junto con variables de estimación para desarrollar proyecciones de costo y de esfuerzo.

En un comienzo el proyecto se disgrega en pequeñas subfunciones que pueden ser estimadas individualmente, ya sea en LDC o PF para cada función. Cuando se utiliza LDC como variable de estimación, la descomposición funcional es absolutamente necesaria.

También debe tenerse en cuenta que mientras las LDC se estiman directamente, los PF se determinan indirectamente mediante la estimación de número de entradas, salidas, archivos de datos, consultas e interfaces, así como también de catorce valores de ajuste de complejidad.

Independientemente de la variable de estimación que use el planificador del proyecto, normalmente proporciona un rango de valores para cada función descompuesta.
¿Serán correctas las estimaciones? La única respuesta razonable a esta pregunta es "No podemos asegurarlo". Cualquier técnica de estimación, no importa como sea de sofisticada, tiene que ser comprobada utilizando otro método. Incluso entonces, deberán prevalecer la experiencia y el sentido común.

¿Qué ocurre cuando la concordancia entre las estimaciones es pobre? Para responder a esta pregunta se debe reevaluar la información que se ha utilizado para hacer las estimaciones. Muchas divergencias entre estimaciones se deben a menudo, a una de dos causas:

- No se entiende adecuadamente el ámbito del proyecto o ha sido malinterpretado por el planificador.
- Los datos de productividad utilizados en la técnica LDC, son inadecuados para esa aplicación, están obsoletos (no reflejan con precisión la organización de desarrollo de software) o se han aplicado mal.

3.20. Conversión de Puntos Función a Líneas de Código Fuente (SLOC).

Para determinar el esfuerzo nominal en el modelo COCOMO II los puntos función no ajustados tienen que ser convertidos a líneas de código fuente considerando el lenguaje de implementación (assembler, lenguajes de alto nivel, lenguajes de cuarta generación, etc.). Esto se realiza para los modelos Diseño Temprano y Post Arquitectura teniendo en cuenta la siguiente tabla.

Tabla 14. Conversión de UFP a SLOC. [COCOMO II.0].

Lenguaje	QSMSLOC / Datos FP			
	Media	Mediana	Bajo	Alto
ABAP (SAP)	18	18	16	20
Acceso *	36	38	15	47
Ada	154	-	104	205
Ventaja	38	38	38	38
APS	86	83	20	184
ASP *	56	50	32	106
Ensamblador *	209	203	91	320
C *	148	107	22	704

C + + *	59	53	29	178
C # *	58	59	51	66
Clipper *	40	39	26	53
COBOL *	80	78	8	400
ColdFusion	68	56	52	105
Cool: Gen / IEF *	37	35	10	180
Datastage	67	79	16	85
DBase IV	52	-	-	-
Easytrieve +	33	34	25	41
Enfoque *	45	45	40	49
FORTRAN	90	118	35	-
FoxPro *	36	35	34	38
HTML	43	42	35	53
J2EE *	57	50	50	67
Java *	55	53	17	214
Php	56	52	12	213
JavaScript *	54	55	45	63
JCL *	96	59	58	173
JSP	59	-	-	-
KML	50	50	49	50
Lotus Notes *	23	21	15	46
Maestro	30	30	30	30
Mantis	71	27	22	250
Mapper *	69	70	58	81
* Natural	51	53	34	60
. NET	60	60	60	60
Openroad	39	34	20	69
Oracle *	42	29	12	217
Oracle Dev. 2K *	35	30	23	100
Pacbase *	42	43	26	52
PeopleSoft *	37	32	34	40
Perl	57	57	45	60
PL / 1 *	58	57	27	92
PL / SQL *	47	39	16	78

Powerbuilder **	28	22	8	105
Central eléctrica	63		25	79
REXX	50	-	-	-
SAS *	50	35	32	102
Siebel Herramientas	13	13	5	20
Slogan *	81	80	66	100
Smalltalk **	28	19	17	55
SQL *	31	30	13	80
SQL formas	11	11	10	15
Taskmate	45	47	37	51
Uniface	61	50	31	120
VB.Net	28	-	-	-
VBScript *	38	37	29	50
Visual Basic *	50	52	14	276
VPF	95	95	92	98
Secuencias de comandos Web	44	15	9	114

3.21. Descripción de entidades para el cálculo de puntos de función.

Para la obtención de los puntos de función se muestra cada una de las tablas con sus respectivos campos, para lo cual las claves principales se representaran con letra negrita, subrayado y las claves foráneas mediante el símbolo (fk) con negrita.

3.21.1. Tabla Usuario.

A continuación se presenta la tabla usuario, la cual contiene atributos, tipos y tamaños de datos la misma que posee como clave principal cedulausuario.

Tabla 15. Usuarios

USUARIO		
Nombre	**Tipo**	**Tamaño**
cedulausuario	Varchar	10

nombreusuario	Varchar	50
teléfono	Varchar	10
drección	Varchar	50
mail	Varchar	100
clave	Varchar	35

3.21.2. Tabla Cliente.

A continuación se presenta la tabla Cliente, la cual contiene atributos, tipos y tamaños de datos la misma que posee como clave principal cedulacliente.

Tabla 16. Clientes

CLIENTES		
Nombre	**Tipo**	**Tamaño**
cedulacliente	Varchar	13
nombrecliente	Varchar	100
teléfono	Varchar	10
dirección	Varchar	50
mail	Varchar	100

3.21.3. Tabla Cuenta.

A continuación se presenta la tabla Cuenta, la cual contiene atributos, tipos y tamaños de datos la misma que posee como clave principal idcuenta.

Tabla 17. Cuenta.

CUENTA		
Nombre	**Tipo**	**Tamaño**
idCuenta	Int	11
nombrecuenta	varchar	50
descripcionC	varchar	100

3.21.4. Tabla Factura.

A continuación se presenta la tabla Factura, la cual contiene atributos, tipo y tamaño de datos como clave principal Idfactura, posee llaves foráneas de la tabla cliente y usuario.

Tabla 18. Factura.

FACTURA		
Nombre	**Tipo**	**Tamaño**
idfactura	Int	11
formapago	varchar	1
Total	Float	
cliente(fk)	Varchar	13
usuario(fk)	Varchar	10
fecha	datetime	

3.21.5. Tabla Subcuenta.

A continuación se presenta la tabla Subcuenta, la cual contiene atributos, tipo y tamaño de datos como clave principal Idsubcuenta, posee llaves foráneas de la tabla factura y cuenta.

Tabla 19. Subcuenta

SUBCUENTA		
Nombre	**Tipo**	**Tamaño**
idsubcuenta	Int	11
cantidad	Float	
formapago	varchar	1
estado	varchar	1
factura(fk)	int	11

cuenta(fk)	Int	11
Fechacobro	Date	
Fechaactual	date	

3.21.6. Tabla Cuenta Banco.

A continuación se presenta la tabla Cuenta Banco, la cual contiene atributos, tipo y tamaño de datos como clave principal mumerocuenta.

Tabla 20. Cuenta Banco

CUENTA BANCO		
Nombre	**Tipo**	**Tamaño**
numerocuenta	Varchar	20
Nombrebanco	Varchar	50
Principal	Varchar	50

3.21.7. Tabla Cheque.

La tabla Cheque, la cual contiene atributos, tipo y tamaño de datos como clave principal Idcheque, posee llaves foráneas de la tabla factura y cuentabanco.

Tabla 21. Cheque

CHEQUE		
Nombre	**Tipo**	**Tamaño**
idcheque	Int	11
factura(fk)	Int	11
cuentabanco(fk)	varchar	20
cantidad	Float	
fechacobro	Date	

3.21.8. Tabla Producto.

A continuación se presenta la tabla Producto, la cual contiene atributos, tipos y tamaños de datos la misma que posee como clave principal Idproducto.

Tabla 22. Producto.

PRODUCTO		
Nombre	**Tipo**	**Tamaño**
idproducto	Int	11
Nombre	varchar	50
costo	Float	
pvp	Float	
descripcion	Varchar	50
stock	int	

3.21.9. Tabla Venta.

A continuación se presenta la tabla venta, la cual contiene atributos, tipo y tamaño de datos, posee llaves foráneas de la tabla factura y producto.

Tabla 23. Venta.

VENTA		
Nombre	**Tipo**	**Tamaño**
factura(fk)	Int	11
producto(fk)	Int	11
cantidad	Float	
pvp	Float	
Subtotal	Float	

3.21.10. Tabla Scc.

A continuación se presenta la tabla Scc, la cual contiene atributos, tipo y tamaño de datos, posee llaves foráneas de la tabla cheque y subcuenta.

Tabla 24. Subcuenta por cobrar.

SCC		
Nombre	**Tipo**	**Tamaño**
cheque(fk)	Int	11
subcuenta(fk)	Int	11
descripción	Varchar	50

3.22. Registro de datos.

Detalle de registro de datos de entrada con su respectiva dificultad, atributos y ficheros.

Tabla 25. Registro de datos

ENTRADAS			
Datos	**Dificultad**	**Atributos**	**Ficheros**
Usuario	Baja	6	1
Cuenta	Baja	3	2
Cliente	Baja	5	1
Factura	Alta	9	3
Subcuenta	Alta	8	3
Cuenta banco	Baja	3	2
Cheque	Alta	5	3
Producto	Media	6	2
Venta	Alta	5	3
Scc	Media	3	3

Detalle de registro de datos de salida con su respectiva dificultad, atributos y ficheros.

Tabla 26. Registro de las Salidas.

SALIDAS			
Datos	**Dificultad**	**Atributos**	**Ficheros**
Usuario	Baja	6	1
Cuenta	Baja	3	2
Cliente	Baja	5	1
Factura	Alta	9	3
Subcuenta	Alta	8	3
Cuenta banco	Baja	3	2
Cheque	Media	5	3
Producto	Madia	6	2
Venta	Media	5	3
Scc	Media	3	3

Detalle de registro de datos de las consultas con su respectiva dificultad, atributos y ficheros.

Tabla 27. Registro de las consultas.

CONSULTAS			
Datos	**Dificultad**	**Atributos**	**Ficheros**
Factura	Alta	9	3
Subcuenta	Alta	8	3
Cuenta banco	Baja	3	2
Cheque	Media	5	3
Producto	Madia	6	2
Venta	Media	5	3
Scc	Media	3	3

3.23. Calcular el total de puntos de función.

Para esta labor se deben tomar los componentes identificados y categorizados en los numerales anteriores y calcular el total de puntos de función para cada requerimiento: este cálculo del total de puntos de función se realiza mediante una ecuación en la cual se le da un peso específico y a cada valor en los que se categorizó cada componente (Alto, Medio, Bajo).

Para cada elemento se debe multiplicar el peso del componente por el número de elementos de este tipo y luego sumar este resultado con el valor dado para cada uno de los componentes, y el resultado de estas operaciones es el total de puntos de función.

Tabla 28. Calculo de puntos de función.

Descripción	Baja	Media	Alta	Total
Entradas	4*3	2*4	4*6	44
Salidas	4*4	4*5	2*7	50
Consultas	1*3	4*4	2*6	31
Archivos Interface de programa	3*7	0*10	0*15	21
Total				**146**

3.24. Modelo para la gestión de costos.

Este modelo de gestión de costos ofrece una herramienta sencilla que permite a sus usuarios realizar este proceso sin la necesidad de usar recursos costosos que le consuman mucho tiempo.

Por otro lado, las actividades de estimación de costos y de presupuesto se realizarán simultáneamente debido a que en esta primera se incluye una división del trabajo por requerimientos y por tanto la estimación total, involucrándolos a todos, dará como resultado el presupuesto total para todo el proyecto.

Otra razón importante para unir estos dos pasos consiste en las propias características del método a utilizar: COCOMO II. Como entrada este método requiere para calcular el costo de un producto de software, conocer el costo para la organización de una persona/mes, en dicha entrada deben venir especificados todos los factores contemplados en el momento de realizar un presupuesto.

3.25. Estimación de costos utilizando COCOMO II

La metodología escogida para realizar la estimación del costo del proyecto fue COCOMO II, específicamente el modelo de diseño anticipado, debido a que es un modelo que ofrece buena seguridad en la estimación sin mostrar toda la complejidad del modelo de postarquitectura.

3.25.1. Definición del modelo.

Los objetivos principales que se tuvieron en cuenta para construir el modelo COCOMO II fueron:

- Desarrollar un modelo de estimación de costo y cronograma de proyectos de software que se adaptará tanto a las prácticas de desarrollo de la década del 90 como a las futuras.
- Construir una base de datos de proyectos de software que permitiera la calibración continua del modelo, y así incrementar la precisión en la estimación.
- Implementar una herramienta de software que soportara el modelo.

- Proveer un marco analítico cuantitativo y un conjunto de herramientas y técnicas que evaluaran el impacto de las mejoras tecnológicas de software sobre los costos y tiempos en las diferentes etapas del ciclo de vida de desarrollo.

3.26. Modelo de Estimación de Costos Cocomo II.

Una de las tareas de mayor importancia en la administración de proyectos de software es la estimación de costos. Si bien es una de las primeras actividades, inmediatamente posterior al establecimiento de los requerimientos, se ejecuta regularmente a medida que el proyecto progresa con el fin de ajustar la precisión en la estimación. La estimación de costos de software tiene dos usos en la administración de proyectos:

Durante la etapa de planeamiento: Permite decidir cuantas personas son necesarias para llevar a cabo el proyecto y establecer el cronograma adecuado.

Para controlar el progreso del proyecto: Es esencial evaluar si el proyecto está evolucionando de acuerdo al cronograma y tomar las acciones correctivas si fuera necesario. Para esto se requiere contar con métricas que permitan medir el nivel de cumplimiento del desarrollo del software.

En el ámbito de la ingeniería de software, la estimación de costos radica básicamente en estimar la cantidad de personas necesarias para desarrollar el producto. A diferencia de otras disciplinas de la ingeniería, en las cuales, el costo de los materiales es el principal componente a ser estimado.

La estimación de costos de software posibilita relacionar conceptos generales y técnicas del análisis económico en el mundo particular de la ingeniería de software. Aunque no es una ciencia exacta no se puede prescindir de ella puesto que hoy en día un error en las predicciones puede conducir a resultados adversos.

Es importante reconocer la fuerte relación entre costo, cronograma y calidad. Estos tres aspectos están íntimamente relacionados y confrontados entre sí. De esta manera, se hace difícil incrementar la calidad sin aumentar el costo y/o el cronograma del software a desarrollar.
Similarmente, el cronograma de desarrollo no puede reducirse dramáticamente sin deteriorar la calidad del producto de software y/o incrementar el costo de desarrollo. Los modelos de estimación juegan un papel importante ya que permiten equilibrar estos tres factores.

Se han propuesto numerosos métodos de estimación. Entre ellos se pueden contar:

Juicio de Expertos: Este método implica la consulta a expertos, quienes usan su experiencia y conocimiento del proyecto propuesto para lograr una estimación de sus costos.

Analogía: Este método implica una estimación por analogía con proyectos similares, que ya han finalizado, de manera de relacionar los costos reales con la estimación del costo del nuevo proyecto. La principal virtud de la estimación por analogía es que está basada en la experiencia real de un proyecto. Esta experiencia puede ser estudiada para determinar las diferencias específicas con un proyecto nuevo y el impacto de los cambios en los costos. Por otra parte, la principal desventaja es que no está claro hasta que punto es realmente representativo el proyecto previo, en lo que se refiere a restricciones, técnicas, personal y funcionalidad requerida.

Parkinson: Este método intenta adaptar la estimación del costo a los recursos disponibles.
En general, es extremadamente inadecuado.

Tasar para ganar: Estima los costos en función del presupuesto adecuado para ganar el trabajo, o el cronograma necesario para estar primero en el mercado con el nuevo producto.

Estimación top-down: A partir de las propiedades globales del producto de software se deriva el costo de todo el proyecto. Después, el costo total es dividido entre las diversas componentes**.**

Estimación bottom-up: El costo de cada componente de software es estimado por separado, generalmente por la persona responsable del desarrollo de la misma, y luego sumados para obtener el costo total del proyecto. Las técnicas de estimación bottom-up y top-down pueden ser usadas en conjunción con cualquiera de los métodos discutidos en esta sección.

Modelos Algorítmicos: Estos métodos proveen uno o más algoritmos que estiman el costo del software en función de un número de variables que se consideran los principales factores de costo. Los valores de los factores se establecen a partir del análisis de regresión de datos confiables recopilados en proyectos anteriores. Comparados con otros métodos una de sus ventajas es la objetividad, ya que están calibrados a partir de experiencias anteriores. Esto mismo constituye la principal desventaja, por no poder asegurar que estas experiencias sean realmente representativas de proyectos futuros, en especial si se desarrollan en nuevas áreas de aplicación, con nuevas técnicas y arquitecturas. Como sucede en cualquier modelo de estimación, no hay forma de compensar la falta o calidad de los datos de entrada y/o precisión de los valores de los factores de costo. El modelo COCOMO es un ejemplo de modelo algorítmico.

3.27. Esfuerzo y Duración.

3.27.1. Consideraciones destacables del modelo.

En el modelo COCOMO se pueden distinguir los siguientes aspectos relevantes:

1. Los factores de costo del modelo son, en orden de importancia:
    - Tamaño: Cantidad de líneas de código fuente.
    - Factor Exponencial de Escala: Representa el impacto de la economía y deseconomía de escala.
    - Factores Multiplicadores de Esfuerzo: Simbolizan características que influyen en el desarrollo del producto, clasificadas en 4 categorías: plataforma, personal, proyecto y producto.
2. COCOMO asume que la especificación de requerimientos no sufrirá cambios fundamentales después de que culmine la fase de planificación de requerimientos. Algunos refinamientos y reinterpretaciones pueden ser inevitables, por lo tanto, cualquier modificación importante implicará una revisión en la estimación de los costos.
3. El período de desarrollo cubierto por este modelo comienza después de la fase de revisión de requerimientos y finaliza con la aprobación de la fase de testeo.
4. El análisis de distribución de esfuerzo y tiempo de desarrollo por fase y actividad se hereda del modelo COCOMO' 81, donde se asume el uso de un modelo de desarrollo secuencial denominado comúnmente "Cascada" (Waterfall)
5. La estimación de COCOMO abarca todas las tareas en relación directa a las actividades del proyecto, quedando de esta manera excluidas las actividades ejecutadas por operadores, secretarias, administradores de alto nivel, etc.
6. COCOMO evita estimar costos en una unidad monetaria determinada puesto que la unidad mes-persona es más estable, al ser inmune a las fluctuaciones monetarias del mercado. Para convertir mes-persona a dólares se aplica un

promedio del valor mes persona diferente para cada fase del proyecto, lo que permite tener en cuenta los distintos niveles salariales.

Una vez obtenido el total de Puntos de Función, en este caso 146 calculado en la tabla 28, se utiliza COCOMO II para estimar el esfuerzo. Este método se basa en la aplicación de ecuaciones matemáticas sea sobre los Puntos de Función sin ajustar o sobre la cantidad de líneas de código fuente estimadas.

Las mencionadas ecuaciones se encuentran ponderadas por los denominados factores de costo. La ecuación base que se toma es la siguiente.

$$\boldsymbol{PMnominal} = A * (Size)^{B}$$

Donde PM nominal: es el esfuerzo nominal requerido en meses-hombre. Size: es el tamaño que se estima tiene el software a desarrollar medido en miles de líneas de código fuente (SLOC- sigla en inglés comúnmente utilizada para este concepto) o Puntos de Función convertidos a éstas mediante un factor que depende de la tecnología utilizada.

Se considera el lenguaje Php de software libre por lo que el factor de conversión es 12 por cada Punto de Función como se especifica en la tabla 14. Haciendo el cálculo de miles de líneas de código fuentes es:

$$\boldsymbol{Size} = 146 * 12$$

$$\boldsymbol{Size} = 1752$$

A es una constante que denota los efectos multiplicativos en el esfuerzo requerido de acuerdo al crecimiento del tamaño del software. Inicialmente se utiliza el valor 2.94.

B es una constante que afecta exponencialmente al esfuerzo y es determinada mediante la ponderación de aspectos positivos sobre los negativos que afectan al proyecto en cuanto a su complejidad y entorno de desarrollo.

Las variables escalares que permiten determinar el factor B son las siguientes:

Tabla 29. Factores Escalares.

VARIABLE	Descripción
PREC	Experiencia en este tipo de desarrollos.
FLEX	Flexibilidad para el desarrollo.
RSEL	Relacionado con la arquitectura y la mitigación de riesgos.
TEAM	Cohesión y madurez del equipo de desarrollo.
PMAT	Proceso de madurez de desarrollo del software.

Los criterios y valores posibles para cada una de estas variables se encuentran determinados en la siguiente tabla.

Tabla 30. Valores para Factores Escalares.

Factor Wi	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
PREC	Completa	Completa	Algo	Familiar	Muy Familiar	Absolutamente Familiar
	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	Rigurosa	Ocasional	Algo	Generalmente Conforme	Algo de Conformidad	Objetivos Generales
	5.07	4.05	3.04	2.03	1.01	0.00
RESL	Poco (20%)	Algo (40%)	A menudo (60%)	Generalmente (75%)	Mayormente (90%)	Totalmente (100%)
	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	Interac. Muy difícil	Algo dificultosa	Algo Cooperativa	Cooperativa	Altamente Cooperativa	Interacción Total
	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	**Promedio de respuestas afirmativas en cuestionario CMM**					
	7.60 6.24	6.24	4.68	3.12	1.56	0.00

Para el proyecto se han determinado los siguientes valores.

Tabla 31. Factores Escalares para el proyecto.

VARIABLE	Valor para este Proyecto	
PREC	Muy Bajo	6.20
FLEX	Muy Bajo	5.07
RSEL	Muy Bajo	20%(7.07)
TEAM	Alto	2.19
PMAT	Nominal	4.68
TOTAL		**25.21**

La justificación de cada uno de los valores elegidos es la siguiente,

- **PREC:** Muy bajo, dado que se tiene una idea básica sobre el tipo de sistema a desarrollar y experiencia en entornos de desarrollo similares.
- **FLEX:** Muy bajo, ya que se requiere una adaptabilidad considerable del software conforme a los requisitos exigidos.
- **RSEL:** Muy bajo, se tiene una idea clara sobre la arquitectura que debe soportar este sistema.
- **TEAM:** Alto, de acuerdo a la predisposición para el desarrollo de producto.
- **PMAT:** Nominal, por ser la primera vez que se intenta un desarrollo organizado bajo los lineamientos de un proceso determinado.

Entonces el factor escalar B se calcula con la siguiente ecuación gracias a los valores obtenidos en la Tabla 31

B = 0.91+0.01*(25.21)
B = 1.16

El esfuerzo nominal resulta.

$$Pnominal = A(size)^{B}$$

PNominal= 2.94*$(1752)^{1.16}$= 17.01 Mes/Hombre

Para completar la estimación se debe ajustar el esfuerzo nominal de acuerdo a las características del proyecto y los multiplicadores de esfuerzo que le corresponden. Se considera el modelo Post-Arquitectura ya que no hay exploración de arquitecturas alternativas y el concepto de operación ya está definido.
La fórmula a utilizar es:

PMajustado= PMnominal*Π(Multiplicadores de Esfuerzo)

Los valores de los multiplicadores para cada factor por nivel son los siguientes:

3.27.2. Valores para Factores Producto.

Tabla 32. Valores para Factores Producto.

Factores Producto	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
RELY	0.82	0.92	1.00	1.10	1.26	Xxx
DATA	Xxx	0.90	1.00	1.14	1.28	Xxx
DOCU	0.81	0.91	1.00	1.11	2.23	Xxx
CPLX	0.73	0.87	1.00	1.17	1.34	1.74
RUSE	Xxx	0.95	1.00	1.07	1.15	1.24

3.27.3. Valores para Factores Plataforma.

Tabla 33. Valores para Factores Plataforma.

Factores Plataforma	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
TIME	Xxx	Xxx	1.00	1.11	1.29	1.63
STOR	Xxx	Xxx	1.00	1.05	1.17	1.46
PVOL	Xxx	0.91	1.00	1.15	1.30	Xxx

3.27.4. Valores para Factor del Personal.

Tabla 34. Valores para Factor del Personal.

Factores Personal	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
ACAP	1.42	1.19	1.00	0.85	0.71	Xxx
PCAP	1.34	1.15	1.00	0.88	0.76	Xxx
AEXP	1.22	1.10	1.00	0.88	0.81	Xxx
PEXP	1.19	1.09	1.00	0.91	0.85	Xxx
LTEX	1.20	1.09	1.00	0.91	0.84	Xxx
PCON	1.29	1.12	1.00	0.90	0.81	Xxx

3.27.5. Valores para Factores del Proyecto.

Tabla 35. Valores para Factores del Proyecto.

Factores Plataforma	Muy Bajo	Bajo	Nominal	Alto	Muy Alto	Extra Alto
TOOL	1.17	1.09	1.00	0.90	0.78	Xxx
SITE	1.22	1.10	1.00	0.93	0.96	0.80
PCED	1.43	1.14	1.00	1.00	1.00	Xxx

3.27.6. Calculo del multiplicador de Esfuerzo.

Tabla 36. Calculo del multiplicador de Esfuerzo.

Multiplicador	Descripción	Ponderación	Valor
QUE AFECTAN AL PRODUCTO			
RELY	Confiabilidad requerida del software	Alto	1.10
DATA	Tamaño de la base de datos.	Muy Alto	1.28
CPLX	Complejidad del Producto	Nominal	1.00
RUSE	Reusabilidad del Código	Nominal	1.00
DOCU	Requerimiento de documentación	Nominal	1.00
QUE AFECTAN A LA PLATAFORMA			
TIME	Restricciones de tiempo de ejecución	Nominal	1.00
STOR	Restricciones de almacenamiento	Nominal	1.00
PVOL	Volatilidad de la plataforma	Alta	0.87
QUE AFECTAN AL PERSONAL			
ACAP	Capacidad de los analistas	Muy Alto	0.71
PCAP	Capacidad de los programadores	Muy Alto	0.76
AEXP	Experiencia en aplicaciones	Nominal	1.00
PEXP	Experiencia en plataforma	Muy Alto	0.85
LTEX	Experiencia el lenguaje y herramienta de desarrollo	Muy Alto	0.84
PCON	Continuidad del personal	Muy Alto	0.81
QUE AFECTAN AL PROYECTO			
TOOL	Uso de la herramienta de software	Muy Alto	0.78
SITE	Desarrollo en múltiples ubicaciones	Bajo	1.10
SCED	Restricciones de calendario	Nominal	1.00
II (Multiplicadores de Esfuerzo)			**0.298**

La justificación de cada uno de los valores elegidos es la siguiente,

- **RELY:** Alto, por el tipo de tareas que se desea que el sistema realice.

- **DATA:** Muy Alto, debido a que se estima muy probable la utilización de una gran cantidad de datos de entrada y de salida para este sistema.
- **CPLX:** Nominal, debido a que se implementan algoritmos ya conocidos al sistema.
- **RUSE:** Nominal, debido a que el mismo sistema utiliza esta facilidad de un entorno previamente desarrollado.
- **DOCU:** Nominal, por ser la primera vez que se intenta un desarrollo de este tipo se pretende cubrir la documentación básica necesaria.
- **TIME:** Nominal, debido a que es factible que el sistema utilice un gran volumen de datos y por ende los tiempos de ejecución pueden ser extensos.
- **STOR:** Nominal, no hay restricciones en cuanto al almacenamiento.
- **PVOL:** Alto, no se estima cambio de plataforma en el mediano plazo.

De acuerdo al cálculo realizado en la Tabla 36 calculamos en PMajustado.

PMajustado= 17.01*0.298= 5.06 meses/hombre

Tomando en cuenta la cantidad de personal se calcula el tiempo de desarrollo:

Duración=PMajustado/Cantidad personal

***Duración* =** 5.06/2

Duración= *3 meses persona.*

Tomando en cuenta la taza salarial de ingeniero en sistemas en nuestro ambiente, establecemos un salario de 800.00 dólares mensuales.

***Costo total de desarrollo* = 3 * 800**

Costo total de desarrollo = $ 2.400

3.28. Factibilidad Económica.

3.28.1. Costos Recursos Hardware.

En cuanto a los recursos hardware a adquirir, se cuentan:

Tabla 37. Costos Recursos Hardware.

Dispositivos	Cantidad	Precio (1)	Subtotal
Computadores de desarrollo	1	$ 800	$ 800
Escáner	1	$50	$ 50
Cámara Digital	1	$ 120	$ 120
Impresora Láser	1	$ 110	$ 110
Hub	1	$ 30	$ 30
UPS	1	$ 170	$ 170
Tarjeta de red	1	$ 17	$ 17
Cableado	30 metros	$ 10	$ 10
		Total	$ **1.307**

Tabla 38. Costos Recursos Hardware.

3.28.2. Costo Recurso Humano Mensual.

Tabla 39. Costo Recurso Humano Mensual

Personal	Cantidad	Precio	Subtotal
Programador	1	$ 800	$ 800
Diseñador	1	$800	$ 800
Desarrollador Base Datos	1	$ 600	$ 800
		Total	**$ 2.400**

3.28.3. Gastos Mensuales (otros).

Tabla 40. Gastos Mensuales.

Cuentas	Precio
Luz eléctrica	$ 50
Agua	$ 10
Teléfono	$ 30
Arriendo	$200
Insumos de Oficina	$100
Total (pesos)	**$ 390**

3.28.4. Total estimado.

Tabla 41. Total estimado

Recursos Utilizados	
Recursos Hardware	$1.307
Recurso Humano	$2.400
Otros Gastos	$ 390
Total	**$ 4.097**

Habiendo visto de qué manera se desglosan los costos tanto de inversión como de operación, y teniendo en cuenta que los recursos físicos que se requieren para el desarrollo de este proyecto, se puede afirmar que éste es técnica y económicamente factible de llevar a cabo.

Tabla 42. Costos Totales de Desarrollo.

COSTO TOTAL DE DESARROLLO					
Nombre Programa	***Size***	***Pnominal*** $PNominal = A * (Size)^B$	***PMajustado*** $PNominal \times \pi Esfuerzo$	***Duración*** *PMajustado/Cantidad personal*	***Costo Total de Desarrollo*** ***Duración *Tasa salarial***
PHP	$Size = 146 * 12 \quad = 1752$	***PNominal*** **= 2.94*** $(1752)^{1.16}$ **= 17.01 M/H**	***PMajustado*** **= 17.01*0.298 = 5.06 M/H**	***Duración*** **= 5.06/2** ***Duración*** **= 3 meses**	**Resultado: 2,400**
JAVA	$Size = 146 * 17$ **= 2480**	***Pnominal*** **= 2.94*** $(2480)^{1.16}$ **= 25.46 M/H**	***Pmajustado*** **= 25.46 *0.298 = 7.58 M/H**	***Duración*** **= 7.58/2** ***Duración*** **= 4 meses**	**Resultado: 3.200**
ASP	$Size = 146 * 32$ **= 4672**	***PNominal*** **= 2.94*** $(4672)^{1.16}$ **= 53.08 M/H**	***Pmajustado*** **= 53.08*0.298 = 15.81 M/H**	***Duración*** **= 15.81/2** ***Duración*** **= 7.90 meses**	**Resultado: 6,320**

3.29. EL RETORNO DE LA INVERSIÓN DE LA EMPRESA

El retorno de la inversión, conocido por sus siglas ROI, es un indicador esencial en el área económica, ya que nos ayuda reducir el costo total de propiedad, mejorar el éxito de los proyectos y aumentar las inversiones empresariales, las mismas que darán éxito a la empresa.

Principalmente, el ROI se usa al momento de evaluar un proyecto de inversión: si el ROI es menor o igual que cero, significa que el proyecto o futuro negocio no es rentable (factible); y mientras mayor sea el ROI, significa que un mayor porcentaje del capital se va a recuperar al ser invertido en el proyecto. La fórmula del índice de retorno sobre la inversión es:

***ROI** = ((Utilidades – Inversión) / Inversión) x 100*

La inversión total para nuestro proyecto de software (capital invertido) es de **4000**, y el total de las utilidades obtenidas es de **2,400** aplicando la fórmula del ROI:

***ROI** = ((4000 – 2,400) / 2,400) x 100*

Nos da un ROI de **66.66%,** es decir, la inversión tiene una rentabilidad de **66.66**%, el proyecto es netamente factible. Para calcular el tiempo de retorno de la inversión será el total del capital invertido dividido para el costo anual del uso del software.

$$\textbf{\textit{Tiempo de recuperación de la inversión}} = \left(\frac{Totol\ del\ capital\ invertido}{Gastos\ anuales}\right) \times 12$$

$$\textbf{\textit{Tiempo de recuperación de la inversión}} = \left(\frac{2.400}{1.300}\right) \times 12$$

***Tiempo de recuperación de la inversión** = 22.15*

***Tiempo de recuperación de la inversión** = 22 meses*

La inversión realizada en la adquisición del software será recuperada en un plazo de 22 meses es decir 1 año y 10 meses aproximadamente.

CAPITULO IV

4. DISEÑO E IMPLEMENTACION DE LA APLICACIÓN.

4.1. Introducción.

El objetivo de este proceso es el análisis del conjunto concreto de necesidades para proponer una solución que tenga en cuenta las restricciones económicas, técnicas, legales y operativas aplicables a la solución. La solución que se obtendrá de este estudio dará paso para la construcción de nuestro producto software que realizará operaciones con datos para posteriormente analizar e interpretar los mismos.

4.2. Definición de la Metodología.

Se han considerado los siguientes aspectos para la elección de la metodología a usar:

- Los requerimientos detallados del sistema se expresaran como casos de uso, RUP utiliza los Casos de Uso como una técnica de captura de requisitos que fuerza a pensar en términos de importancia para el usuario y no sólo en términos de funciones.

  Se define un Caso de Uso como un fragmento de funcionalidad del sistema que proporciona al usuario un valor añadido. Los Casos de Uso representan los requisitos funcionales del sistema.
- La metodología es flexible a cambios durante el proceso por lo que se ajusta a nuestras necesidades ya que la evolución de sistema puede tener cambios de bajo nivel en el desarrollo.

Es así que la metodología RUP a sido escogida como la ideal para el desarrollo de nuestro sistema ya que al trabajar en conjunto con el modelo de espiral traerán los mejores resultados al finalizar el proceso.

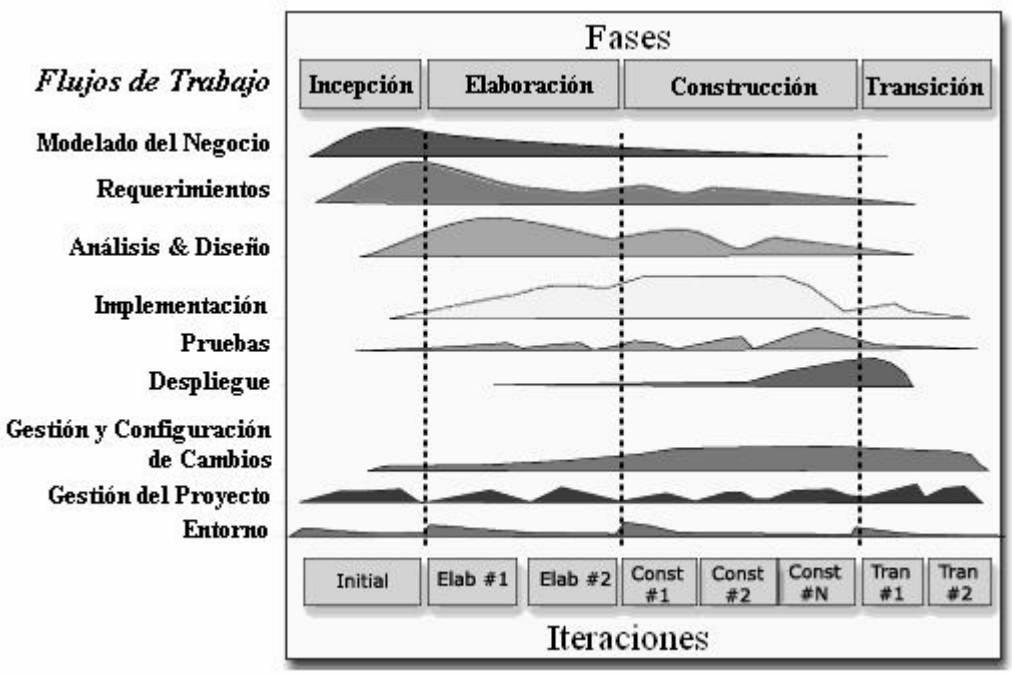


Figura 7. Fases de la metodología.

4.3. Análisis de la situación y restricciones.

Es importante que las interfaces de usuarios sean lo más amigables posible, además debe contener un esquema de seguridad para resguardar a cada usuario, y el trabajo que estese desempeñando.

Se tratará de estudiar el alcance de la necesidad planteada, analizando las posibles restricciones que pueden afectar al producto. Deberán considerarse como antecedentes las aproximaciones iníciales.

4.4. El Proceso de Ingeniería

Un proceso incremental y evolutivo en el desarrollo de aplicaciones Web, permite que el usuario se involucre activamente, facilitando el desarrollo de productos que se ajustan mucho lo que éste busca y necesita, las actividades que formarían parte del marco serían aplicables a cualquier aplicación Web, independientemente de tamaño y complejidad de la misma.

Las actividades que forman parte del proceso son: formulación, planificación, análisis, modelado, generación de páginas, test y evaluación del cliente.

- La Formulación identifica objetivos y establece el alcance de la primera entrega.
- La Planificación genera la estimación del coste general del proyecto, la evaluación de riesgos y el calendario del desarrollo y fechas de entrega.
- El Análisis especifica los requerimientos e identifica el contenido.
- La Modelización se compone de dos secuencias paralelas de tareas. Una consiste en el diseño y producción del contenido que forma parte de la aplicación. La otra, en el diseño de la arquitectura, navegación e interfaz de usuario.
- Es importante destacar la importancia del diseño de la interfaz. Independientemente del valor del contenido y servicios prestados, una buena interfaz mejora la percepción que el usuario tiene de estos.
- La Generación de páginas se integra contenido, arquitectura, navegación e interfaz para crear estática o dinámicamente el aspecto más visible de las aplicaciones, las páginas.
- El Test busca errores a todos los niveles: contenido, funcional, navegacional, rendimiento, etc.

4.5. Descripción General del Sistema.

El sistema fundamentalmente permitirá a los usuarios trabajar con la información de la empresa para poder llevar un control sobre las ventas realizadas y la facturación de las mismas.

Será una aplicación ejecutable bajo el sistema operativo Microsoft Windows y será testeado para las versiones XP, Vista, Windows7. La misma será desarrollada con herramientas de programación software Libre, lenguajes que soportan orientación a objetos y que además poseen una gran ductilidad en el manejo de datos.

El producto final permitirá al usuario manipular información de clientes productos, cuentas por cobrar, cuentas de Banco.

4.6. Definición del Prototipo.

La finalidad de esta fase es definir el prototipo que facilite la determinación de los requerimientos de la aplicación ante el cliente. Antes de realizar la aplicación se presentó un documento a la empresa dando a conocer los procesos que contendrá el sistema, representando por la integración usuario/aplicación.

4.7. Definición de Requisitos Generales del Sistema.

4.7.1. Requisitos Funcionales.

Tabla 43. Requisitos funcionales generales.

ID	Nombre	Descripción
RF-1	Ingreso al Sistema	Una vez que el usuario ha ingresado al sistema observara un mensaje de bienvenida.
RF-2	Salir del Sistema	Al salir del sistema, el mismo emitirá un mensaje comprobando la salida exitosa del sistema.

4.7.2. Requisitos funcionales relacionados con Usuarios.

Tabla 44. Requisitos funcionales relacionados con los usuarios

ID	Nombre	Descripción
RF-3	Agregar Usuario	El sistema podrá registrar nuevos usuarios, para el sistema otorgándole un nombre y un password, cada usuario tendrá una categoría.
RF-4	Modificar Usuario	El sistema de ser necesario modificara un usuario ubicándolo por su nombre y password.
RF-5	Eliminar Usuario	El sistema permitirá la eliminación a un usuario, identificándolo mediante su nombre y password.

4.7.3. Requisitos funcionales relacionados con Clientes.

Tabla 45. Requisitos funcionales relacionados con Clientes

ID	Nombre	Descripción
RF-6	Registrar un nuevo cliente.	El sistema permitirá el ingreso de nuevos clientes, llenando toda la información necesaria.
RF-7	Eliminar cliente del sistema.	El sistema permitirá la eliminación a un cliente, identificándolo mediante la cédula de identidad.
RF-8	Actualización o corrección de los datos del cliente	El sistema permitirá actualización o corrección los datos del cliente identificándolo mediante la cédula de identidad.

4.1.1. Requisitos funcionales relacionados con los Productos.

Tabla 46. Requisitos funcionales relacionados con los Productos.

ID	Nombre	Descripción
RF-9	Registrar un nuevo Producto.	El sistema permitirá el ingreso de nuevos productos, proporcionando toda la información necesaria.
RF-10	Eliminar productos del sistema.	El sistema permitirá la eliminación a un producto, identificándolo mediante su código.
RF-11	Actualización o corrección de información de productos	El sistema permitirá actualización o corrección de la información de productos, identificándolo mediante su código.

4.1.2. Requisitos funcionales relacionados con las Venta y formas de pago.

Tabla 47. Requisitos funcionales relacionados con las Venta y formas de pago.

ID	Nombre	Descripción
RF-12	Realizar una venta	Luego que el cliente realice el pedido de los productos el sistema deberá registrar una venta.
RF-13	Asignar una forma de pago.	El sistema otorgara una forma de pago dependiendo del monto que alcance la compra, (Contado, Crédito o Cheque).
RF-14	Emitir comprobante de Pago.	El sistema emitirá un comprobante por cada pago realizado dado el caso de que la forma de pago sea (crédito o cheque).
RF-15	Emitir Factura	Una vez realizada la venta o cancelados todos los pagos el sistema emitirá una factura con todos los detalles de pago.

4.8. Requisitos de rendimiento.

4.8.1. Estáticos.

Se prevé la instalación de la herramienta en una estación de trabajo. Además se puede consultar como guardar datos en tablas externas para lo cual se requiere la instalación

en un servidor de base de datos y una red LAN. El acceso a datos dependerá del perfil del usuario del sistema.

4.9. Atributos del sistema software.

4.9.1. Fiabilidad.

Por tratarse de un sistema no-crítico se someterá al producto a un programa de pruebas empleando el método de caja negra, lo que asegurará una fiabilidad razonable. Poniendo énfasis en que los algoritmos utilizados estén correctamente implementados.

4.9.2. Disponibilidad.

El sistema deberá asegurar una disponibilidad alta y cortos períodos de mantenimiento, en tal sentido se debe lograr un diseño modular que permita el remplazo de módulos sin afectar el funcionamiento general del sistema.

4.9.3. Seguridad.

La información que accederá y generará es sistema una vez en funcionamiento será de carácter sensible y poseerá gran importancia comercial. En tal sentido, será por parte del usuario el disponer de medidas de seguridad adecuadas con las salidas generadas y el tener conocimiento de los niveles de permisos asignados en las tablas de la base de datos que utilizará.

4.9.4. Portabilidad.

En esta etapa no se prevén requerimientos de portabilidad para este Sistema.

4.10. Requisitos no funcionales (Normas y Estándares).

Tabla 48. Requisitos no funcionales – Normas y Estándares.

ID	Descripción
RNF-01	El sistema debe ser desarrollado empleando la metodología RUP.
RNF-02	El sistema debe presentar una interface con el Usuario.
RNF-03	El subsistema correrá con una configuración de pantalla 1024 x 768
RNF-04	Las interfaces con los usuarios se implementarán mediante entornos visuales propios de los lenguajes de programación que se utilicen.

4.11. Requisitos no funcionales – Seguridad.

Tabla 49. Requisitos no funcionales – Seguridad.

ID	Descripción
RNF-05	Los usuarios deben ingresar su identificación y contraseña para tener acceso al Sistema.
RNF-06	Las claves de usuario deben ser almacenadas de manera encriptado.
RNF-07	Los valores de Disponibilidad de Componentes o sistemas deben ser guardados de manera encriptada para que los usuarios no tengan acceso a esos valores.
RNF-08	.Los operadores no podrán extraer listados de estados de disponibilidad del sistema.

4.12. Requisitos no funcionales – Organización.

Tabla 50. Requisitos no funcionales – Organización.

ID	Descripción
RNF-09	Las herramientas de desarrollo de software y soporte deben ser las escogidas en el análisis del capítulo anterior.
RNF-10	Se requiere personal capacitado.

4.13. Requisitos no funcionales – Backup.

Tabla 51. Requisitos no funcionales – Backup

ID	Descripción
RNF-11	El sistema deberá poseer facilidades para realizar copias de seguridad y, en caso de contingencias, poder restaurar la información existente al momento de realizar la última copia de seguridad.
RNF-12	Se requiere personal capacitado.

4.14. Catálogo de Usuarios.

Tabla 52. Catálogo de Usuarios.

Usuario Administrador	Es el responsable de la administración total del sistema.
Usuario Analista	Es el responsable de las emitir facturas, comprobantes seguimiento de cuentas por cobrar, reportes etc.

4.15. Modelo de Negocio.

El modelo de negocio contempla los procesos principales del negocio bajo análisis y la forma en que los mismos se llevan a cabo. Dentro de este modelo, los procesos se representan mediante casos de uso de negocio. El detalle sobre las actividades llevadas a cabo y las entidades utilizadas para completar cada proceso, se documentan mediante diagramas de actividades.

4.16. Identificación de los Usuarios Participantes y Finales.

En esta tarea se identifican los usuarios participantes y finales, interlocutores tanto en la obtención de requisitos como en la validación de los distintos productos y la aceptación final del sistema. Para ello, se actualiza el catálogo de usuarios.

Dada la importancia que la colaboración de los usuarios tiene en el proceso de obtención de los requisitos, es conveniente determinar quiénes van a participar en las sesiones de trabajo, especificando sus funciones y asignando responsabilidades. Así mismo, se informa del plan de trabajo a los usuarios identificados.

4.17. Objetivos y Alcance del Sistema.

El sistema fundamentalmente permitirá a los usuarios realizar la facturación de la venta de productos dentro de la empresa, Los datos obtenidos podrán ser almacenados en la base de Datos o bien en un archivo plano, el sistema contara con ayudas para facilitar el funcionamiento del mismo.

4.18. Establecimiento de requisitos.

4.18.1. Especificación de Casos de Uso.

Esta tarea es obligatoria en el caso de orientación a objetos, y opcional en el caso de análisis estructurado, como apoyo a la obtención de requisitos.

El objetivo de esta tarea es especificar cada caso de uso identificado en la tarea anterior, desarrollando el escenario.
Para completar los casos de uso, es preciso especificar información relativa a:

- Descripción del escenario, es decir, cómo un actor interactúa con el sistema, y cuál es la respuesta obtenida.
- Precondiciones y poscondiciones.
- Identificación de interfaces de usuario.
- Condiciones de fallo que afectan al escenario, así como la respuesta del sistema (escenarios secundarios).

4.19. Casos de Uso.

A continuación, se detalla el diagrama de casos de uso de sistema.

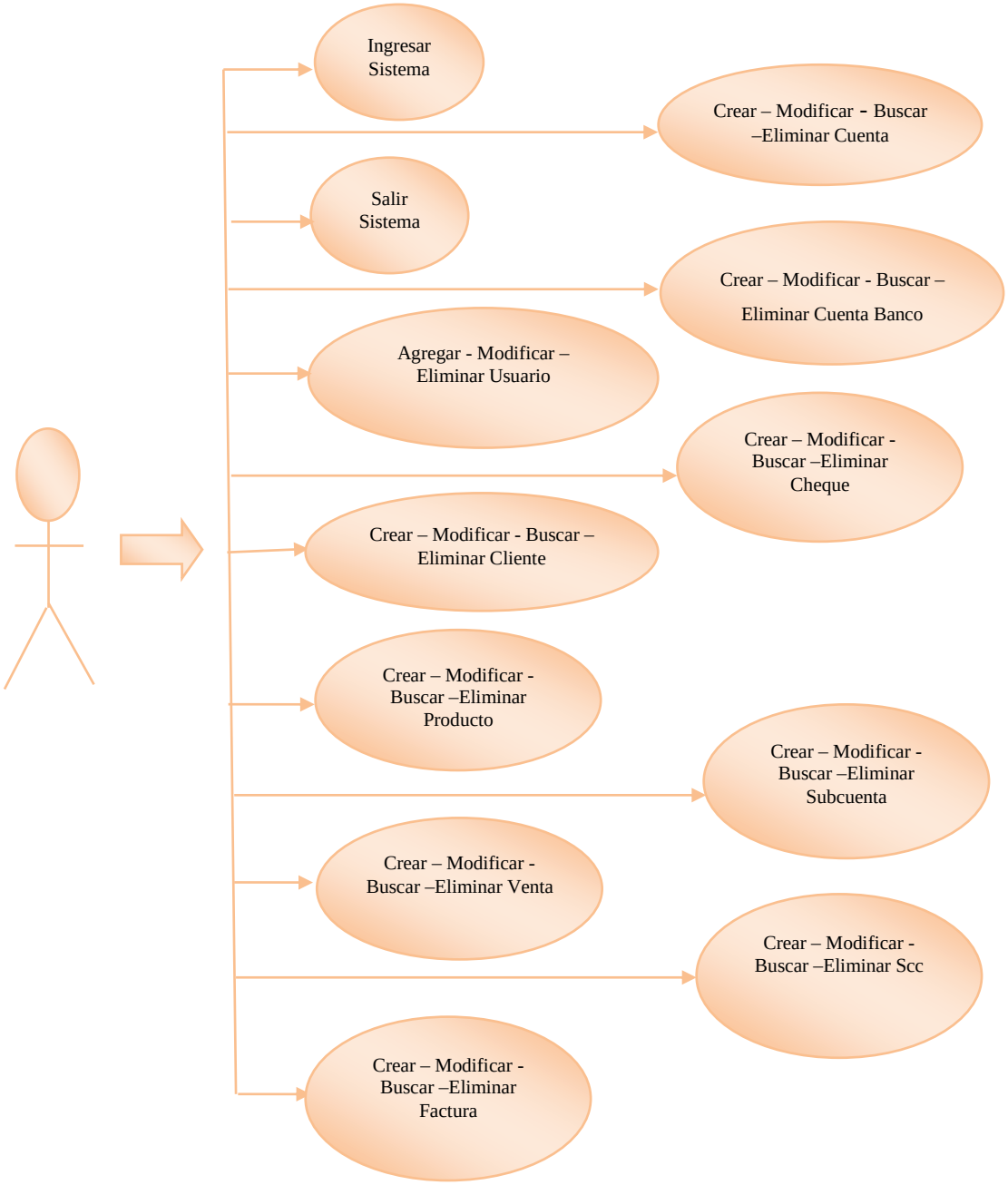


Tabla 53. Diagrama de casos de uso.

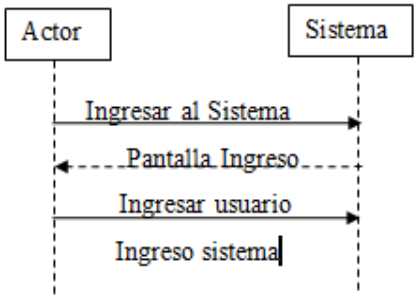
A continuación se detalla la simbología aplicada a las flechas del diagrama:

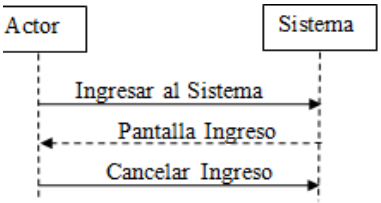
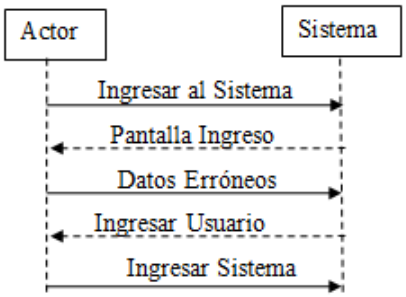
- Las flechas de línea de punto sin estereotipo indican que entre ambos casos de uso existe una relación de Inclusión.
- Las flechas de línea llena que se identifican con el estereotipo “extends” indican que existe una relación de extensión entre ambos casos de uso.
- Las fechas de línea llena sin estereotipo indican que existen una relación de uso entre los participantes.

Descripción de los casos de uso detallados en la figura anterior.

4.19.1. Caso de Uso Ingreso al Sistema.

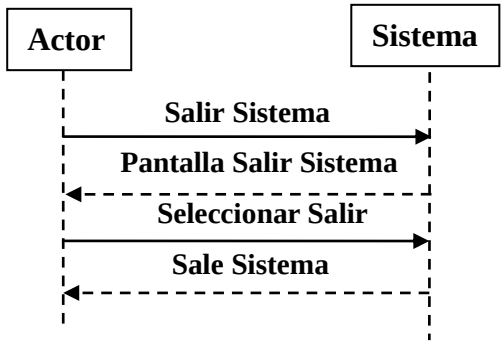
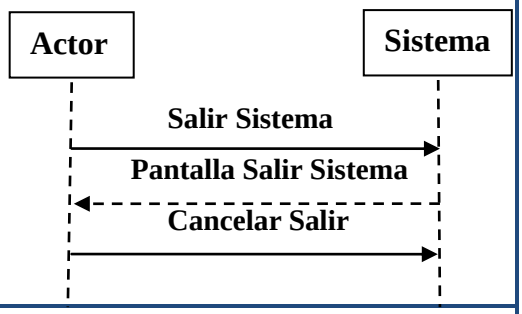
Tabla 54. Descripción del caso de uso Ingreso al Sistema.

Caso de Uso	**Ingreso al sistema (RF01)**
Descripción del caso de uso	Para utilizar el sistema los usuarios debe ingresar un número identificatorio de usuario y una clave válida.
Flujo de eventos	
Activación	Cuando el usuario activa la ejecución del sistema en cuestión se despliega una pantalla tipo “login” denominada “Datos de acceso...”.
1. El usuario ejecuta el sistema. 2. El sistema solicita el ingreso de los datos del usuario. 3. El usuario carga los datos solicitados para ingresar. 4. El sistema valida que los datos ingresados sean correctos. 5. Si la validación es exitosa, el usuario ingresa al sistema, si no solicita la corrección de algún dato. 6. Fin del caso de uso.	

Flujos Alternativos	
7. El sistema debe concluir la ejecución del mismo. 8. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
9. El sistema muestra un mensaje con el error encontrado. Esto sucede 3 veces, si no hay coincidencia a la tercera vez, el sistema termina su ejecución. 10. El sistema vuelve al paso 2	
Requisitos especiales	Cuando se produzcan tres intentos de ingreso fallido consecutivos se debe abandonar el sistema, de forma similar a cuando el usuario oprime el botón de cancelar. Cabe destacar que el usuarios no podrá ingresar al sistema hasta tanto el administrador del mismo lo rehabilite.
Precondiciones	No posee
Puntos de extensión	No posee

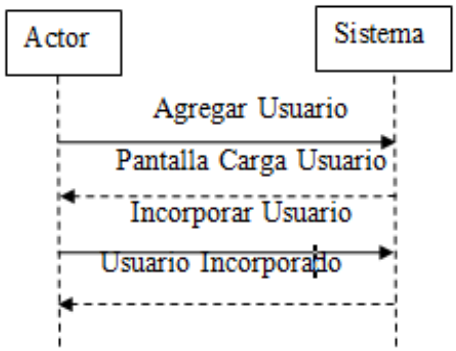
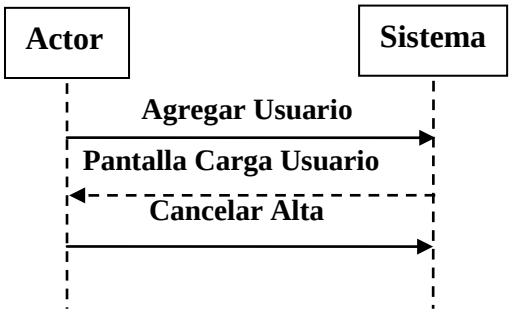
4.19.2. Caso de uso Salir del Sistema.

Tabla 55. Descripción del caso de uso Salir del Sistema.

<table>
<tr><th>Caso de Uso</th><th>Salir del Sistema (RF02)</th></tr>
<tr><td>Descripción del caso de uso</td><td>Con esta opción, los usuarios cierran el sistema.</td></tr>
<tr><td colspan="2">Flujo de eventos</td></tr>
<tr><td>Activación</td><td>El usuario en el menú principal debe ingresar en el grupo “Proyectos” a la opción “Salir”.</td></tr>
<tr><td>Flujo principal</td><td></td></tr>
<tr><td>1. El usuario selecciona la opción “Salir”.
2. Se muestra una ventana “Si / No” con la leyenda “¿Esta seguro que quiere abandonar la aplicación?”.
3. Si el usuario elije “Si”, el sistema cierra todas las tablas y variables y guarda los datos necesarios de la sesión.
4. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “No”.</td></tr>
<tr><td>5. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
6. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Precondiciones</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

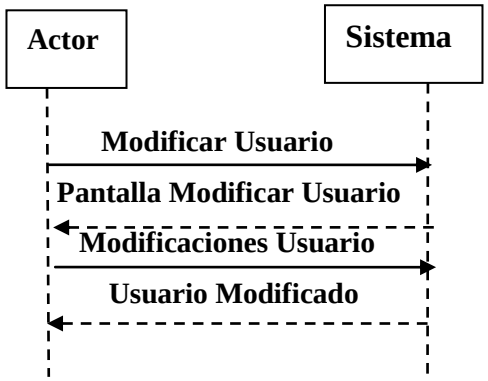
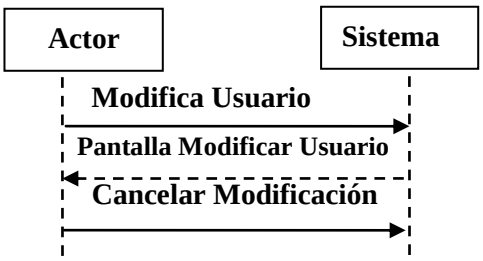
4.19.3. Caso de uso Agregar Usuario.

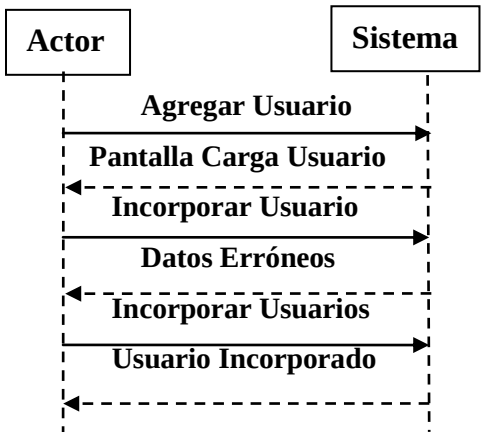
Tabla 56. Caso de uso Agregar Usuario.

Caso de Uso	**Agregar Usuario (RF03)**
Descripción del caso de uso	Se podrá ingresar un nuevo usuario cada vez que se requiera.
Activación	El usuario en la pantalla Ingreso activa la opción “Usuario nuevo”.
Flujo principal	
1. El usuario selecciona la opción “Usuario nuevo”. 2. El sistema solicita el ingreso de los datos del usuario a dar de alta. 3. El usuario ingresa los datos solicitados para dar de alta. 4. El sistema valida que los datos ingresados sean correctos y que no exista en el sistema otro usuario con igual identificador. 5. Si la validación es exitosa, el usuario es ingresado a la base de datos del sistema, si no, solicita la corrección de algún dato. 6. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
7. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción. 8. Fin del caso de uso.	

4.19.4. Caso de uso Modificar Usuario.

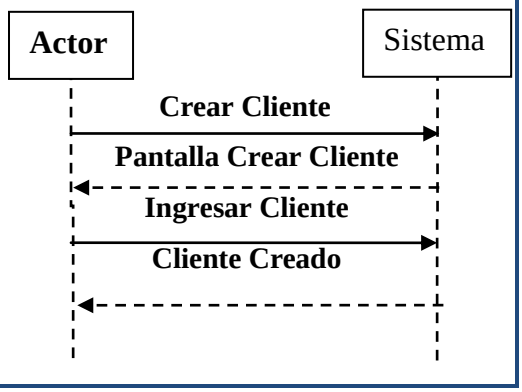
Tabla 57. Descripción del caso de uso Modificar Usuario.

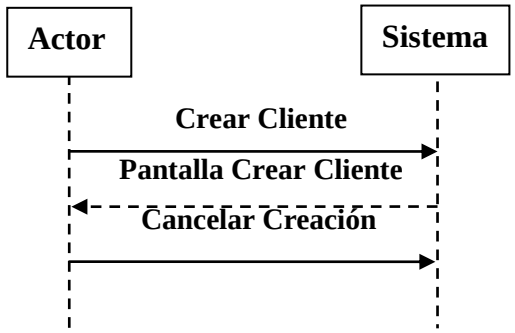
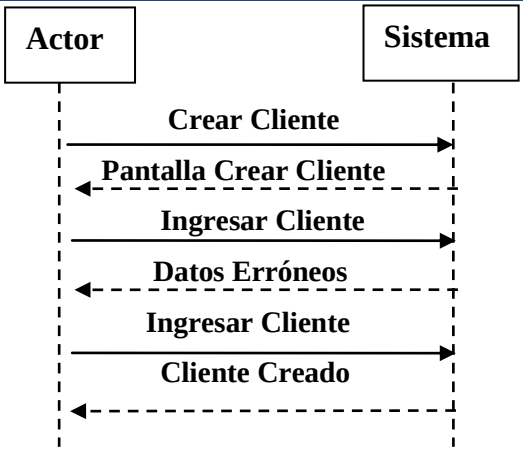
Caso de Uso	**Modificar Usuario (RF04)**
Descripción del caso de uso	Los usuarios ya dados de alta en el sistema pueden ser modificados con tan solo ingresar la cedulausuario.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Usuarios" a la opción "Modificar datos de usuario".
Flujo principal	
El usuario selecciona la opción "Modificar datos de usuario". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados del usuario son actualizados en la base de datos del sistema, si no se solicitara la corrección de algún dato. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
9. El sistema muestra un mensaje con el error	

encontrado 10. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Puntos de extensión	Cambiar Clave

4.19.5. Caso de Uso Crear Cliente.

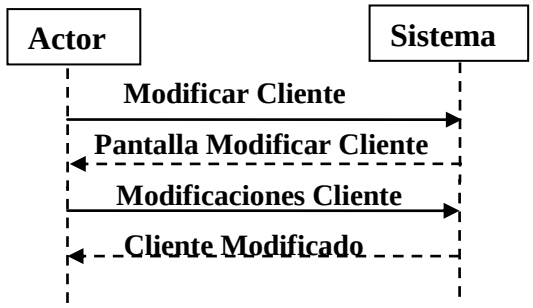
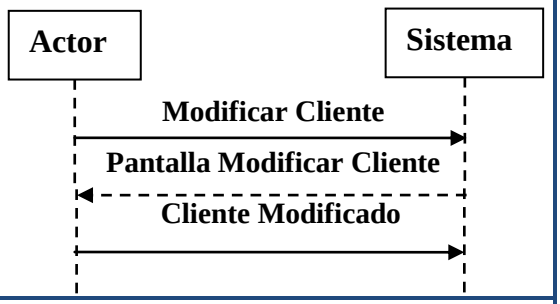
Tabla 58. Descripción del caso de uso Crear Cliente.

Caso de Uso	**Crear Cliente (RF05)**
Descripción del caso de uso	Los usuarios pueden crear un nuevo cliente cuando se lo solicite.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cliente” a la opción “Crear Cliente”.
Flujo principal	
1. El usuario selecciona la opción “Crear Cliente”. 2. El usuario ingresa la descripción del cliente. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos del nuevo cliente son almacenado en la base de	

datos del sistema, si no solicita la corrección de algún dato. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Poscondiciones	No posee
Puntos de extensión	

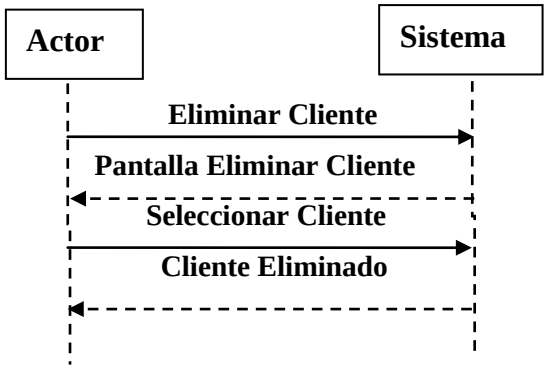
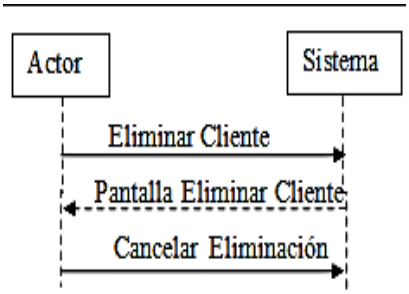
4.19.6. Caso de Uso Modificar Cliente

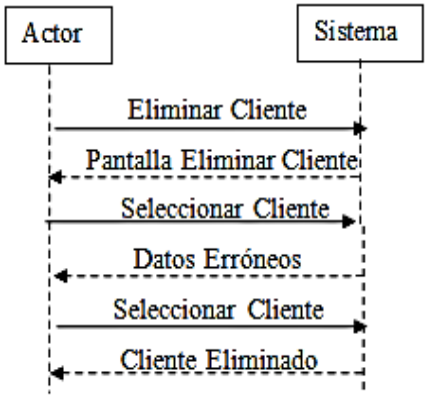
Tabla 59. Descripción del caso de uso Modificar Cliente.

Caso de Uso	**Modificar Cliente (RF06)**
Descripción del caso de uso	Los usuarios podrán modificar un cliente con tan solo ingresar el número de cédula se detallara la información requerida, el usuario seleccionara los campos a modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Clientes" a la opción "Modificar datos de Cliente".
1. El usuario selecciona la opción "Modificar datos Cliente". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados del cliente son actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
Precondiciones	No posee
Puntos de extensión	No posee

4.19.7. Caso de Uso Eliminar Cliente

Tabla 60. Descripción del caso de uso Eliminar Cliente

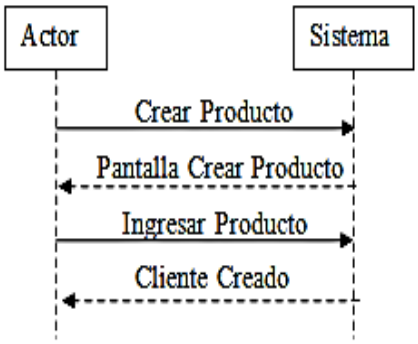
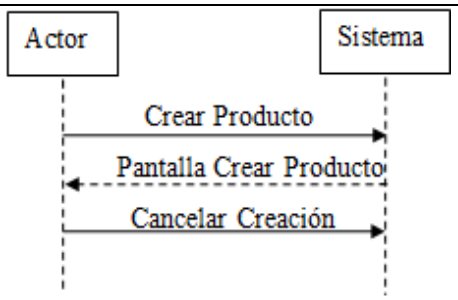
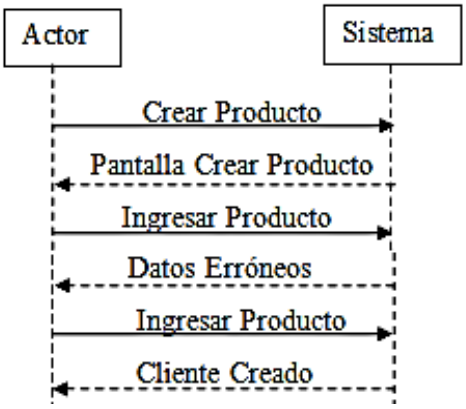
Caso de Uso	Eliminar Cliente (RF07)
Descripción del caso de uso	Los usuarios pueden eliminar un cliente del sistema ingresando el número de cédula se detallara la información requiera, el usuario seleccionara el cliente que será eliminado.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Clientes" a la opción "Eliminar Cliente".
Flujo principal	
1. El usuario selecciona la opción "Eliminar Cliente". 2. El usuario elige de una lista los clientes que van a ser eliminados. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los clientes seleccionados del sistema serán eliminados.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.

8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.8. Caso de Uso crear Producto

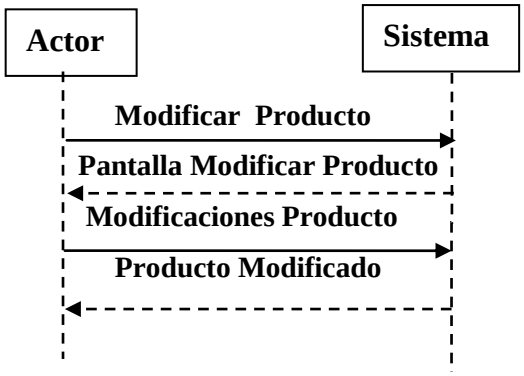
Tabla 61. Descripción del caso de uso Crear Producto.

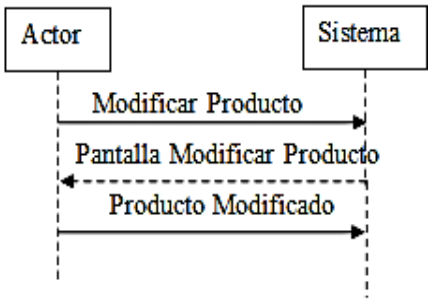
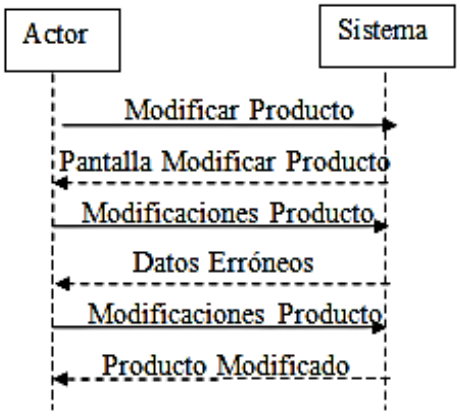
Caso de Uso	**Crear Producto (RF08)**
Descripción del caso de uso	Los usuarios pueden ingresar un nuevo producto cuando se lo solicite.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Producto” a la opción “Crear Producto”.
Flujo principal	
1. El usuario selecciona la opción “Crear Producto”. 2. El usuario ingresa la descripción del producto. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos del	

nuevo producto son almacenado en la base de datos del sistema, si no solicita la corrección de algún dato. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Puntos de extensión	No posee

4.19.9. Caso de uso Modificar Producto.

Tabla 62. Descripción del caso de uso Modificar Producto.

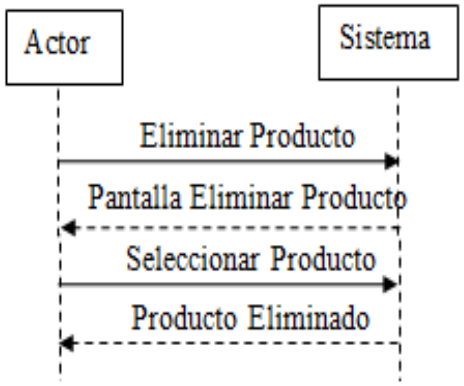
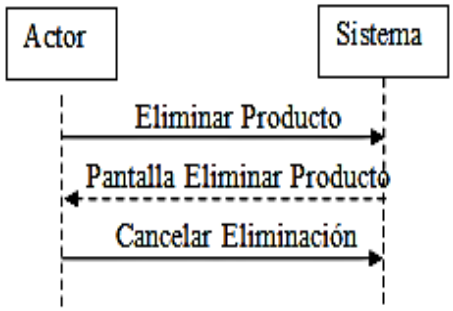
Caso de Uso	**Modificar Producto (RF06)**
Descripción del caso de uso	Los usuarios podrán modificar un producto con tan solo ingresar idproducto automáticamente se detallara toda la información tiene, los usuarios seleccionara los datos que desea modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Productos" a la opción "Modificar datos de Producto".
Flujo principal	
1. El usuario selecciona la opción "Modificar datos Producto". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados del cliente son actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	

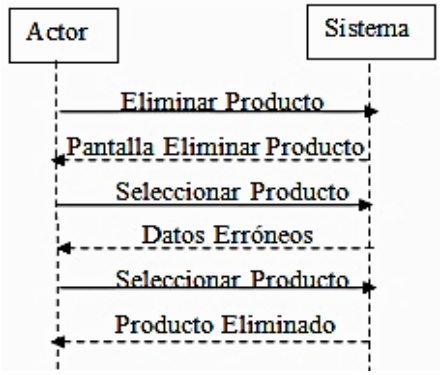
	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.10. Caso de Uso Eliminar Producto.

Tabla 63. Descripción del caso de uso Eliminar Producto.

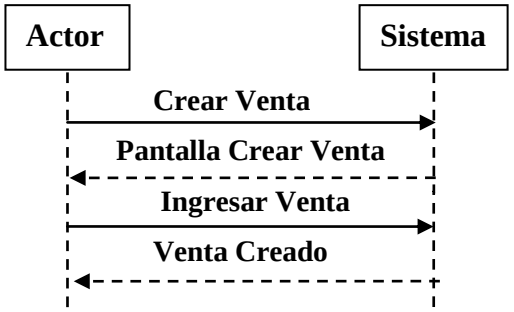
Caso de Uso	**Eliminar Producto(RF10)**
Descripción del caso de uso	Los usuarios pueden eliminar un producto con tan solo ingresar el idproducto se detallara la información que posee, entonces el usuario seleccionara el producto a eliminar.
Flujo de eventos	

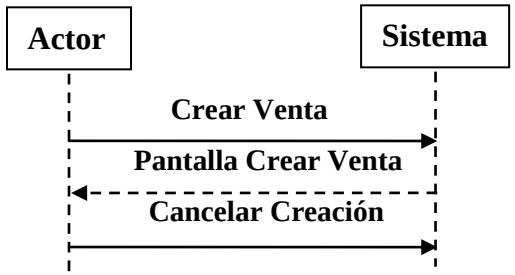
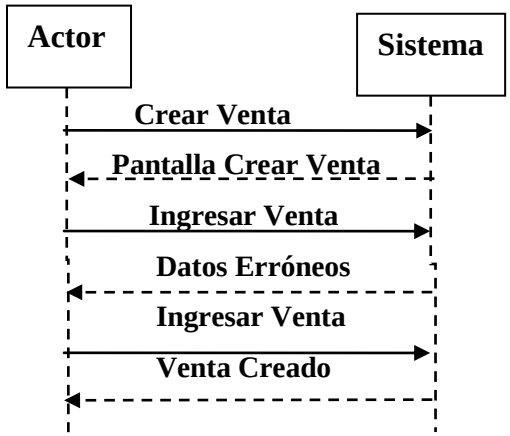
Activación	El usuario en el menú principal debe ingresar en el grupo “Producto” a la opción “Eliminar Producto”.
Flujo principal	
1. El usuario selecciona la opción “Eliminar Producto”. 2. El usuario elige de una lista de productos que desea eliminar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, la versión seleccionada de un producto será eliminada, si no solicita la corrección de algún dato.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	

	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.11. Caso de Uso Crear Venta.

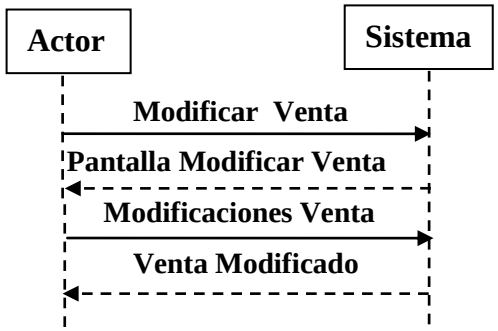
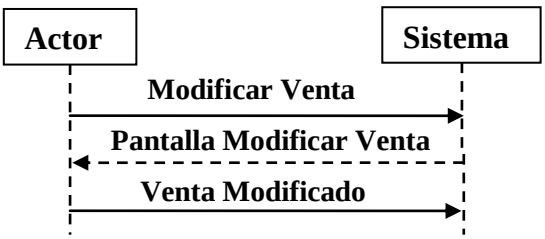
Tabla 64. Descripción del caso de uso Crear Producto.

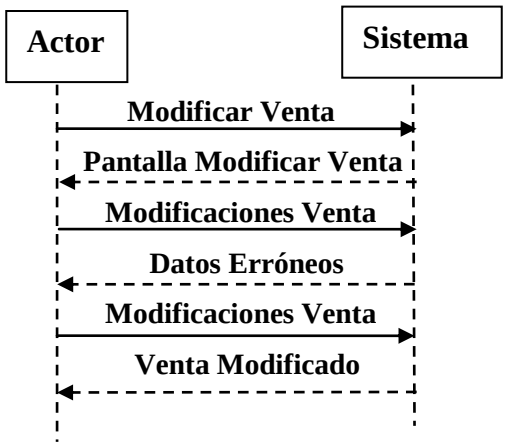
Caso de Uso	Crear Venta (R11)
Descripción del caso de uso	Los usuarios pueden generar una venta simplemente con ingresar los códigos de los productos, se listara los productos existentes, entonces el usuario podrá realizar una venta.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Venta" a la opción "Crear Venta".
Flujo principal	
1. El usuario selecciona la opción "Crear Venta". 2. El usuario ingresa la descripción de la venta. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, la venta	

realizada será almacenada en la base de datos del sistema, o si no se solicitara la corrección de algún dato. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.12. Caso de Uso Modificar Ventas.

Tabla 65. Descripción del caso de uso Modificar Venta.

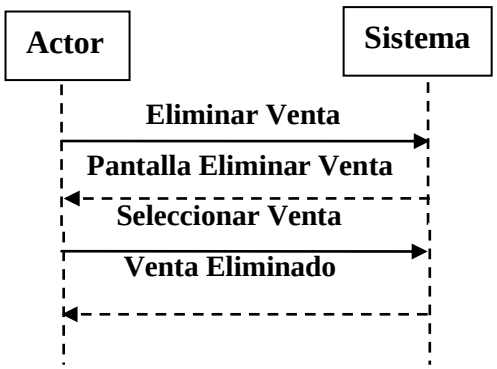
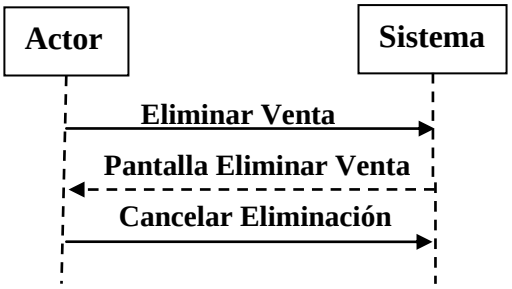
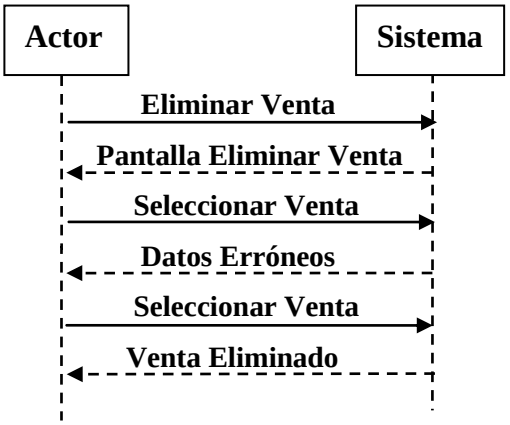
Caso de Uso	Modificar Ventas(RF12)
Descripción del caso de uso	Los usuarios podrán modificar una venta únicamente con ingresar el idfactura en el cual se detallara la venta realizada, entonces el usuario seleccionara el campo que desea modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Ventas" a la opción "Modificar datos de Venta".
Flujo principal	
1. El usuario selecciona la opción "Modificar datos Venta". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados de la venta serán actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.

8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.13. Caso de Uso Eliminar Venta.

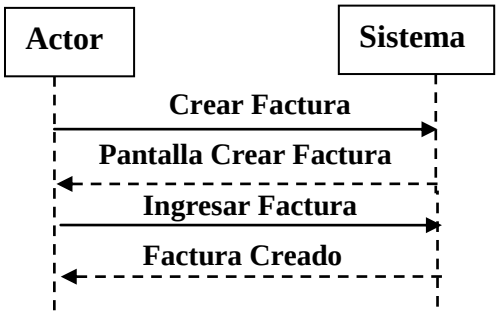
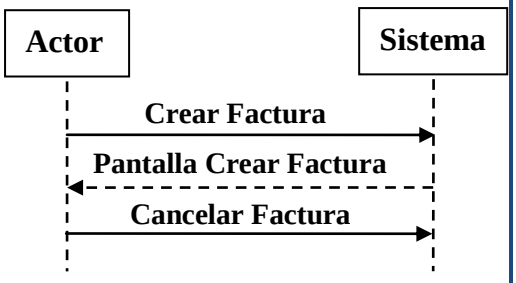
Tabla 66. Descripción del caso de uso Eliminar Venta.

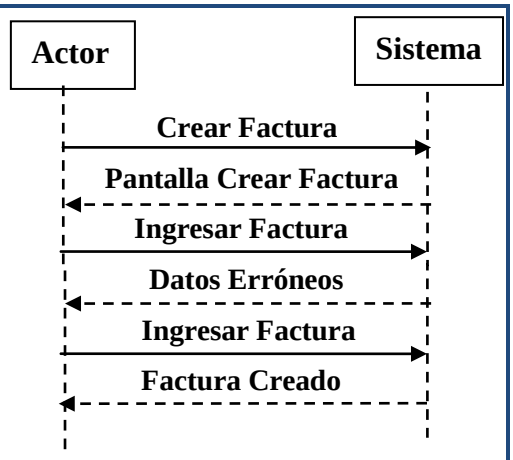
Caso de Uso	Eliminar Venta (RF13)
Descripción del caso de uso	Los usuarios pueden eliminar una venta únicamente con ingresar idfactura automáticamente se detallara la información requerida, el usuario seleccionara los datos que desea ser eliminados.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Venta” a la opción “Eliminar Venta”.
Flujo principal	

<table>
<tr><td>1. El usuario selecciona la opción “Eliminar Venta”.
2. El usuario elige la venta que desea eliminar.
3. El sistema valida que los datos ingresados sean correctos.
4. Si la validación es exitosa, la venta seleccionada será eliminada, o si no solicita la corrección de algún dato.</td><td> 
</td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “Cancelar”.</td></tr>
<tr><td>6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
7. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Alternativa al paso 4</td><td>Los datos ingresados no son correctos.</td></tr>
<tr><td>8. El sistema muestra un mensaje con el error encontrado
9. El sistema vuelve al paso 2</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

4.19.14. Caso de Uso Crear Factura.

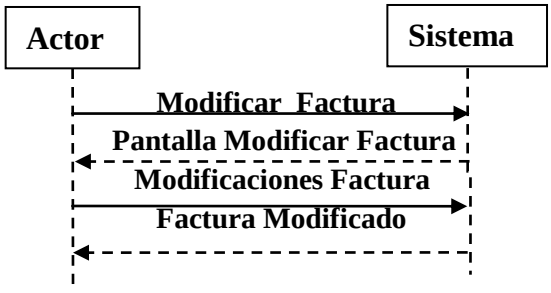
Tabla 67. Descripción del caso de uso Crear Factura.

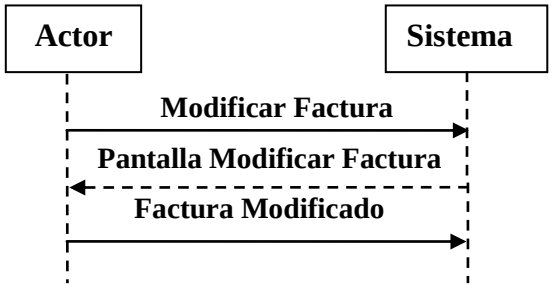
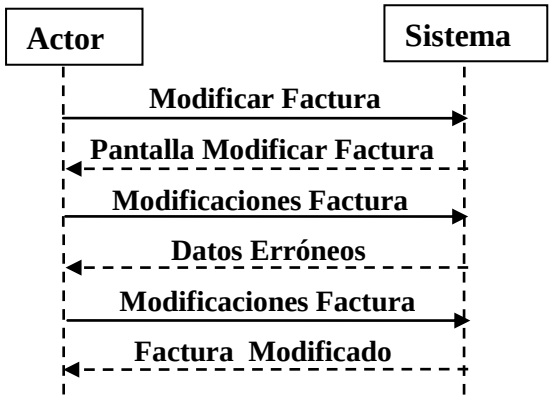
Caso de Uso	Crear Factura (R14)
Descripción del caso de uso	Los usuarios después de haber realizado la venta podrán crear una factura de los productos que sean adquiridos.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Venta" a la opción "Crear Venta".
Flujo principal	
1. El usuario selecciona la opción "Crear Factura". 2. El usuario ingresa los datos respectivos de la factura. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, la venta realizada será almacenada en la base de datos del sistema, o si no solicita la corrección de algún dato. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.

8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.15. Caso de Uso Modificar Factura.

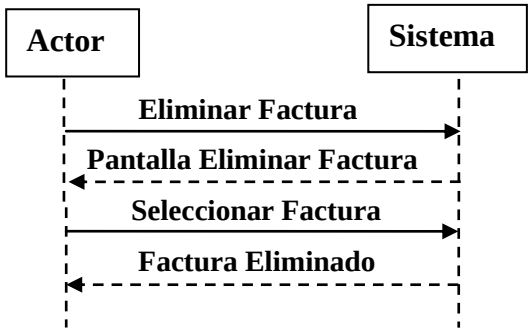
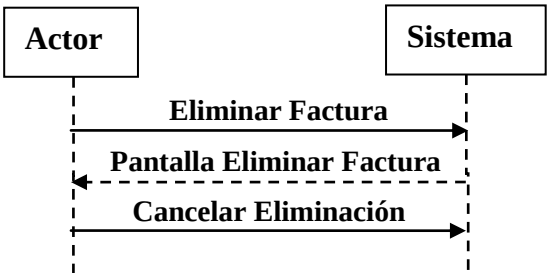
Tabla 68. Descripción del caso de uso Modificar Factura.

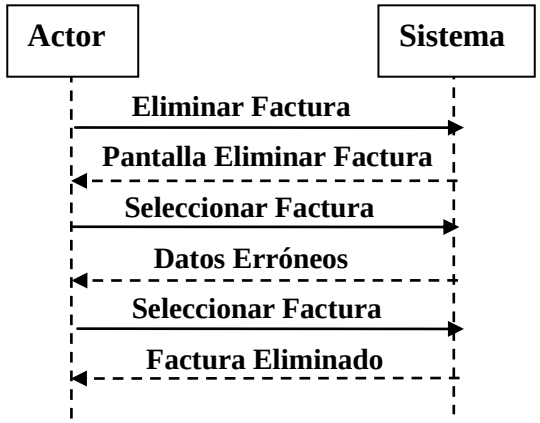
Caso de Uso	Modificar Factura (RF15)
Descripción del caso de uso	Los usuarios podrán modificar las facturas únicamente con ingresar el número de factura automáticamente el sistema mostrara la factura que va a ser modificada.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Factura” a la opción “Modificar datos de Factura”.
Flujo principal	
1. El usuario selecciona la opción “Modificar datos Factura”. 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos	

modificados de la factura serán actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.16. Caso de Uso Eliminar Factura.

Tabla 69. Descripción del caso de uso Eliminar Factura.

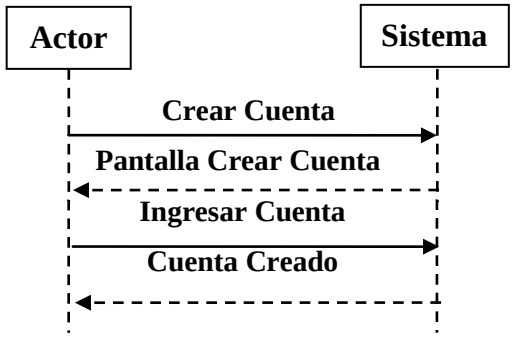
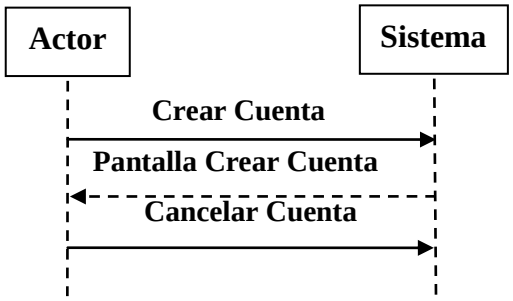
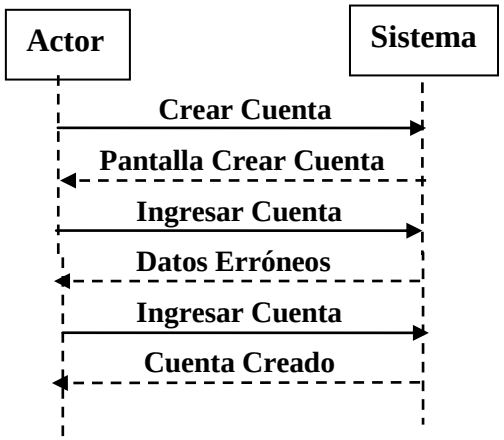
Caso de Uso	Eliminar Factura (RF13)
Descripción del caso de uso	Los usuarios podrán eliminar una factura únicamente con ingresar el número de factura automáticamente se detallara lo que la factura posee y se seleccionara la factura a ser eliminada.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Factura" a la opción "Eliminar Factura".
Flujo principal	
1. El usuario selecciona la opción "Eliminar Factura". 2. El usuario selecciona la factura a ser eliminada. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, la factura seleccionada será eliminada, o si no solicita la corrección de algún dato.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	

Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.17. Caso de Uso Crear Cuenta.

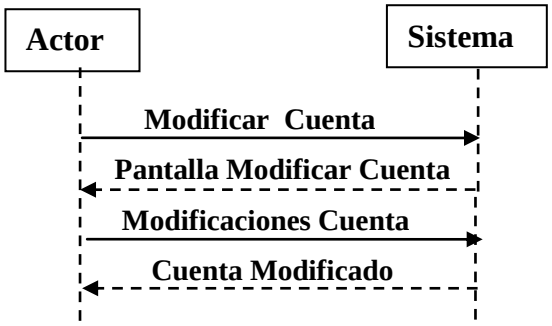
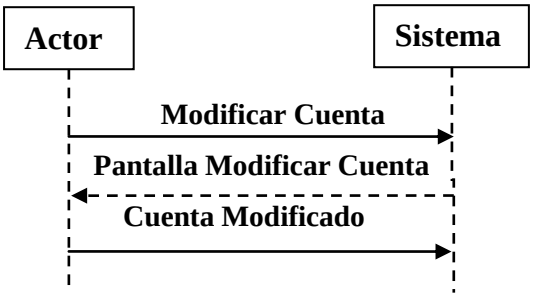
Tabla 70. Descripción del caso de uso Crear Cuenta.

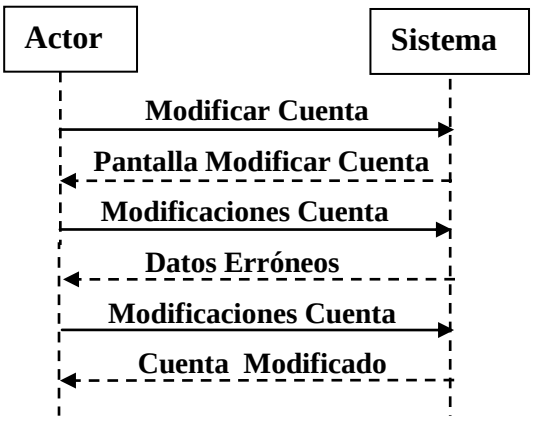
Caso de Uso	Crear Cuenta (R17)
Descripción del caso de uso	Los usuarios podrán crear el modulo cuenta en el cual se detallara toda la información que la misma va a poseer..
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cuenta” a la opción “Crear Cuenta”.
Flujo principal	
1. El usuario selecciona la opción “Crear Cuenta”. 2. El usuario ingresa los datos respectivos de la cuenta. 3. El sistema valida que los datos ingresados	

<table>
<tr><td>sean correctos.
4. Si la validación es exitosa, la cuenta será almacenada en la base de datos del sistema, o si no se solicitara la corrección de algún dato.
5. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “Cancelar”.</td></tr>
<tr><td>6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
7. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Alternativa al paso 4</td><td>Los datos ingresados no son correctos.</td></tr>
<tr><td>8. El sistema muestra un mensaje con el error encontrado
9. El sistema vuelve al paso 2</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Precondiciones</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

4.19.18. Caso de Uso Modificar Cuenta.

Tabla 71. Descripción del caso de uso Modificar Factura.

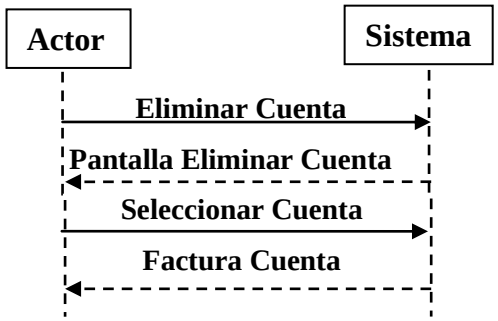
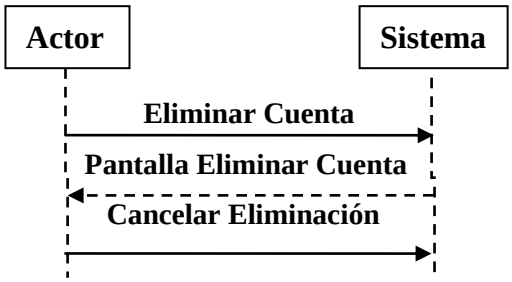
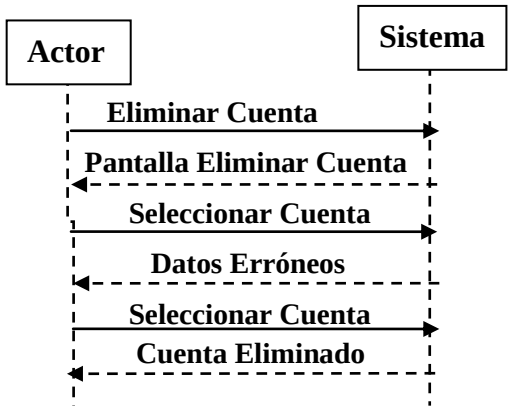
Caso de Uso	Modificar Cuenta (RF18)
Descripción del caso de uso	Los usuarios podrán modificar una cuenta con tan solo ingresar idcuenta automáticamente se detallara los datos de la información que posee el usuario seleccionara los campos a modificar.
Activación	El usuario en el menú principal debe ingresar en el grupo "Cuenta" a la opción "Modificar datos de Cuenta".
Flujo principal	
1. El usuario selecciona la opción "Modificar datos Cuenta". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados de la cuenta serán actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	

	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.19. Caso de Uso Eliminar Cuenta.

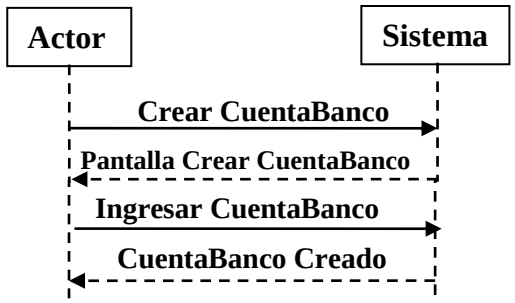
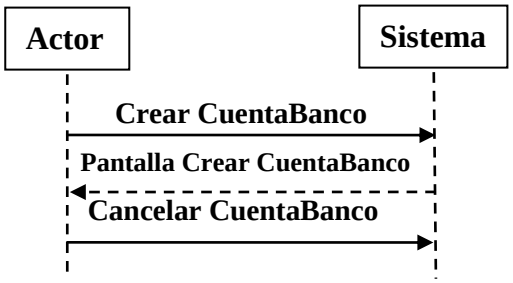
Tabla 72. Descripción del caso de uso Eliminar Cuenta

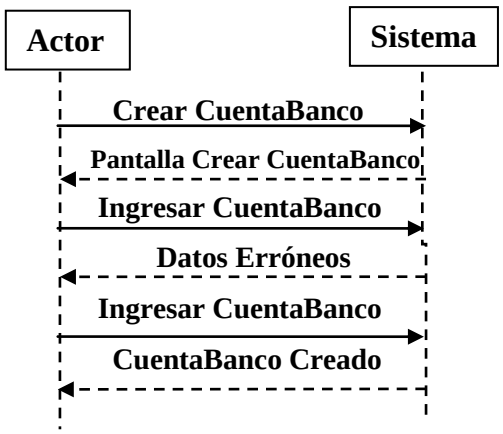
Caso de Uso	Eliminar Cuenta (RF19)
Descripción del caso de uso	Los usuarios podrán eliminar una cuenta únicamente con ingresar idcuenta automáticamente se detallara los datos que posee el usuario seleccionara los campos que van hacer eliminados.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cuenta” a la opción “Eliminar Cuenta”.
Flujo principal	

<table>
<tr><td>1. El usuario selecciona la opción “Eliminar Cuenta”.
2. El usuario selecciona la cuenta a ser eliminada.
3. El sistema valida que los datos ingresados sean correctos.
4. Si la validación es exitosa, la cuenta seleccionada será eliminada, o si no se solicitara la corrección de algún dato.</td><td> 
</td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “Cancelar”.</td></tr>
<tr><td>6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
7. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Alternativa al paso 4</td><td>Los datos ingresados no son correctos.</td></tr>
<tr><td>8. El sistema muestra un mensaje con el error encontrado
9. El sistema vuelve al paso 2</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

4.19.20. Caso de uso Crear CuentaBanco.

Tabla 73. Descripción del caso de uso Crear CuentaBanco.

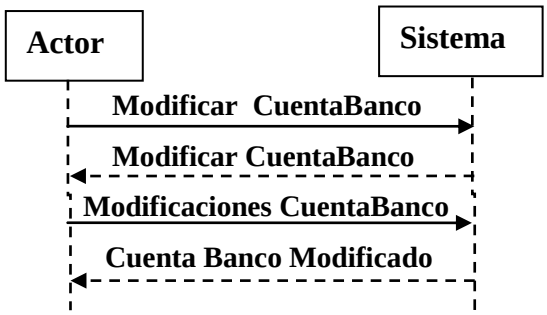
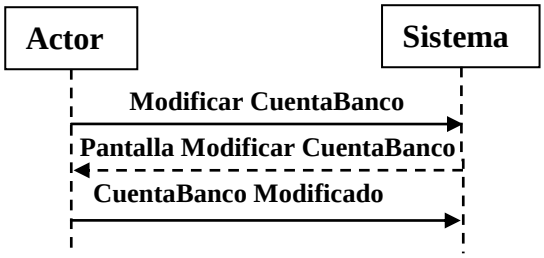
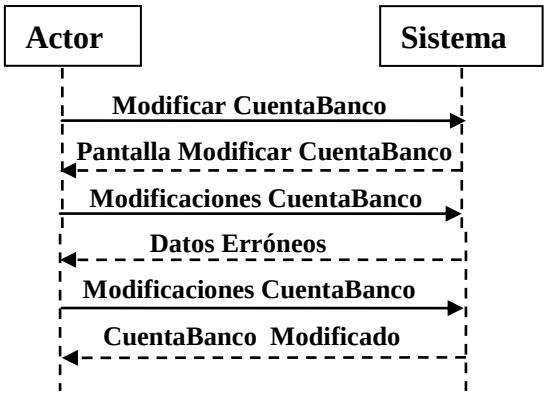
Caso de Uso	Crear CuentaBanco (R20)
	Los usuarios podrán crear un módulo cuentabanco en el cual se registraran toda la información del dueño de la cuenta.
Activación	El usuario en el menú principal debe ingresar en el grupo "CuentaBanco" a la opción "Crear CuentaBanco".
Flujo principal	
1. El usuario selecciona la opción "Crear CuentaBanco". 2. El usuario ingresa los datos respectivos de la cuentabanco. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos del campo cuentbanco será almacenada en la base de datos del sistema, o si no se solicitara la corrección de algún dato. 5. Fin del caso de uso.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el	

error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.21. Caso de Uso Modificar Cuenta Banco.

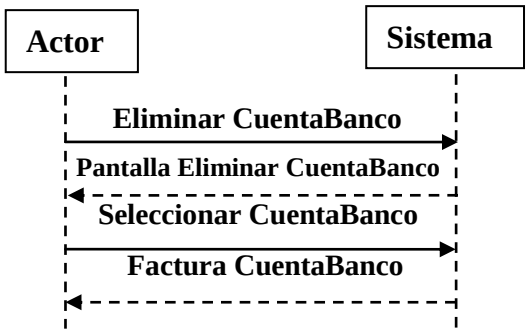
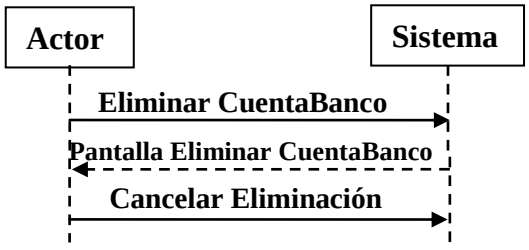
Tabla 74.Descripción del caso de uso Modificar CuentaBanco.

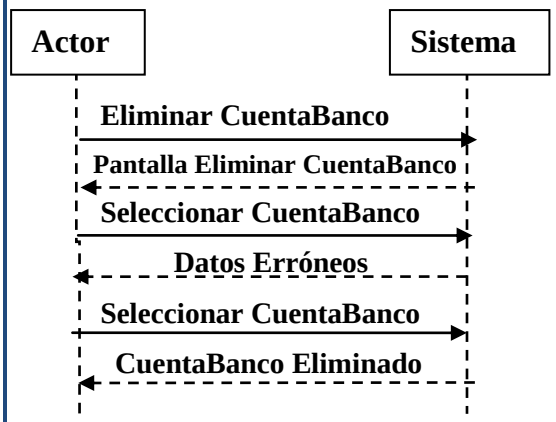
Caso de Uso	Modificar Cuenta Banco (RF21)
Descripción del caso de uso	Los usuarios podrán modificar una la cuenta Banco con tan solo ingresar el número de cuenta que se desea modificada automáticamente se detallara los datos de la cuenta que se desea modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cuenta” a la opción “Modificar datos de Cuenta”.
Flujo principal	
1. El usuario seleccionara la opción “Modificar datos Cuenta”. 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados	

sean correctos. 4. Si la validación es exitosa, los datos modificados de la cuenta banco serán actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón “Cancelar”.
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.22. Caso de uso Eliminar CuentaBanco.

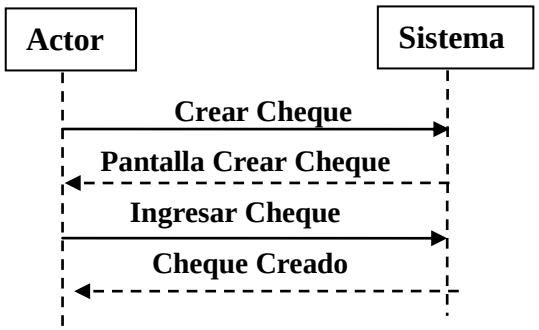
Tabla 75. Descripción del caso de uso Eliminar CuentaBanco.

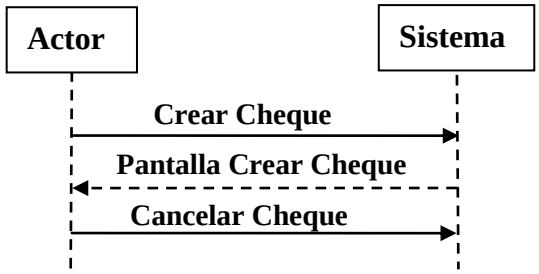
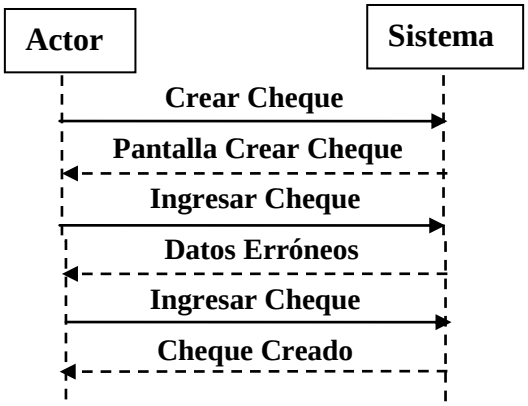
Caso de Uso	Eliminar Cuenta Banco (RF22)
Descripción del caso de uso	Los usuarios podrán eliminar la cuenta Banco únicamente con ingresar el número de cuenta automáticamente se detallara los datos a eliminar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Cuenta Banco" a la opción "Eliminar Cuenta Banco".
Flujo principal	
1. El usuario selecciona la opción "Eliminar Cuenta Banco". 2. El usuario selecciona la cuenta banco a ser eliminada. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, la cuenta banco seleccionada será eliminada, o si no se solicitara la corrección de algún dato.	
Flujos Alternativos	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	

Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado 9. El sistema vuelve al paso 2.	
Requisitos especiales	No posee
Puntos de extensión	No posee

4.19.22. Caso de Uso Crear Cheque.

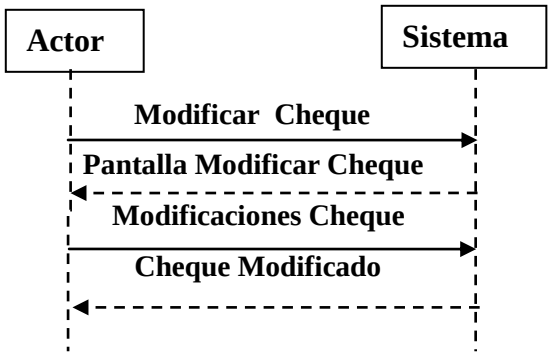
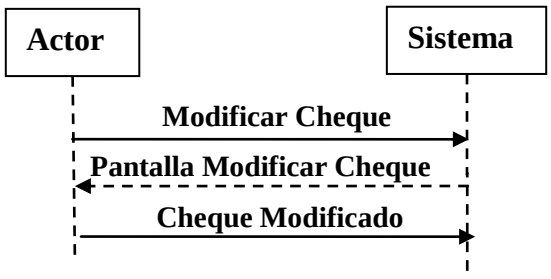
Tabla 76. Descripción del caso de uso Crear Cheque.

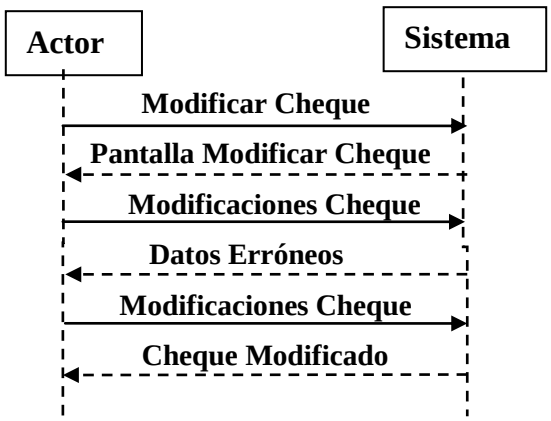
Caso de Uso	**Crear Cheque (R23)**
Descripción del caso de uso	Los usuarios podrán crear un módulo cheque en el mismo que se detallará toda la información del cheque.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cheque” a la opción “Crear Cheque”.
Flujo principal	
1. El usuario selecciona la opción “Crear Cheque”. 2. El usuario ingresa los datos respectivos del cheque. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, el cheque será	

<table>
<tr><td>almacenada en la base de datos del sistema, o si no se solicitara algún cambio en los datos.
5. Fin del caso de uso.</td><td></td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “Cancelar”.</td></tr>
<tr><td>6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
7. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Alternativa al paso 4</td><td>Los datos ingresados no son correctos.</td></tr>
<tr><td>8. El sistema muestra un mensaje con el error encontrado
9. El sistema vuelve al paso 2</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Precondiciones</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

4.19.23. Caso de Uso Modificar Cheque.

Tabla 77. Descripción del caso de uso Modificar Cheque.

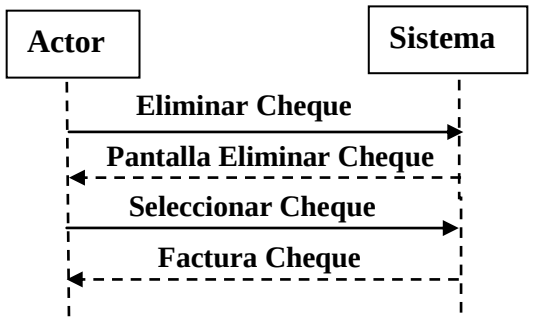
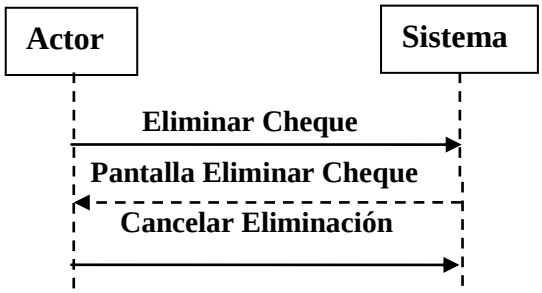
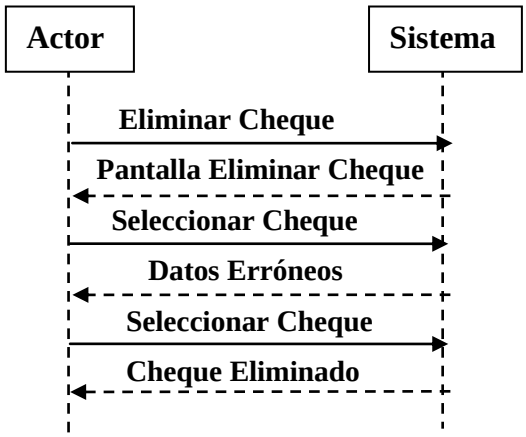
Caso de Uso	Modificar Cheque (RF24)
Descripción del caso de uso	Los usuarios podrán modificar el módulo cheque con tan solo ingresar idcheque automáticamente se detallará toda la información que posee y se seleccionará los campos a modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo "Cheque" a la opción "Modificar datos de Cheque".
Flujo principal	
1. El usuario selecciona la opción "Modificar datos Cheque". 2. El usuario ingresa los datos a modificar. 3. El sistema valida que los datos ingresados sean correctos. 4. Si la validación es exitosa, los datos modificados del cheque serán actualizados en la base de datos del sistema. 5. Fin del caso de uso.	
Alternativa al paso 2	El usuario presiona el botón "Cancelar".
6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú. 7. Fin del caso de uso.	
Alternativa al paso 4	Los datos ingresados no son correctos.
8. El sistema muestra un mensaje con el error encontrado	

9. El sistema vuelve al paso 2	
Requisitos especiales	No posee
Precondiciones	No posee
Puntos de extensión	No posee

4.19.24. Caso de Uso Eliminar Cheque.

Tabla 78. Descripción del caso de uso Eliminar Cheque.

Caso de Uso	Eliminar Cheque (RF25)
Descripción del caso de uso	Los usuarios podrán eliminar el campo cheque únicamente con ingresar el idcheque automáticamente se detallará los datos que el cheque posee el usuario seleccionara los campos a modificar.
Flujo de eventos	
Activación	El usuario en el menú principal debe ingresar en el grupo “Cheque” a la opción “Eliminar Cheque”.
Flujo principal	

<table>
<tr><td>1. El usuario selecciona la opción “Eliminar Cheque”.
2. El usuario selecciona el cheque a ser eliminada.
3. El sistema valida que los datos ingresados sean correctos.
4. Si la validación es exitosa, el cheque seleccionada será eliminada, o si no solicita la corrección de algún dato.</td><td> 
</td></tr>
<tr><td colspan="2">Flujos Alternativos</td></tr>
<tr><td>Alternativa al paso 2</td><td>El usuario presiona el botón “Cancelar”.</td></tr>
<tr><td>6. El sistema debe regresar a la pantalla donde se encontraba antes de que se seleccione esta opción de menú.
7. Fin del caso de uso.</td><td> 
</td></tr>
<tr><td>Alternativa al paso 4</td><td>Los datos ingresados no son correctos.</td></tr>
<tr><td>8. El sistema muestra un mensaje con el error encontrado
9. El sistema vuelve al paso 2</td><td> 
</td></tr>
<tr><td>Requisitos especiales</td><td>No posee</td></tr>
<tr><td>Puntos de extensión</td><td>No posee</td></tr>
</table>

4.20. Análisis de los Casos de Uso.

4.20.1. Descripción de la Interacción de Objetos.

El objetivo de esta tarea es describir la cooperación entre los objetos utilizados para la realización de un caso de uso, que ya fueron identificados en la tarea anterior.

Para representar esta información, se expondrán a modo de esquemas gráficos que contienen instancias de los actores participantes, objetos, y la secuencia de mensajes intercambiados entre ellos. Se pueden establecer criterios para determinar qué tipo de objetos y mensajes se va a incluir en este diagrama, como por ejemplo: si se incluyen objetos y llamadas a bases de datos, objetos de interfaz de usuario, de control, etc.

Estos diagramas pueden ser tanto de secuencia como de colaboración, y su uso depende de si se quieren centrar en la secuencia cronológica o en cómo es la comunicación entre los objetos.

En aquellos casos en los que se especifique más de un escenario para un caso de uso, puede ser conveniente representar cada uno de ellos en un diagrama de interacción, también es recomendable, sobre todo en el caso anterior, completar los diagramas con una descripción textual.

4.21. Identificación de la Relación entre Objetos.

A continuación se muestran los esquemas de relaciones entre objetos del sistema.

4.21.1. Caso de Uso Ingresar al sistema (RF01).

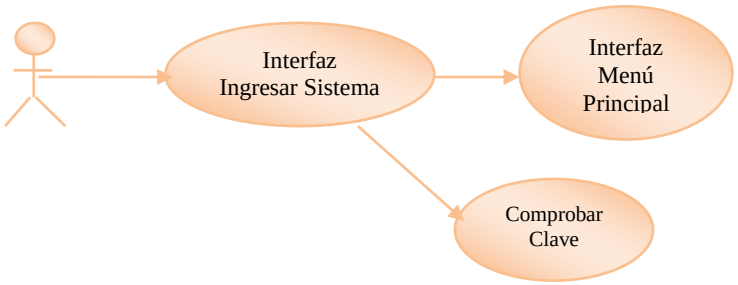


Figura 8. Ingresar al Sistema

4.21.2. Caso de Uso Agregar Usuario (RF02).

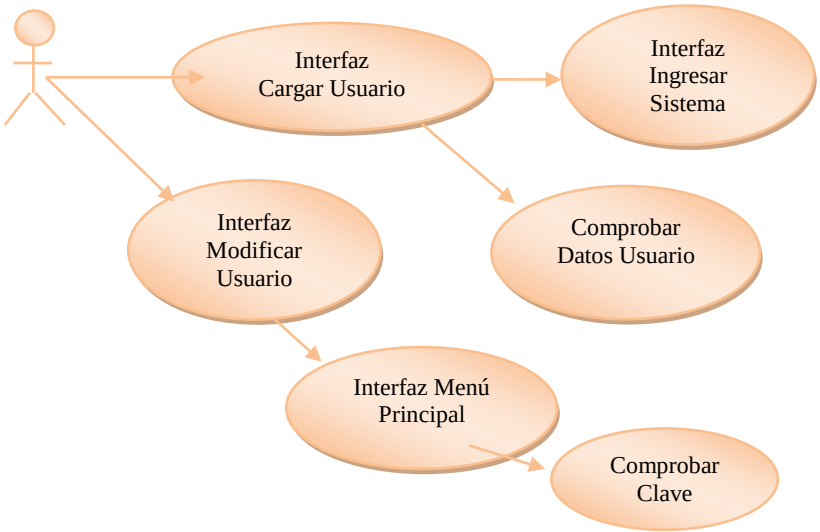


Figura 9. Agregar Usuario.

4.21.3. Caso de Uso Crear – Buscar – Modificar – Eliminar Cliente (RF03).

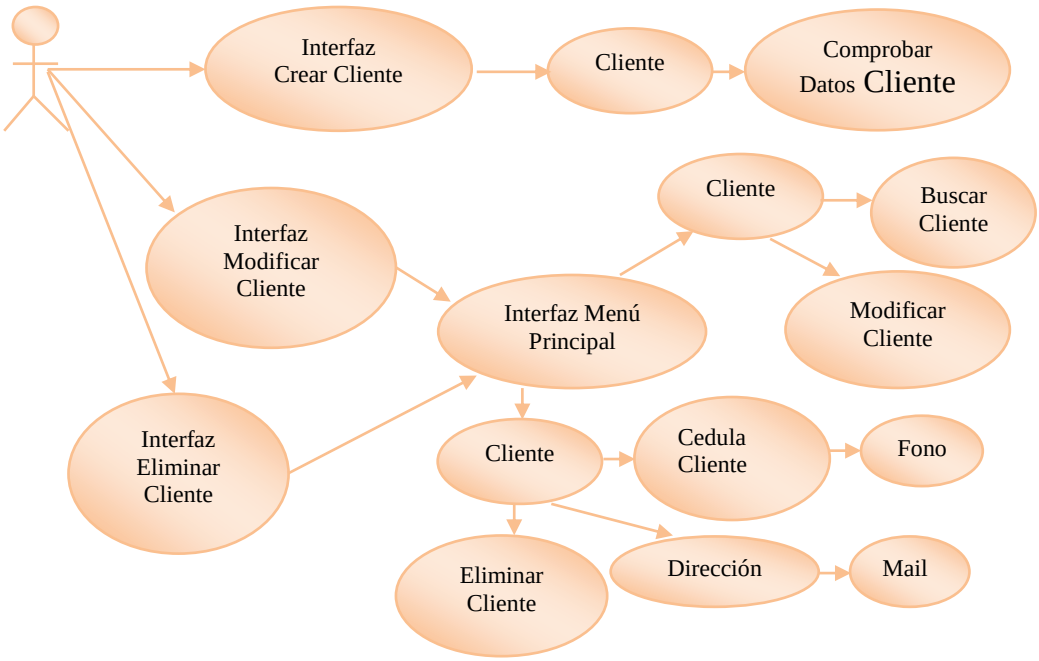


Figura 10. Caso de uso crear – buscar – modificar – eliminar Cliente.

4.21.4. Caso de Uso Crear – Buscar- Modificar – Eliminar Producto (RF04).

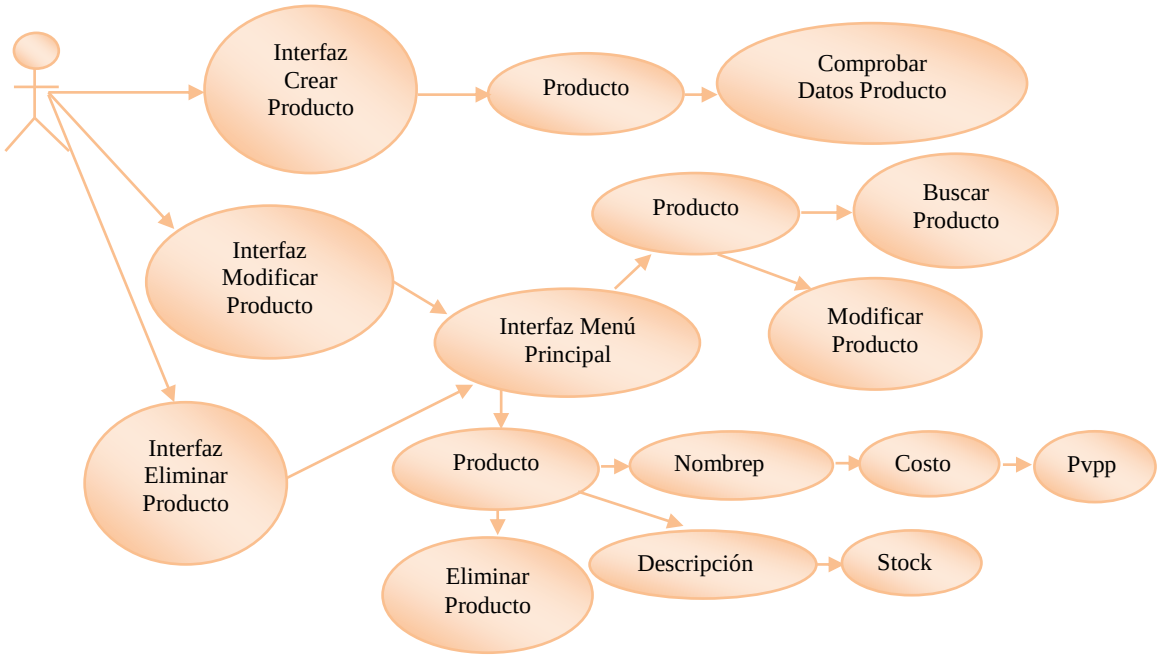


Figura 11. Caso de uso crear – buscar- modificar – eliminar Producto

4.21.5. Caso de Uso Crear – Buscar- Modificar – Eliminar Ventas (RF05).

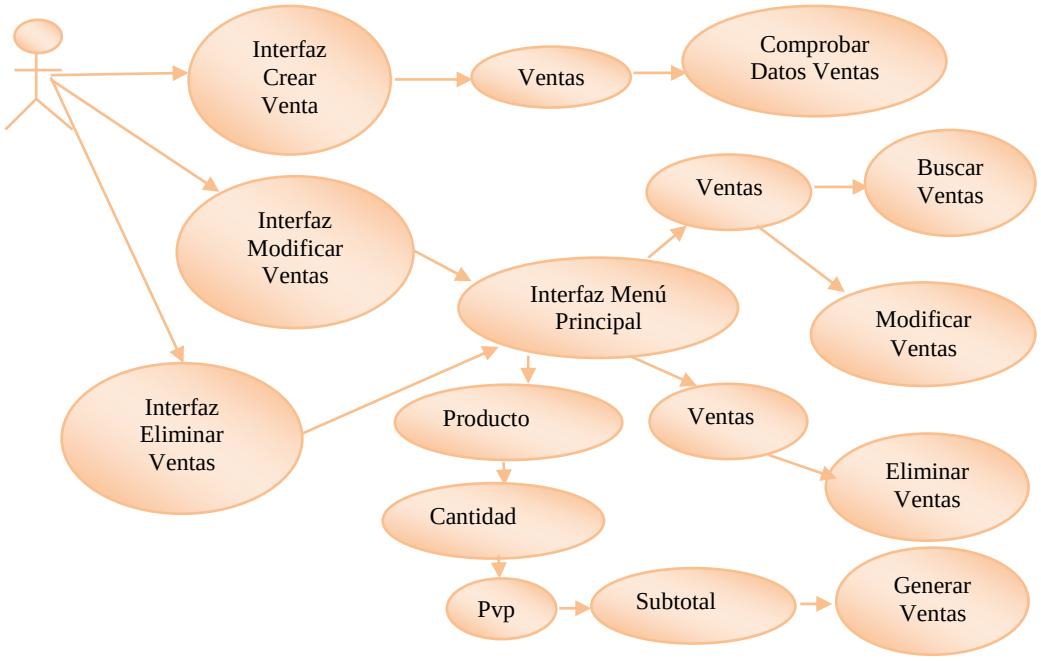


Figura 12. Caso de uso crear – buscar- modificar – eliminar Ventas

4.21.6. Caso de Uso Crear – Buscar- Modificar – Eliminar Factura (RF06).

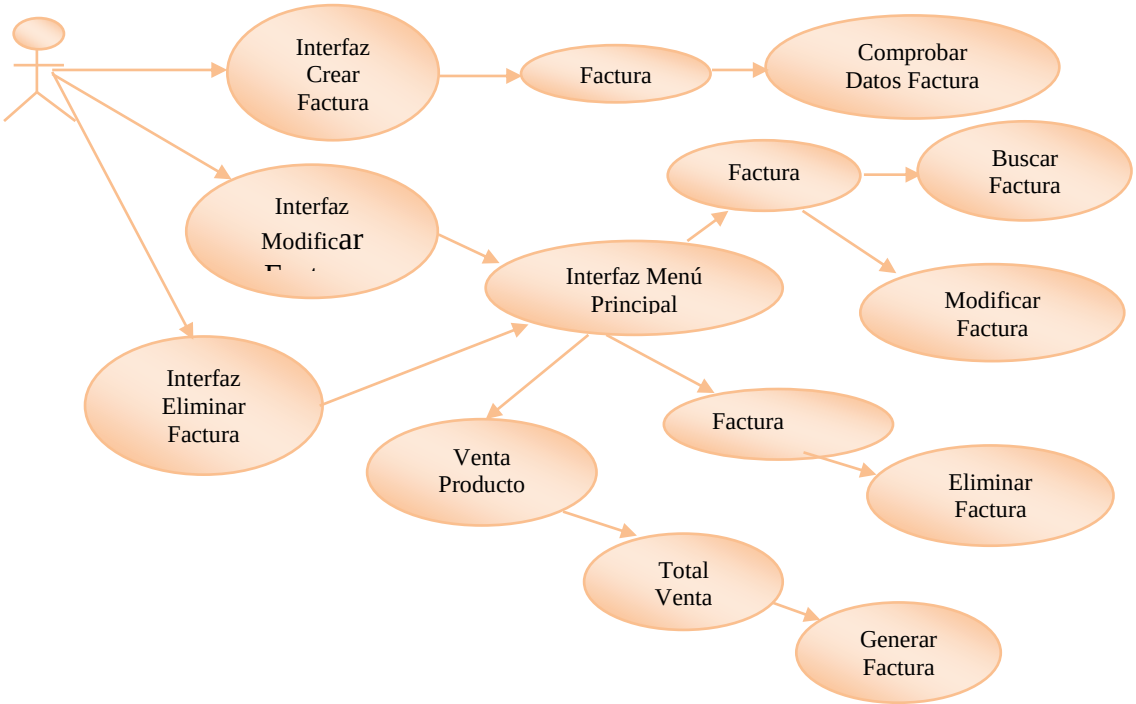


Figura 13. Caso de Uso Crear – Buscar- Modificar – Eliminar Factura

4.21.7. Caso de Uso Crear – Buscar- Modificar – Eliminar Cuenta (RF07).

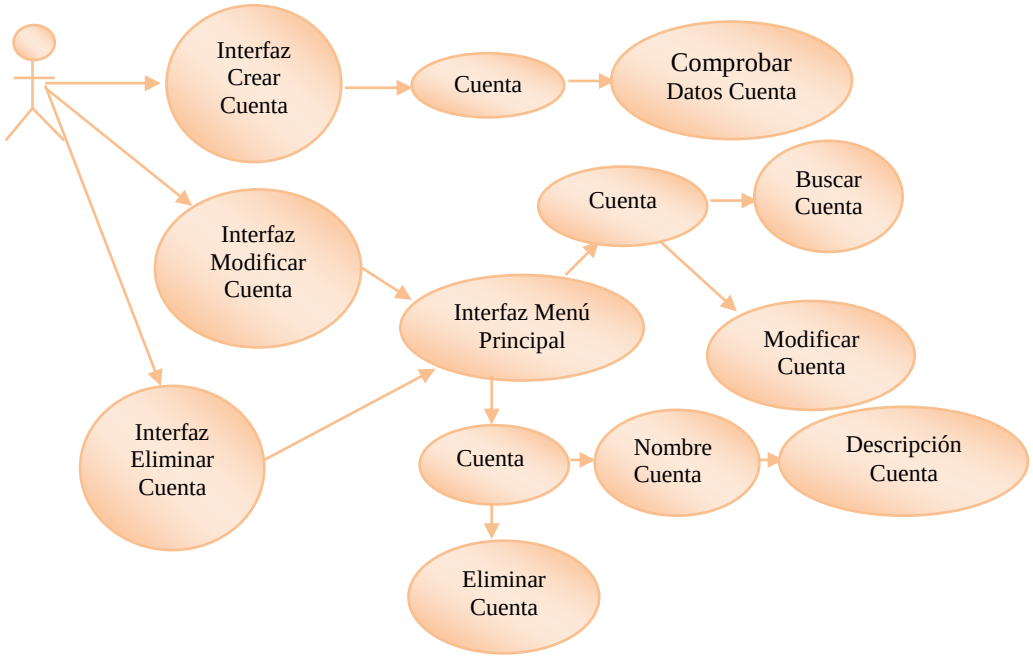


Figura 14. Caso de uso crear – buscar- modificar – eliminar Cuenta

4.21.8. Caso de Uso Crear – Buscar- Modificar – Eliminar Cuenta Banco (RF08).

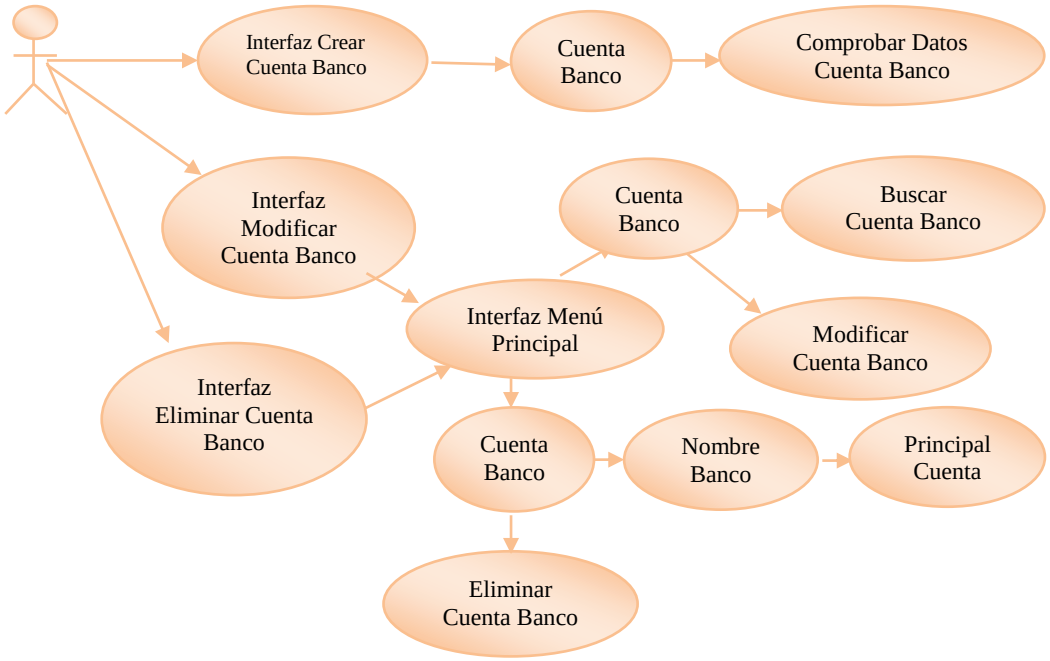


Figura 15. Caso de Uso Crear – Buscar- Modificar – Eliminar Cuenta Banco

4.21.9. Caso de Uso Crear – Buscar- Modificar – Eliminar Cheque (RF09).

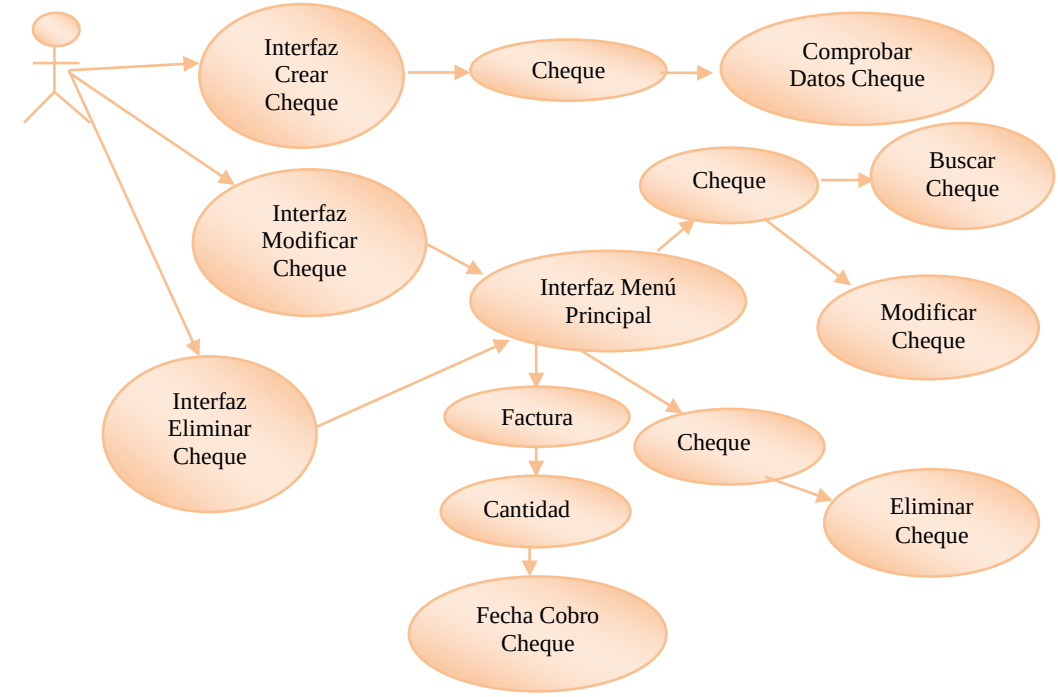


Figura 16. Caso de uso crear – buscar- modificar – eliminar Cheque

4.21.10. Caso de Uso Crear – Buscar- Modificar – Eliminar Subcuenta (RF10).

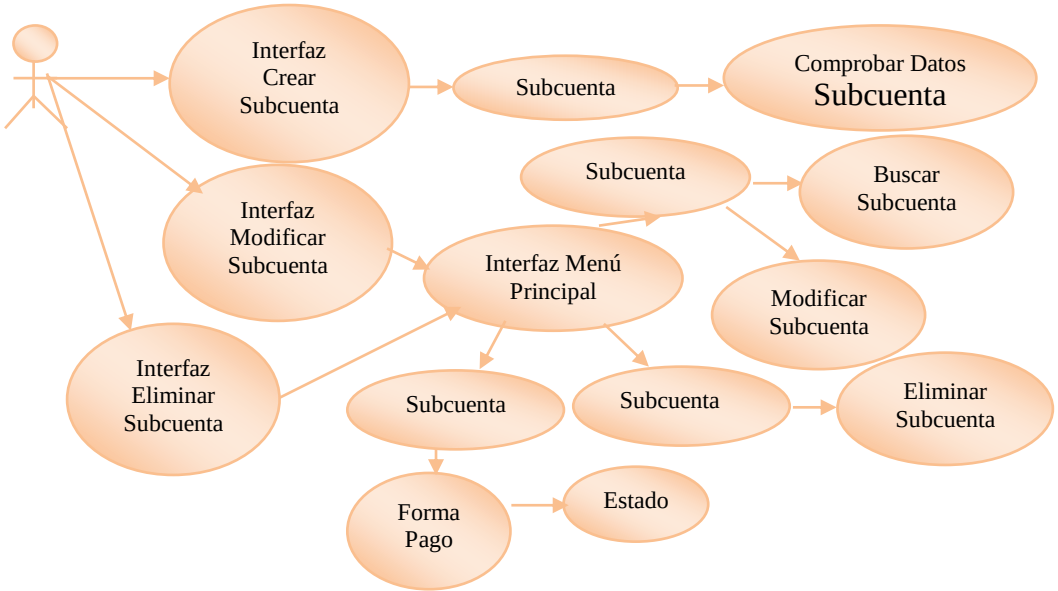


Figura 17. Caso de Uso Crear – Buscar- Modificar – Eliminar Subcuenta

4.21.11. Caso de Uso Crear – Buscar- Modificar – Eliminar SCC (RF11).

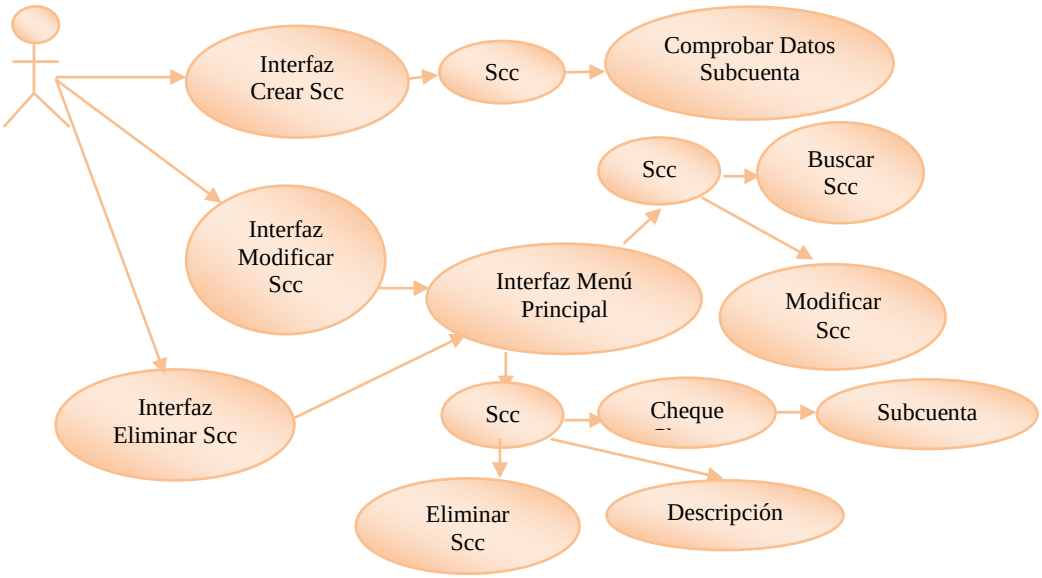


Figura 18. Caso de uso crear – buscar- modificar – Eliminar SCC

4.22. Gestión de Configuración.

A continuación se detalla lo concerniente a la administración de configuración de software Nachos Sport que engloba todo lo necesario para gestionar la integridad de los productos software implementados durante el desarrollo del proyecto.

Las actividades relacionadas con el Nachos Sport para este proyecto son:

- Identificación de la Configuración.
- Control de la Configuración.

4.23. Gestión de Aseguramiento de Calidad.

Las políticas de calidad están basadas en asegurar que, el producto final satisfaga los requisitos y que durante el desarrollo se adapte a los cambios que puedan producirse en los requerimientos.

Para llevar a cabo el control de la calidad se ejecutaran revisiones y pruebas de software.

4.24. Identificación de las Propiedades de Calidad.

Las propiedades de calidad para el sistema Nachos Sport pueden resumirse en los siguientes puntos:

- **Corrección:** El sistema debe satisfacer las especificaciones de requisitos establecidas. Se verificará el número de requisitos entregados con el producto contra los definidos.
- **Facilidad de Uso:** Operatoria simple e intuitiva por parte del usuario.
- **Seguridad:** Acceso a la funcionalidad del sistema únicamente a través de la combinación de nombre de usuario y clave.
- **Perfomance:** Tiempo de respuesta conforme a las necesidades del cliente o usuario.
- **Facilidad de mantenimiento:** Se establecerá bajo el análisis de la documentación, buscando la correspondencia entre lo escrito y el código ejecutable.

4.25. Revisiones.

Para todos los productos se debe realizar una revisión puntual antes de realizar la incorporación a la línea base (ya sea como consecuencia de cambios realizados en un documento existente en línea base, o durante su primera inserción en la misma) a efectos de que se cumplan las condiciones de calidad establecidas.

4.26. Catálogo de Requisitos.

Mediante una lista de preguntas de verificación se comprueba la precisión y completitud de los requisitos.

- ¿Los requisitos son entendibles?
- ¿Son factibles de implementar?
- ¿Existen requisitos superpuestos o que se complementan?
- ¿Son importantes para solucionar los problemas detectados?
- ¿Falta especificar algún requisito?

4.27. Consistencia entre productos del Análisis.

En esta actividad se controla que se haya realizado la validación y verificación de los productos obtenidos durante la fase de análisis. La lista de verificación es:

- ¿Todos los requisitos tienen asociado un caso de uso?
- ¿Los casos de uso tienen su descripción completa?
- ¿Los casos de uso tienen su correspondiente clase de análisis?
- ¿Los casos de uso tienen definido su diagrama de colaboración correspondiente?

4.28. Revisiones definidas para la etapa de Diseño.

Bajo esta actividad se controla que la arquitectura del sistema responda a los requisitos establecidos. La verificación se realiza mediante la lista de preguntas que siguen a continuación:

- ¿Los casos de uso tienen su correspondiente diagrama de clases de análisis y de diseño?
- ¿Las clases de diseño tienen su correspondiente descripción?
- ¿Para cada requisito funcional existe su correspondiente caso de uso, diagrama de clase de diseño y su descripción?

4.29. Revisiones definidas para la etapa de las Pruebas.

Para esta actividad se definen las siguientes cuestiones:

- ¿Son probados todos los requisitos?
- ¿Se verifican los límites de rango?
- ¿Se realiza la prueba de interfaz entre cada módulo?
- ¿Han sido verificados los valores representativos de las clases?
- ¿Los casos de prueba verifican todos los procesos?

4.30. Pruebas de Software.

Las pruebas que se necesitan desarrollar comprenden los siguientes tipos:

- Pruebas Unitarias: son las realizadas sobre cada modulo en forma independiente.
- Pruebas de Integración: son las realizadas en forma conjunta con los módulos que han pasado las Pruebas Unitarias.
- Pruebas del Sistema: Se prueba el sistema como un todo, verificando que se cumplan los requisitos de calidad estipulados.

4.31. ESTRUCTURA DE LA BASE DE DATOS.

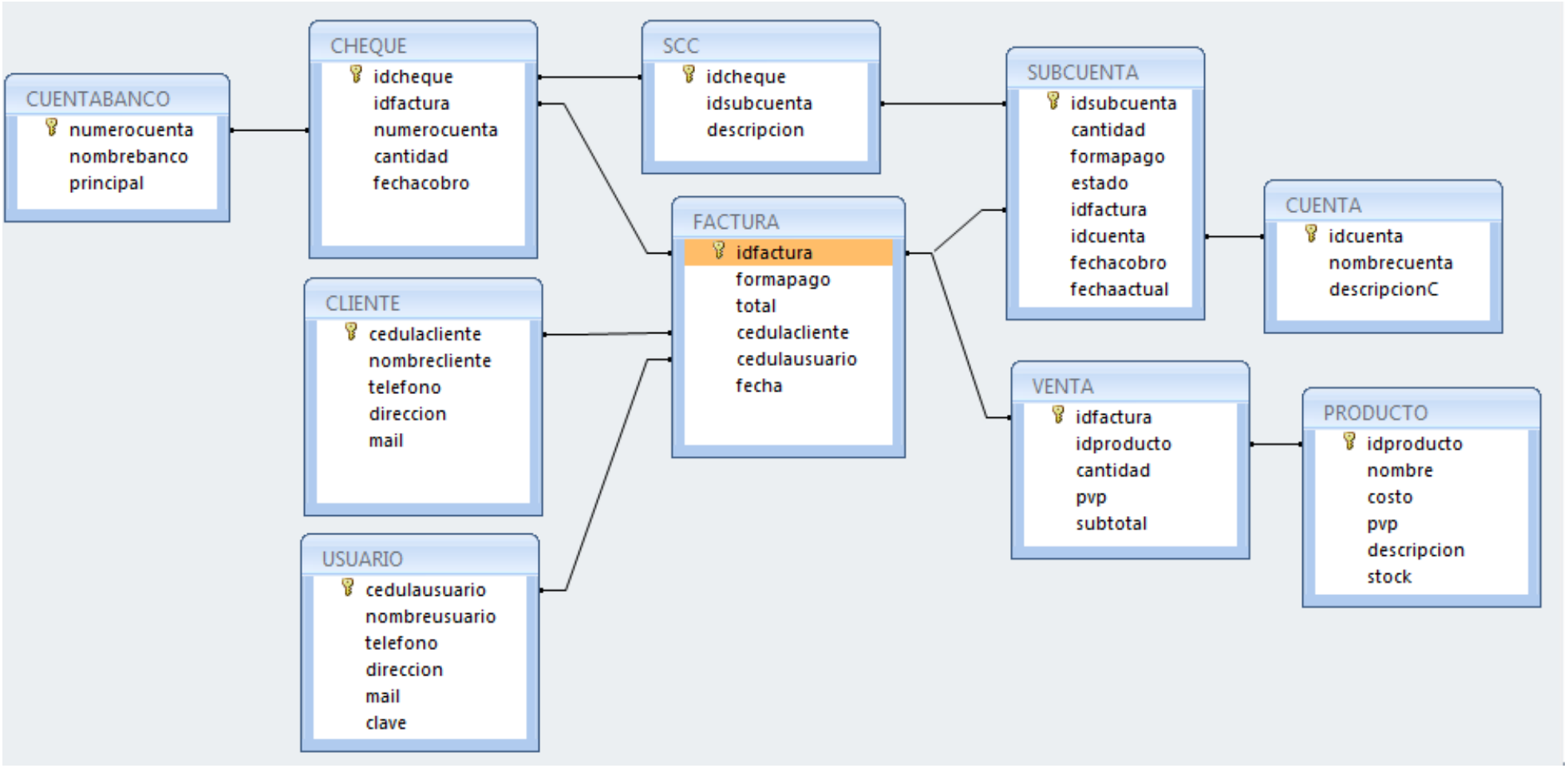


Figura 19. Estructura de la base de datos.

CAPÍTULO VI

5.1. METODOLOGÍA.

El estudio se basa en la investigación cualitativa, cuyo enfoque principal es la producción de conocimiento que permite entender y explicar el mundo y los fenómenos sociales por su contexto propio de productor de conocimiento, investigación de campo, participación de los involucrados, observaciones en el área misma. La investigación cualitativa es uno de los factores que posibilitan el uso de nuevas perspectivas en el desarrollo de investigaciones en el ámbito de los sistemas de información. En esta tesis se trabaja con datos cualitativos, cuyas fuentes incluyen documentos y textos (artículos y libros), cuestionarios, observación, recolección digital de información, entrevistas a propietarios de las fábricas, visitas a los productores y participación.

5.2. Tipo de Estudio.

Campo.- Se efectuó en el lugar y tiempo real en que ocurren los fenómenos objetos de estudios. Se refiere a un estudio en una situación real, que al efectuarse en condiciones no artificiales permite detectar mejor las posibles consecuencias de la investigación.
Las técnicas empleadas son:

5.2.1. Observación Descriptiva.

Permite conocer la realidad del sondeo, la visualización del lugar del proyecto, la distribución de la Microempresa, diagramas de flujo de procesamiento, tipo de almacenamiento de los productos, sistemas de comercialización, etc.; Para concretar acciones que se ejecutan en el trabajo.

5.3. POBLACIÓN Y MUESTRA

5.3.1. Población:

El universo de esta investigación serán quienes integran la Empresa Directivos y trabajadores.

5.3.2. Muestra:

5.3.2.1. Muestra Directa:

Las personas involucradas directamente con el Sistema son quienes forman parte de la Empresa.

Población: 25 Persona (Empresa Nachos Sport).

Muestra: Estimado 6 personas que utilizaran el sistema.

5.4. Procedimientos.

5.4.1. Fuentes de Información.

Entre las fuentes de información consta la Primaria y Secundaria

a) **Primarias**.- Esta información se obtendrá basándose en la Observación y Conversación con el gerente de la empresa, trabajadores.

b) **Secundarias**.- Las fuentes secundarias se obtendrá de folletos, revistas, trípticos relativos al tema, así como del Internet.

5.5. Procesamiento y Análisis.

5.5.1. Teoría fundamentada en datos.

La teoría fundamentada en datos es un método de investigación cualitativa que ayuda en la colecta, análisis sistemático de datos y en la generación de la teoría.
En el desarrollo de esta tesis este método se ha utilizado para precisar la colecta y el análisis general de los datos pertinentes a su ordenación en cuanto a los criterios económicos, técnicos y en cuanto al análisis de datos.

5.5.2. Análisis de tareas:

En este proceso se describirá las tareas realizadas actualmente por los usuarios, sus patrones definidos de flujo de trabajo, los cuales se originan de sus esquemas mentales y las necesidades de información para realizar su trabajo. Es decir, se procura identificar "qué el usuario hace", "de qué manera lo hace", y "qué necesita para hacerlo". De esa manera, se logra el entendimiento conceptual de las tareas que deberán formar parte del sistema en desarrollo. Para la obtención de dicho entendimiento se pueden utilizar varias técnicas tales como entrevistas, observación sistemática, etc.

CAPITULO VI

6.1. ANÁLISIS DE RESULTADOS.

Se ha desarrollado un sistema de facturación de diseño modular y en capas el mismo que permitirá a los usuarios trabajar con la información de la empresa para poder llevar un control sobre las ventas realizadas y la facturación de las mismas.

Será una aplicación ejecutable bajo el sistema operativo Microsoft Windows y será testeado para las versiones XP, Vista, Windows7. La misma será desarrollada con herramientas de programación software Libre, lenguajes que soportan orientación a objetos y que además poseen una gran ductilidad en el manejo de datos.

El producto final permitirá al usuario manipular información de clientes productos, cuentas por cobrar, cuentas de Banco. Además se proveerá de un esquema que facilitará la administración. El sistema contará con ayudas en línea y un asistente.

6.2. Elección de herramienta de Estudio.

Después del estudio de las herramientas de software libre se han seleccionado dos herramientas eficientes como son Php y Java se ha realizado un estudio profundo, lo cual no ayudo a elegir a Php como la herramienta de estudio para nuestro caso aplicativo ya que nuestro tema central está ligado en el costo de desarrollo de la aplicación.

6.3. Cuadro comparativo de Tecnologías de estudio.

7. Tabla 79. Comparación Php – Java.

Comparación Tecnologías	PHP	JAVA
Modularización		✓
Mantenibilidad	✓	✓
Coste de desarrollo.	✓	
Integración externa.		✓
Seguridad		✓
Rendimiento	✓	
Escalabilidad	✓	✓

6.4 Metodologías de Estimación.

6.4.1 Análisis de Puntos Función.

El Análisis de Punto Función es una técnica que, mediante la descomposición de un sistema en componentes más pequeños, permite que éstos puedan ser mejor comprendidos y analizados en forma individual.

El Análisis de Punto Función se basa en la teoría de que las funciones de una aplicación son la mejor medida del tamaño de un sistema. El Punto Función mide el software mediante la cuantificación de la funcionalidad que el sistema le brinda al usuario basado fundamentalmente en el diseño lógico. Es independiente del lenguaje de computación, de la metodología de desarrollo, de la tecnología utilizada y de la capacidad del equipo de trabajo para desarrollar la aplicación.

5.4.2. Metodología escogido para la estimación de costos.

En el caso de la estimación de costos se escogió como metodología COCOMO II, aunque esta es un tanto complicada, debido a la utilización de varias fórmulas que estiman el costo de un proyecto, cuenta con la ventaja de usar para su estimación un amplio dominio de factores que inciden sobre el costo del proyecto, tales como: retrasos en el cronograma, pérdida de personal, etc.

5.5. Comprobación de Hipótesis

5.5.1. Hipótesis

El uso de tecnologías Open Source incide en el costo del desarrollo de aplicaciones web.

5.5.2. Comprobación

Para la comprobación de la hipótesis descrita anteriormente se realizó estudios comparativos entre herramientas de software libre y propietario, con las cuales se pudo medir el costo de desarrollo de las aplicaciones web, quedando demostrada nuestra hipótesis como afirmativa.

Tabla 80. Comprobación de la hipótesis.

Variable Independiente	Variable Dependiente
El uso tecnologías Open Source	**Costo de desarrollo de aplicaciones web**
PHP	**$ 2,400**
JAVA	**$ 3.200**
ASP	**$ 6,320**

Se puede observar claramente que el costo de desarrollo de software depende de la herramienta que se utilice para su desarrollo.

5.6. CONCLUSIONES Y RECOMENDACIONES

5.6.1. Conclusiones.

- Al realizar el estudio comparativo de las herramientas se software libre para el desarrollo de aplicaciones web se concluye que PHP en un 55% es el lenguaje de programación idóneo para la realización de la aplicación ya que considera aspectos trascendentales como: modularización, mantenibilidad, costo de desarrollo, integración externa, seguridad, rendimiento y escalabilidad.

- Los factores que interviene en forma directa en el incremento o disminución en el costo de desarrollo del sistema son: tamaño del sistema, líneas de código fuente, plataforma, personal y tiempo del proyecto según lo establecido en el modelo COCOMO II.

- El uso de herramientas de software libre para la implementación de sistemas para pymes el costo de desarrollo disminuye de manera notoria y la recuperación de la inversión para la empresa se da en menor tiempo como se demuestra en el cálculo del ROI del 66% palpando una recuperación en menos de 2 años.

- Para realizar una estimación se debe identificar los componentes funcionales como son las entradas, salidas, consultas y los archivos lógicos, relacionarlos según los valores del factor complejidad asignadas por el modelo COCOMO II y multiplicarlos según la tabla de conversión de UFP a SLOC dando como resultado el total de puntos de función.

5.6.2. Recomendaciones

- Para la estimación de costos en herramienta Open Source se debe considerar el uso del modelo COCOMO II ya que permite estimar el costo, esfuerzo y tiempo de duración de un proyecto de software, siendo de a poyo para las pymes ya que permiten la planificación, prevención, preparación, replanificación y seguimiento de las funciones se va a realizar.

- Se recomienda la utilización de los puntos de función para la estimación de costos por que es un método que permite la flexibilidad, funcionalidad que presenta en etapas tempranas de desarrollo en donde no es mucho lo que se conoce con respecto a las características del proyecto y por la independencia de lenguaje de programación que se utilice.

- Es necesario poner énfasis en la definición de requerimientos funcionales ya que es el punto de partida para el desarrollo del mismo y la base fundamental para el costo de desarrollo, se este modo los resultados obtenidos tendrán mayor exactitud ya que la magnitud del sistema estará totalmente definida.

- La capacitación en las herramientas escogidas para el desarrollo de la aplicación es importante ya que la experiencia del personal es uno de los factores que afectan al tiempo de desarrollo del sistema como lo planteado en el modelo COCOCMO II en la fase de multiplicador de esfuerzo.

5.6.3. GLOSARIO.

PHP.- Es un acrónimo recursivo que significa PHP Hypertext Pre-processor (inicialmente PHP Tools, o, Personal Home Page Tools).

POSTGRESQL.- PGDG (PostgreSQL Global Development Group).

APLICACIONES.- Nombre que reciben los programas especializados en tareas concretas y de una cierta complejidad.

GNL.- Licencia Pública General

AGPL.- Licencia Pública General de Affero

JSP.- Es un lenguaje para la creación de sitios web dinámicos, acrónimo de Java Server

LDC.- Líneas de Código

PF.-Puntos de Función

CT.- Codificación y Testeo de Unidades

IT.- Integración y Testeo

PMnominal. – PM nominal es el esfuerzo nominal requerido en meses-hombre.

PREC.- Experiencia en este tipo de desarrollos.

FLEX.- Flex

RSEL.- Relacionado con la arquitectura y la mitigación de riesgos.

TEAM.- Cohesión y madurez del equipo de desarrollo.

PMAT.- Proceso de madurez de desarrollo del software.

RELY.- Confiabilidad requerida del software

DATA.- Tamaño de la base de datos.

CPLX.- Complejidad del Producto

RUSE.- Reusabilidad del Código

DOCU.- Requerimiento de documentación

TIME.- Restricciones de tiempo de ejecución

STOR.- Restricciones de almacenamiento

PVOL.- Volatilidad de la plataforma

ACAP.- Capacidad de los analistas

PCAP.- Capacidad de los programadores

AEXP.- Experiencia en aplicaciones

PEXP.- Experiencia en plataforma

LTEX.- Experiencia el lenguaje y herramienta de desarrollo

PCON.- Continuidad del personal

TOOL.- Uso de la herramienta de software

SITE.- Desarrollo en múltiples ubicaciones

SCED.- Restricciones de calendario

DM.- Este riesgo significa la dificultad de implementar algunas de las funcionalidades exigidas a través del entorno de desarrollo elegido.

5.7. BIBLIOGRAFIA.

http://www.liderdeproyecto.com/articulos/estimacion_costos_de_software.html

http://www.cimat.mx/Eventos/setys2009/jfcastillo.pdf

http://es.scribd.com/doc/32166526/Ingenieria-de-Software-Pressman-capitulos-1-9

http://www.slideshare.net/yandry2010/metodologa-espiral

http://www.slideshare.net/equipo2/cocomo-ii

http://trevinca.ei.uvigo.es/~cfajardo/Nueva_carpeta/presentaciones/cocomo2k.pdf

PRESSMAN, R; Ingeniería de Software, un enfoque práctico, Quinta Edición 2000.

RUMBAUGH, J; Modelado y Diseñado Orientado a Objetos, Metodología RUP; Santa Fe Bogotá; Pretice Hall 2001.

Cultural S.A. Curso de Informática Personal; Madrid España: Cultural; 1999.

5.8. ANEXOS.

INSTALACIÓN

DEL SISTEMA

INDICE GENERAL

INDICE DE FIGURAS

1. Descripción de Xamp.

XAMPP es una fuente libre y abierto multiplataforma servidor web paquete de soluciones de pila, que consiste principalmente en el servidor HTTP Apache, base de datos MySQL, y los intérpretes de scripts escritos en el PHP y los lenguajes de programación Perl.

2. Requerimientos De Xampp

- Sistema Operativo: Windows XP, GNU Linux.
- Software de la Terminal: Navegador web (Mozilla Firefox o InternetExplorer).

3. Instalación de xampp.

Nota: Antes de instalar un servidor de páginas web es conveniente comprobar si no hay ya uno instalado. Para ello, es suficiente con abrir el navegador y escribir la dirección http://localhost. Si no se obtiene un mensaje de error es que hay algún servidor de páginas web instalado.

La última versión de XAMPP disponible actualmente (abril de 2011) es la versión 1.7.4 (del 22 de enero de 2011), que incluye Apache 2.2.17, PHP 5.3.5, MySQL 5.5.8 y otras utilidades y la versión para Windows. Una vez obtenido el archivo de instalación de XAMPP, hay que hacer doble clic sobre él para ponerlo en marcha. Las imágenes que se muestran a continuación corresponden a la instalación de XAMPP 1.7.4 en Windows XP.

La primera pantalla permite elegir el idioma de instalación, entre los que no se encuentra el español. Para empezar la instalación, hay que hacer clic en el botón "OK".

Figura 20. Seleccionar Idioma.

A continuación se inicia el asistente de instalación. Para continuar, hay que hacer clic en el botón "Next".

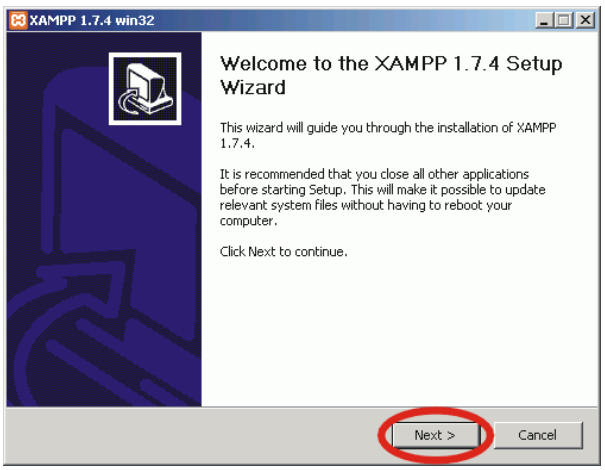


Figura 21. Asistente de Instalación.

En la siguiente pantalla se puede elegir la carpeta de instalación de XAMPP. La carpeta de instalación predeterminada es C:\xampp. Si se quiere cambiar, hay que hacer clic en "Browse..." y seleccionar la carpeta donde se quiere instalar XAMPP. Para continuar la configuración de la instalación, hay que hacer clic en el botón "Next".

Nota: En los ordenadores de clase hay que tener en cuenta la unidad en la que se quiere instalar XAMPP (C:\ o D:\), ya que por omisión se instala en la unidad C:\.

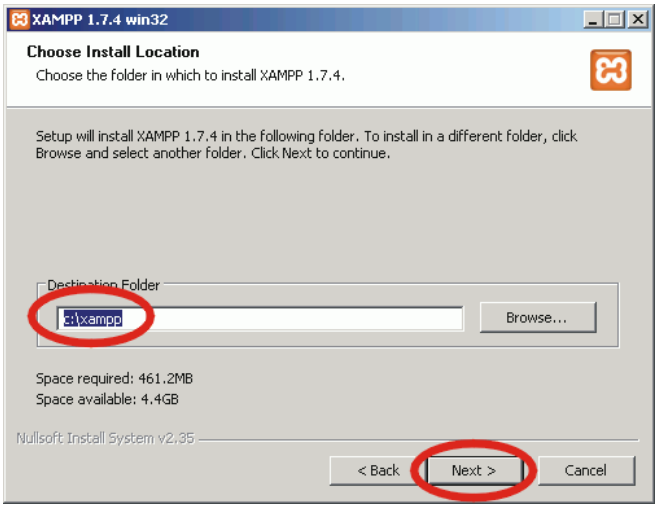


Figura 22. Ubicación de la Instalación.

En la pantalla siguiente se puede configurar XAMPP como servicio, para que se inicie cada vez que se inicie Windows. En este curso se recomienda instalar tanto Apache como MySQL. Para completar la configuración de la instalación e iniciar la copia de archivos, hay que hacer clic en el botón "Install".

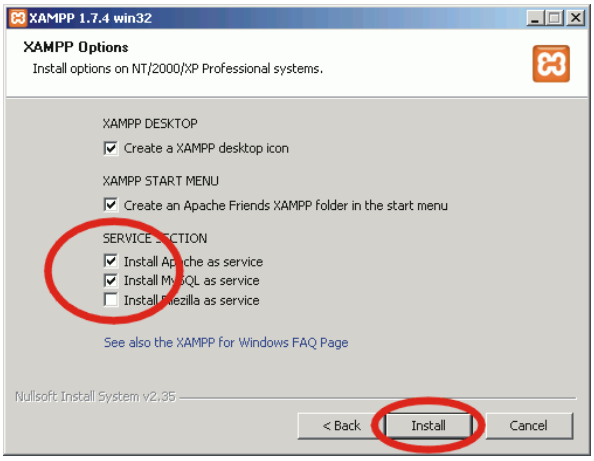


Figura 23.Opciones de Xampp.

A continuación, se inicia el proceso de copia de archivos, que puede durar unos minutos.

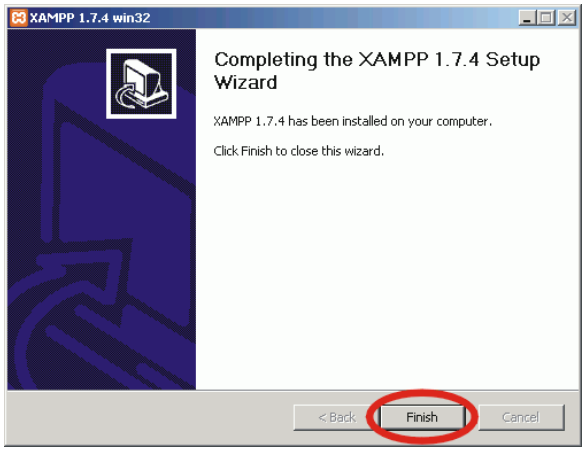


Figura 24. Instalación Completa.

Una vez terminada la copia de archivos, se muestra la pantalla que confirma que XAMPP ha sido instalado. Hay que hacer clic en el botón "Finish".

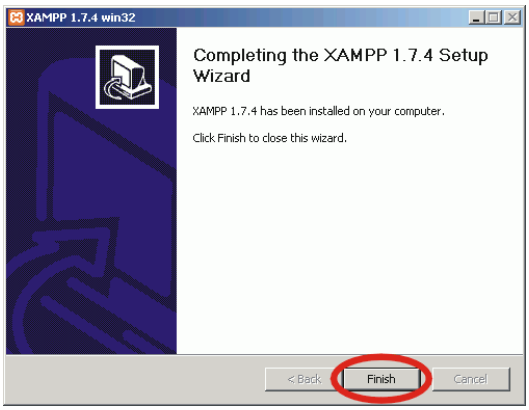


Figura 25. Finalizar Instalación.

Comienza entonces la instalación de los servicios. Al iniciarse el servidor Apache por primera vez, el cortafuegos de Windows muestra un alerta de seguridad para que indiquemos si debe bloquearse el puerto 80 utilizado por el servidor.

Es necesario hacer clic en "Desbloquear" para poder acceder a páginas web en el servidor.

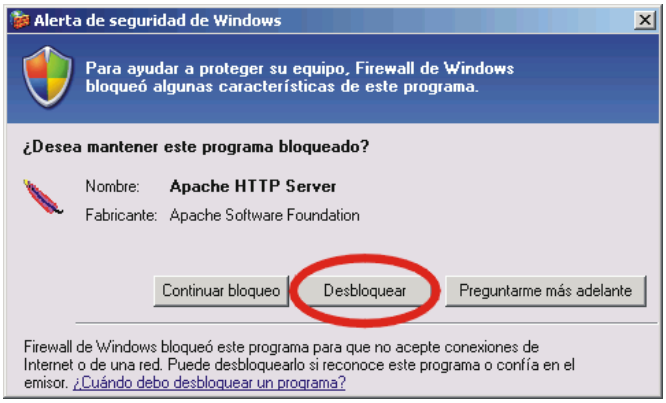


Figura 26. Cortafuegos de Windows.

Cuando se termina la instalación de los servicios, se muestra una ventana que confirma que los servicios han sido instalados. Hay que hacer clic en el botón "Aceptar".

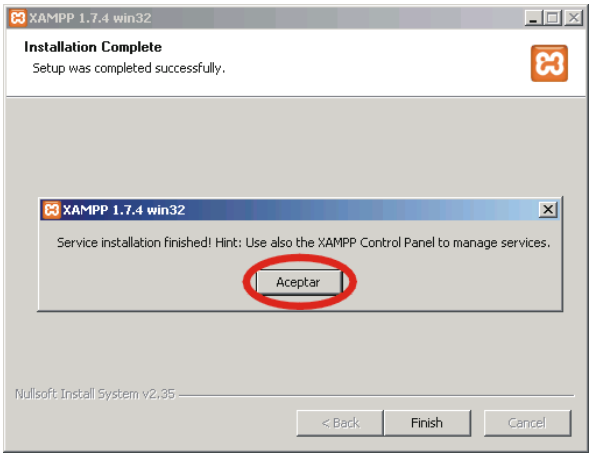


Figura 27. Servicios Instalados correctamente.

Finalmente, se termina la instalación y se da la posibilidad de abrir el panel de control de XAMPP. Para arbirlo y comprobar los servcicios instalados, hay que hacer clic en "Sí".

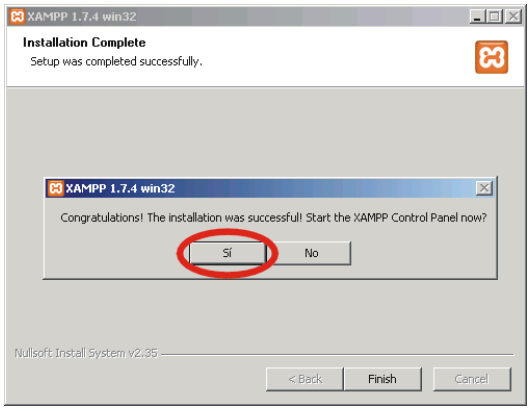


Figura 28. Servicios en función correcta

El panel de control de XAMPP muestra los servicios instalados.

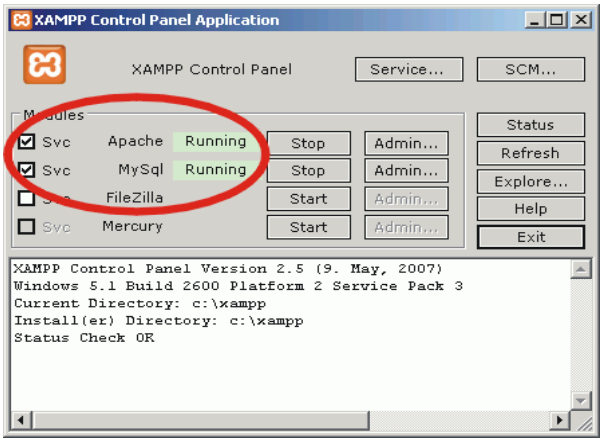


Figura 29. Servicios Instalados.

Para comprobar que todo funciona correctamente, hay que escribir en el navegador la dirección "http://localhost". Al abrir la página por primera vez, XAMPP pedirá seleccionar el idioma:

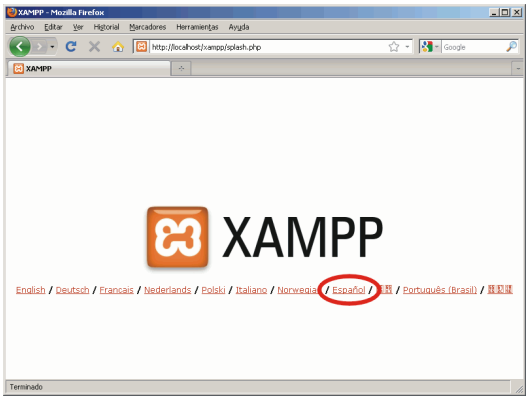


Figura 30. Pantalla principal de Xampp.

Una vez elegido el idioma, se mostrará la página de configuración de XAMPP:

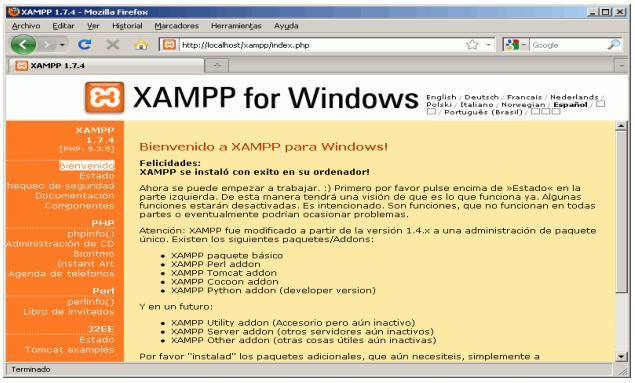


Figura 31. Pagina de configuración de Xampp.

4. Panel de Control de XAMPP.

Al panel de control de XAMPP se puede acceder mediante el acceso directo del escritorio, el menú "Inicio > Programas > XAMPP for Windows > XAMPP Control Panel" o, si ya está iniciado, mediante el icono del área de notificación.

El panel de control permite detener o reiniciar los servidores y los servicios:

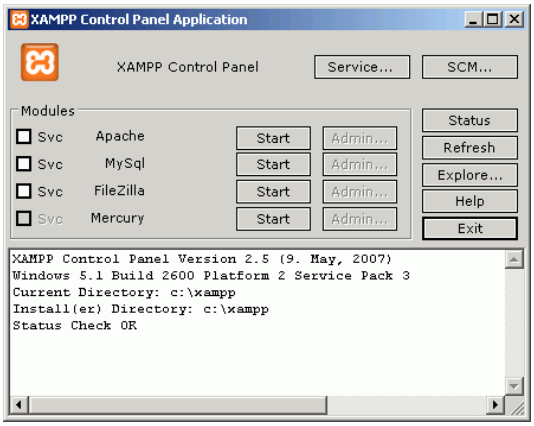


Figura 32. Detener o reiniciar los servicios.

Si queremos que Apache o MySql arranquen como servicio, es decir, que se pongan en marcha cada vez que arrancamos el ordenador, hay que marcar las casillas Svc correspondientes (XAMPP solicita confirmación).

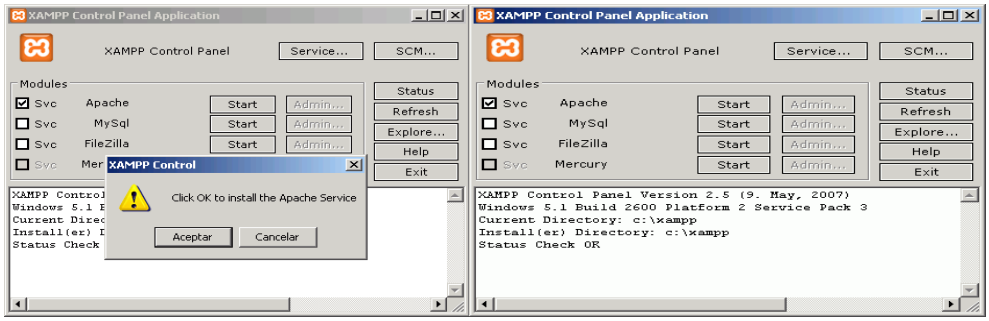


Figura 33. Marcar casillas indicadas.

Para que Apache o MySql dejen de arrancar como servicio, hay que desmarcar las casillas Svc correspondientes (XAMPP solicita confirmación). Para poder desmarcar una casilla el servidor correspondiente debe estar detenido.

Para poner en marcha Apache o MySql, se debe hacer clic en el botón Start correspondiente.

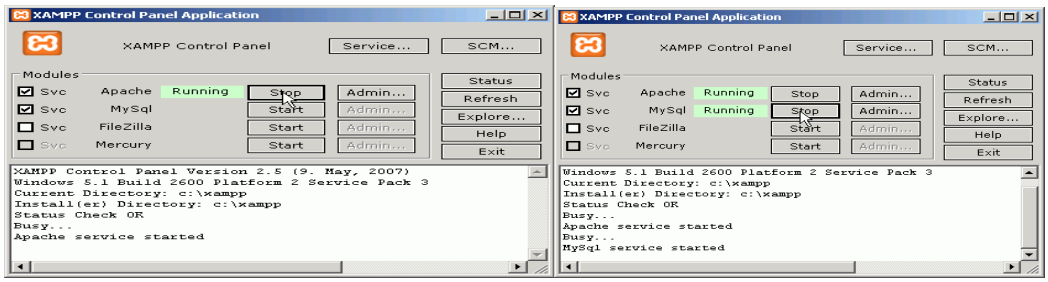


Figura 34. Activar Php y My Sql.

Para detener Apache o MySql se debe hacer clic en el botón Stop correspondiente.

Nota: En algunos ordenadores, cuando el servicio de Apache está activado (casilla Svc marcada), si se detiene Apache y se reinicia, el letrero verde "Running" tarda un poco en mostrarse o no se muestra nunca. La solución en esos casos es detener Apache, desmarcar la casilla Svc, marcarla de nuevo la casilla Svc y reiniciar Apache.

5. Instrucciones de Instalación de NetBeans.

Este documento describe como instalar NetBeans en su sistema. Para información acerca de los sistemas operativos soportados y configuraciones de hardware.

6. Para instalar el software:

1. Después de completarse la descarga, ejecute el instalador.

    - Para Windows, el archivo ejecutable de instalación tiene extensión `.exe`. Haga doble click en el archivo de instalación para ejecutarlo.
    - Para plataformas Solaris y Linux, el archivo de instalación tiene extensión `.sh`. Para estas plataformas, usted necesita convertir los archivos de instalación en archivos ejecutables usando el siguiente comando: chmod +x ./<*nombre-de-archivo-instalador*>

2. Para seleccionar las herramientas y rutinas a instalar, realice los siguientes pasos en la página de Bienvenida (Welcome) del asistente de instalación:

    - Haga click en Personalizar (Customize).
    - En el cuadro de diálogo Personalizar, haga sus selecciones.
    - Haga click en OK

En la página de instalación de NetBeans IDE, haga lo siguiente:

1. Acepte el directorio de instalación predeterminado para NetBeans IDE o especifique otro directorio.
   Nota: El directorio de instalación debe estar vacío, y el perfil de usuario que usted está usando debe tener permiso de lectura/escritura para este directorio.

2. Acepte la instalación predeterminada JDK para utilizar con NetBeans IDE o seleccione una instalación diferente desde la lista desplegable.
   Si el asistente de instalación no halla una instalación JDK compatible para su uso con NetBeans IDE, su JDK no se encuentra instalado en el directorio por defecto. En ese caso, especifique el directorio de instalación JDK o cancele la instalación en curso e instale la versión JDK necesaria y reinicie la instalación de NetBeans IDE.
3. Haga click en Next.
4. Si se abre la página de instalación del servidor GlassFish, haga lo siguiente:
5. Acepte el directorio de instalación predeterminado para el servidor o especifique otro directorio de instalación.
   Nota: El directorio de instalación que usted especifique debe estar vacío, y el perfil de usuario que usted está usando debe tener permiso de lectura/escritura para este directorio.
6. Desde la lista de instalaciones compatibles JDK, elija la JDK que quiera que su servidor utilice o haga click en el botón Explorar (Browse) para especificar otra instalación JDK.
7. Cambie el nombre de usuario y contraseña para el dominio del servidor o acepte los que aparecen por defecto y haga click en Next.
   El nombre de usuario y la contraseña que aparecen por defecto son admin y adminadmin respectivamente.
8. Verifique los valores predeterminados de los puertos (`HTTP`, `HTTPS` y `Admin`) para el servidor o cámbielos si es necesario.
9. Haga click en Next.

Si se abre la página de instalación de la aplicación Apache Tomcat, acepte el directorio de instalación predeterminado o especifique otro directorio de instalación.

Manual de Usuario Nachos Sport

Índice General

Índice Figuras.

Manual de Usuario

Detalle General del Sistema.

Principales Características

Nachos Sport es un sistema de facturación muy fácil de manejarlo, su increíble facilidad de operación, con pantallas sencillas permiten que cualquier usuario, sin conocimientos previos de computación, comience a utilizar el sistema apenas este haya sido instalado.

Debido a su variedad de funciones, **Nachos Sport** se adapta a los más diversos Rubros, entre ellos:

- Toda clase de Comercios Minoristas
- Toda clase de Comercios Mayoristas
- Depósitos
- Distribuidoras
- Fábricas
- Venta de Servicios

Una vez que haya leído este manual aprenderá a explotar las características del sistema, entre las que encontrará:

- Poderoso manejo de Cuentas Corrientes de Clientes y Proveedores
- Control Stock de productos, multidepósito, totalmente automatizado y configurable
- Venta de Servicios
- Emisión de Presupuestos, Remitos, Recibos, Ordenes de pago, etc.
- Además, el sistema tiene soporte para distintos dispositivos electrónicos que puedan esta conectados a la computadora.

Pantalla de inicio

Ilustramos interfaz de inicio del sistema.

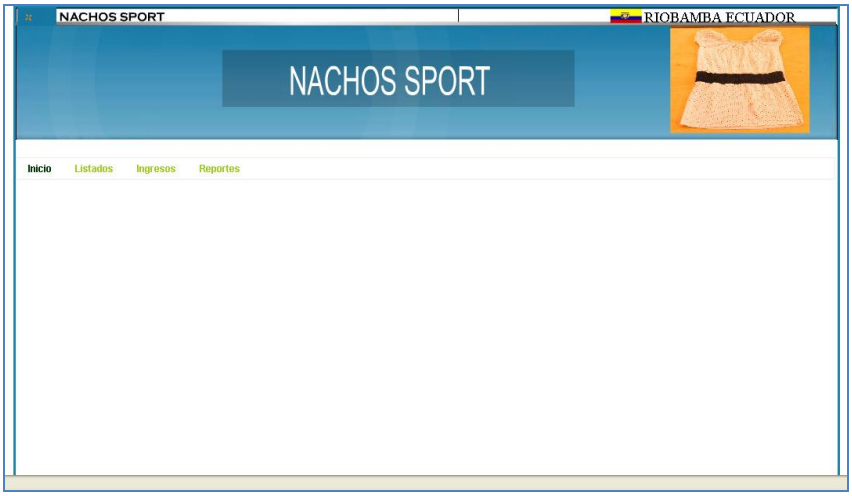


Figura 35. Pantalla de inicio

Listados Clientes

El sistema permitira listar los nuevos clientes.

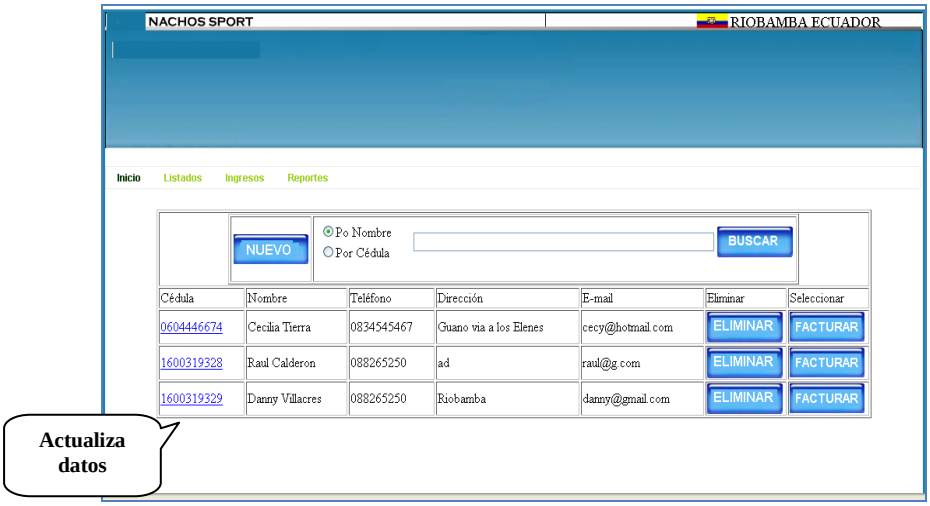


Figura 36. Listados Clientes

En esta plantilla se listara los clientes existentes en el sistema, se podrá realizar ingreso de nuevos clientes, búsquedas (nombre, cedula), eliminación (nombre, cedula), al mismo tiempo permitirá generar facturas.

Con dar clic en el número de cedula que esta en la parte izquierda del sistema con letra de color azul, subrayado (**0604446474**), permitirá la actualización de los datos del cliente.

Generar Factura

El sistema permitirá generar facturas cada vez que el usuario solicite.

Figura 37. Factura.

La Interfaz de la factura tendrá los siguientes campos.

- La Factura tendrá un encabezado con los datos de la empresa.
- Ruc del gerente
- Número de la Factura que se genera automáticamente.
- Código autorizado del SRI.
- Lugar y Fecha de emisión.
- Fecha de emisión de la factura.
- Nombre del cliente.
- Ruc del Cliente.
- Dirección del Cliente.
- Teléfono Cliente.
- Cantidad Productos comprados.
- Descripción de productos.
- Valor Unitario de cada producto.
- Valor Total
- Sub Total
- Grabado Iva tarifa 12%
- Importe de Iva
- Total.

Listados Cuenta Bancos

Este sistema permitirá listar las cuentas de los bancos.

Figura 38. Listados Cuenta

En esta pantalla se listara la cuenta de los bancos existentes, se realizara el ingreso de una nueva cuenta, la búsqueda mediante (principal cuenta, número cuenta), eliminación (principal cuenta, número cuenta).

Con dar clic en el número que se encuentra en la parte izquierda del sistema que esta con letra de color azul y subrayado (**3701011593253**), se podrá actualizar la cuenta bancos.

Listados Productos

El sistema permitirá listar los productos.

Figura 39. Listados Productos.

En esta plantilla se listara los productos existentes en el sistema, permitirá el ingreso de un nuevo producto, búsqueda mediante (nombre, código), eliminación (nombre, código).

Con dar clic en el producto que se encuentra en la parte izquierda que esta con letra de color azul, subrayado (**camisetas**), se podrá actualizar el producto.

Listados Cuentas

El sistema permitira listar la cuentas existentes en el sistema.

Figura 40. Listados Cuenta

En esta plantilla se permitira el ingreso de una nueva cuenta, busqueda mediante(nombre), eliminacion (nombre).

Con dar clic en el nombre de la cuenta que esta en la parte izquierda del sistema con letra de color azul, subrayado se podra realizar la actualizacion.

Ingreso Clientes

El sistema permitira ingreso de nuevos clientes detallamos campos obligatorios.

Figura 41. Ingreso Clientes

Cedula o RUC: En este campo los usuarios pueden ingresar la cedula del cliente con un máximo de diez números.

Nombre: En este campo se escribirá los nombres y apellidos del nuevo cliente.

Teléfono: En este campo se escribe el número telefónico, solo admite números.

E_mail: Correo electrónico del cliente admite números y símbolos.

Dirección: Ubicación del domicilio del cliente admite números y símbolos.

Ingreso Cuenta Bancos

El sistema permitirá el ingresos de nuevas cuentas de bancos detallamos campos.

Figura 42. Ingreso Cuentas Bancos.

Número: Este campo permitirá el ingreso del número de cuenta de cliente.

Banco: En este campo se ingresara el nombre del banco.

Principal: En este campo se ingresara el nombre del dueño de la cuenta.

Ingreso Empleados

El sistema permitirá el ingreso de nuevos empleados.

Figura 43. Ingreso Empleados.

Cedula o RUC: En este campo los usuarios pueden ingresar la cedula del nuevo empleado.

Nombre: En este campo se escribirá los nombres y apellidos del nuevo empleado.

Teléfono: En este campo se escribe el número telefónico, solo admite números.

E_mail: Correo electrónico del cliente admite números y símbolos.

Dirección: Ubicación del domicilio del empleado admite números y símbolos.

Ingreso Productos

El sistema permitirá el ingreso de nuevos productos.

Figura 44. Ingreso Productos.

Nombre: En este campo se ingresará el nombre del producto.

Código: En este campo se ingresará el código del producto.

PVP: En este campo se ingresará el precio de venta ala público del producto.

Stock: En este campo se ingresara el número de productos en stock.

Descripción: En este campo se ingresará un pequeño detalle del producto.

Ingreso Cuentas

El sistema permitirá el ingreso de nuevas Cuentas.

Figura 45. Ingreso Cuentas.

Nombre: En este campo se ingresara el nombre de la cuenta.

Descripción: En este campo se realizara una pequeña descripci

Manual

Técnico

Nachos Sport

INDICE

INDICE DE FIGURAS

1. Introducción.

La empresa Nachos Sport ubicada en la ciudad de Riobamba ofrece el servicio de confección y venta de ropa deportiva, A la fecha actual la empresa cuenta con clientes y proveedores dentro y fuera del país, por eso se decidió implementar un sistema de información dentro de la empresa ya que es imprescindible contar con un sistema que facilite la gestión de la información que maneja la empresa. Además con el sistema se podrá tener un balance de la economía ya que se registraran los ingresos diarios de la misma.

2. Objetivo.

El objetivo del presente manual es mostrar los datos técnicos en cuanto al sistema desarrollado, en si para facilitar la modificación o actualizaciones de el mismo en caso de que así sea necesario, o bien para el mantenimiento posterior del mismo con el fin de que analistas, programadores puedan leerlo e interpretarlo para los objetivos anteriormente descritos.

Este manual se encuentra las secciones de estructura de Base de Datos, codificación del sistema donde se describirán los aspectos en los que se conforma cada uno de ellos. A continuación vera una breve descripción del sistema, los objetivos y las prestaciones que le ofrece el sistema.

2.1. Objetivo General: Este sistema está enfocado a mejorar el servicio que brinda la empresa Nachos Sport ya que la misma realiza gran cantidad de ventas a diario por lo que se vio la necesidad de implementar un sistema de facturación. El mismo que será capaz de registrar clientes, proveedores, productos y facilitar la emisión de facturas en diferentes formas de pago.

2.2. Objetivo del Sistema: Identificar al cliente tener acceso directo a sus datos y registrar su compra para poder emitir una factura, teniendo en cuenta que el cliente puede acceder a diferentes formas de pago, el sistema será capaz de emitir reportes sobre cuentas pendientes, ingresos diarios, mejores productos, mejores clientes.

3. Modelo Utilizado Para El Desarrollo Del Sistema

3.1. Modelo Espiral.

Elegimos este modelo porque nos permite regresar al punto donde se encuentra el error.

Este modelo fue propuesto por Boehm en 1988. Básicamente consiste en una serie de ciclos que se repiten en forma de espiral, comenzando desde el centro. Se suele interpretar como que dentro de cada ciclo de la espiral se sigue un Modelo Cascada, pero no necesariamente debe ser así. El Espiral puede verse como un modelo evolutivo que conjuga la naturaleza iterativa del modelo MCP con los aspectos controlados y sistemáticos del Modelo Cascada, con el agregado de gestión de riegos.

Tareas

Para cada ciclo habrá cuatro actividades:

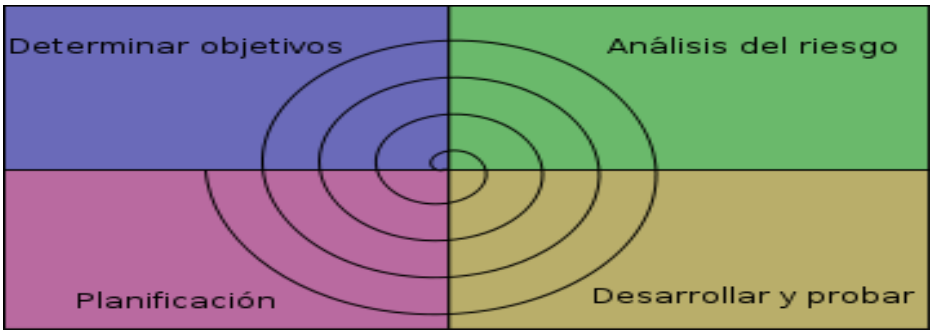


Figura 46. Modelo utilizado.

En la siguiente página se explicaran los 4 pasos que consta este modelo:

Determinar o fijar objetivos

- Fijar también los productos definidos a obtener: requerimientos, especificación, manual de usuario.
- Fijar las restricciones.
- Identificación de riesgos del proyecto y estrategias alternativas para evitarlos.
- Hay una cosa que solo se hace una vez: planificación inicial o previa.

Análisis del riesgo.

- Se estudian todos los riesgos potenciales y se seleccionan una o varias alternativas propuestas para reducir o eliminar los riesgos.

Desarrollar, verificar y validar (probar).

- Tareas de la actividad propia y de prueba.
- Análisis de alternativas e identificación resolución de riesgos.
- Dependiendo del resultado de la evaluación de los riesgos, se elige un modelo para el desarrollo, el que puede ser cualquiera de los otros existentes, como formal, evolutivo, cascada, etc. Así si por ejemplo si los riesgos en la interfaz de usuario son dominantes, un modelo de desarrollo apropiado podría ser la construcción de prototipos evolutivos. Si lo riesgos de protección son la principal consideración, un desarrollo basado en transformaciones formales podría ser el más apropiado.

Planificar

- Revisamos todo lo hecho, evaluándolo, y con ello decidimos si continuamos con las fases siguientes y planificamos la próxima actividad.

4. Diseño De Las Bases De Datos.

Para un funcionamiento eficiente se diseñaron las Bases de Datos que conforman el sistema con la aplicación MySQL debido a que esta aplicación nos permite un diseño amplio y concreto de las tablas y los campos que contiene la Base de Datos. La implementación de restricciones de seguridad y llaves foráneas entre otras cosas que incluye.

En primera instancia se muestran las tablas que conforman la Base de Datos.

TABLA USUARIO		
Nombre	**Tipo**	**Tamaño**
cedulausuario	Varchar	10
nombreusuario	Varchar	50
Teléfono	Varchar	10
Dirección	Varchar	50
Mail	Varchar	100
Clave	Varchar	35

TABLA CLIENTES		
Nombre	**Tipo**	**Tamaño**
cedulacliente	Varchar	13
nombrecliente	Varchar	100
Teléfono	Varchar	10
Dirección	Varchar	50
Mail	Varchar	100

TABLA CUENTA		
Nombre	**Tipo**	**Tamaño**
IdCuenta	Int	11
nombrecuenta	varchar	50
descripcionC	varchar	100

TABLA FACTURA		
Nombre	**Tipo**	**Tamaño**
IdFactura	Int	11
numero1	Int	11
numero2	Int	11

numero3	Int	11
Formapago	varchar	1
Total	Float	
Cliente(fk)	Varchar	13
Usuario(fk)	Varchar	10
Fecha	datetime	

TABLA SUBCUENTA		
Nombre	**Tipo**	**Tamaño**
IdSubCuenta	Int	11
Cantidad	Float	
Formapago	varchar	1
Estado	varchar	1
Facture(fk)	int	11
Cuenta(fk)	Int	11
Fechacobro	Date	
Fechaactual	date	

TABLA CUENTA BANCO		
Nombre	**Tipo**	**Tamaño**
numerocuenta	Varchar	20
Nombrebanco	Varchar	50
Principal	Varchar	50

TABLA CHEQUE		
Nombre	**Tipo**	**Tamaño**
IdCheque	Int	11
Factura(fk)	Int	11
Cuentabanco(fk)	varchar	20

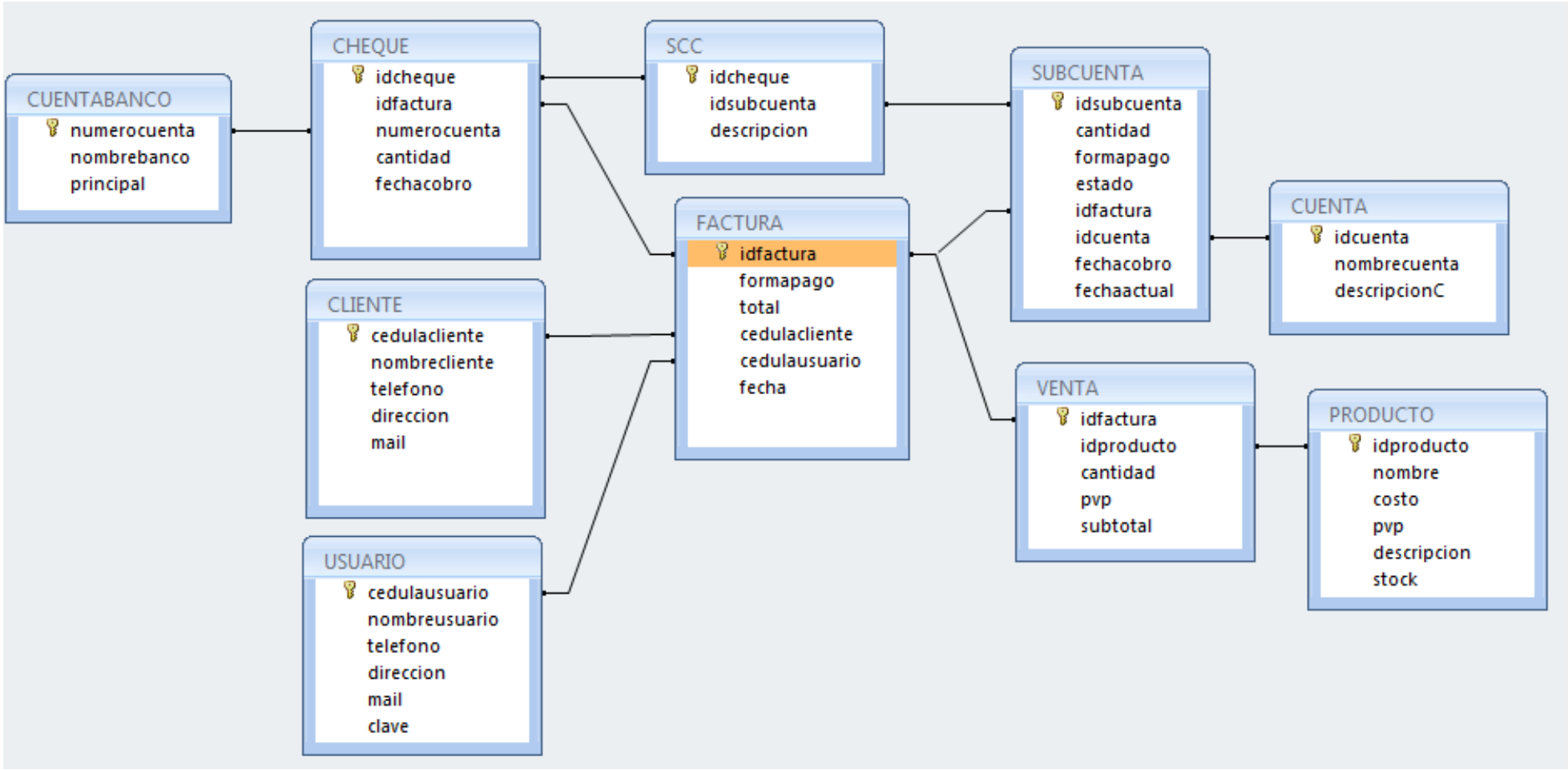
Cantidad	Float	
Fechacobro	Date	

TABLA PRODUCTO		
Nombre	**Tipo**	**Tamaño**
IdProducto	Int	11
Nombrep	varchar	50
Costo	Float	
Pvp	Float	
Descripcion	varchar	50
Stock	int	

TABLA VENTA		
Nombre	**Tipo**	**Tamaño**
Factura(fk)	Int	11
Producto(fk)	Int	11
Cantidad	Float	
Pvp	Float	
Subtotal	Float	

TABLA SCC		
Nombre	**Tipo**	**Tamaño**
Cheque(fk)	Int	11
Subcuenta(fk)	Int	11
Descripción	Varchar	50

5. Modelo Entidad Relación.



6. Codificacion De Las Pantallas.

El diseño de las pantallas se llevo a cabo con la aplicación PHP, al igual que la codificación de las mismas la cual se muestra en la siguiente sección:

Interface.

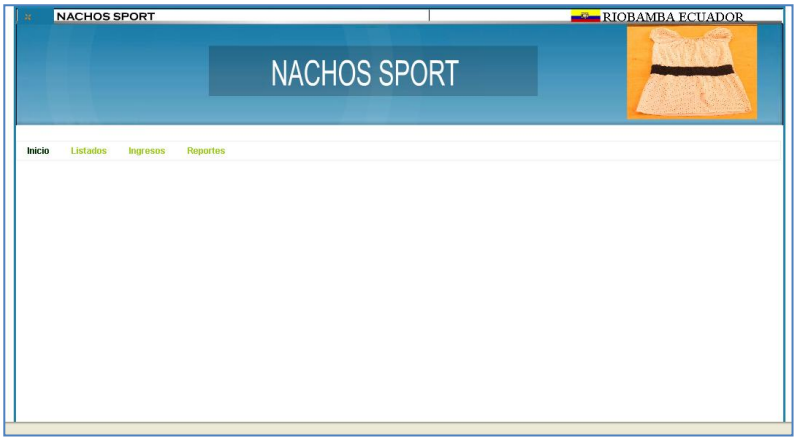


Figura 47. Pagina inicio.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<title>Nacho's Sport</title>
<script src="Scripts/swfobject_modified.js"
type="text/javascript"></script>
<link href="hojaStilo/menu.css" rel="stylesheet"
type="text/css" />
</head>
<body>
<table cellpadding="0" cellspacing="0" align="center"
style="border:solid 3px #2282AE">
<tr>
<td valign="top">
  <object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
width="1050" height="160" id="FlashID" title="banner">
    <param name="movie" value="imagenes/Banner.swf" />
    <param name="quality" value="high" />
    <param name="wmode" value="opaque" />
    <param name="swfversion" value="6.0.65.0" />
```

```
    <!-- Esta etiqueta param indica a los usuarios de Flash
Player 6.0 r65 o posterior que descarguen la versión más
reciente de Flash Player. Elimínela si no desea que los
usuarios vean el mensaje. -->
    <param name="expressinstall"
value="Scripts/expressInstall.swf" />
    <!-- La siguiente etiqueta object es para navegadores
distintos de IE. Ocúltela a IE mediante IECC. -->
    <!--[if !IE]>-->
    <object type="application/x-shockwave-flash"
data="imagenes/Banner.swf" width="1050" height="160">
      <!--<![endif]-->
      <param name="quality" value="high" />
      <param name="wmode" value="opaque" />
      <param name="swfversion" value="6.0.65.0" />
      <param name="expressinstall"
value="Scripts/expressInstall.swf" />
      <!-- El navegador muestra el siguiente contenido
alternativo para usuarios con Flash Player 6.0 o versiones
anteriores. -->
      <div>
        <h4>El contenido de esta página requiere una versión
más reciente de Adobe Flash Player.</h4>
        <p><a
href="http://www.adobe.com/go/getflashplayer"><img
src="http://www.adobe.com/images/shared/download_buttons/get_fl
ash_player.gif" alt="Obtener Adobe Flash Player" width="112"
height="33" /></a></p>
      </div>
      <!--[if !IE]>-->
    </object>
    <!--<![endif]-->
  </object></td>
</tr>
<tr>
<td valign="top">
<div  style=" border:solid 0px; top:inherit; vertical-
align:top">
<ul id="menu">
     <li><a class="selected" tihref="#">Inicio</a></li>
     <li><a href="#">Listados</a>
           <ul>
                 <li><a href="ListaClientes.php"
target="principal">Clientes</a></li>
                 <li><a href="ListaCuentasBancos.php"
target="principal">Cuentas Bancos</a></li>
            <li><a href="ListaEmpleados.php"
target="principal">Empleados</a></li>
            <li><a href="ListaProductos.php"
target="principal">Productos</a></li>
            <li><a href="IngresarCuenta.php"
target="principal">Cuentas</a></li>
          </ul>
```

```
</li>
<li><a href="#">Ingresos</a>
    <ul>
           <li><a href="IngresarCliente.php"
target="principal">Clientes</a></li>
           <li><a href="IngresarCuentaBanco.php"
target="principal">Cuentas de Banco</a></li>
        <li><a href="IngresarEmpleado.php"
target="principal">Empleados</a></li>
        <li><a href="IngresarProducto.php"
target="principal">Productos</a></li>
        <li><a href="IngresarCuenta.php"
target="principal">Cuentas</a></li>
     </ul>
</li>
<li><a href="#">Modificación</a>
<ul>
           <li><a href="ModificaProducto.php"
target="principal">Productos</a></li>
        <li><a href="ModificaCuentaBanco.php"
target="principal">Cuentas de Banco</a></li>
        <li><a href="ModificaEmpleado.php"
target="principal">Empleados</a></li>
        <li><a href="ModificaProducto.php"
target="principal">Productos</a></li>
        <li><a href="ModificaCuenta.php"
target="principal">Cuentas</a></li>
     </ul>
</li>
<li><a href="#">Venta</a></li>
<li><a href="#">Factura</a></li>
</ul>
</div>
</td>
</tr>
<tr>
<td align="center">
<iframe src="" frameborder="1" width="1000" height="500"
name="principal"></iframe>
</td>
</tr>
<tr >
<td height="30" style="background-color:#1677A2">
</td>
</tr>
</table>
<script type="text/javascript">
swfobject.registerObject("FlashID");
swfobject.registerObject("FlashID");
swfobject.registerObject("FlashID");
swfobject.registerObject("FlashID");
</script>
</body>
```

```
</html>
```

Crear Factura.

Figura 48. Crear Factura.

```
<?php
 include 'config/AccesoDatos/Componentes/Cliente.php';
                include 'config/AccesoDatos/clsCliente.php';
                include 'config/AccesoDatos/Componentes/Factura.php';
                include 'config/AccesoDatos/clsFactura.php';
                include 'config/AccesoDatos/DB_SQLserver.php';
                $cedula = $_REQUEST["id"];
                $factura = $_REQUEST["max"];
                $facturanumero = $_REQUEST["numero"];
         ?>
        <html>
            <head>
```

```
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
    </head>
    <body>
        <table align="center" border="1">
            <tr>
                <td><a
href="ListaProductosFacturar.php?max=<?php echo $factura
;?>&cedula=<?php echo $cedula ;?>&numero=<?php echo
$facturanumero ;?>"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                <td><a
href="config/FacturaProcesaE.php?factura=<?php echo $factura
;?>"><input type="image" src="imagenes/cancelar.png"
/></a></td>
            </tr>
        </table>
        <br>
        <table border="1" align="center" width="600">
                <tr>
                    <td height="1"
align="center"><h5><br>Direccion:Av. Daniel Leon Borja y La
Valle <br>
                        Telefono: 088265250 - Riobamba -
Ecuador</h5>
                    </td>

                    <td align="center" rowspan=="2">
                        <h5>RUC:1600319329001</h5>
                        <h3>FACTURA</h3>
                        <h5>NUM: <?php echo $facturanumero
;?></h5>
                        <h5>Cod. Autorizacion S.R.I.</h5>
                        <h5>1110257548</h5>
                    </td>
                </tr>
                <tr>
                    <?php
                        $objAcceso = new clsCliente();
                        $resultado = $objAcceso-
>BuscarCliente($cedula);
                        if($resultado == 0)
                        {
                            echo 0;
                        }
                        else
                        {
                            $total = count($resultado);
                            for($i = 0; $i < $total; $i++)
                            {?>
                                <td>Lugar y Fecha de Emicion:
Riobamba <?php echo date ( "j/n/Y" );?><br>
```

```
                                    Cliente: <?php echo
$resultado[$i][1] ;?><br>
                                    R.U.C./C.I.: <?php echo
$resultado[$i][0] ;?><br>
                                    Direccion: <?php echo
$resultado[$i][3] ;?><br>
                                    Telefono: <?php echo
$resultado[$i][2] ;?><br>
                                </td>
                        <?php
                            }
                        }
                        ?>
                </tr>
        </table>
        <table border="1" align="center" width="600">
            <thead>
                <tr>
                    <th width="80">CANTIDAD</th>
                    <th>DESCRIPCION</th>
                    <th width="80">V. UNIT.</th>
                    <th width="80">V. TOTAL</th>
                </tr>
            </thead>
            <tbody>
                    <?php
                        $objAcceso = new clsFactura();
                        $resultado = $objAcceso-
>BuscarProductosFactura($factura);
                        if($resultado == 0)
                        {
                        }
                        else
                        {
                            $total = count($resultado);
                            for($i = 0; $i < $total; $i++)
                            {?>
                                <tr>
                                    <td><?php echo
$resultado[$i][1] ;?></td>
                                    <td><?php echo
$resultado[$i][2] ;?></td>
                                    <td><?php echo
$resultado[$i][3] ;?></td>
                                    <td><?php echo
$resultado[$i][4] ;?></td>
                                </tr>
                     <?php
                            }
                        }
                        ?>
                    <table border="1" align="center"
width="600">
```

```
                        <tr><td align="right">Sub. Total
$</td><td width="80">1</td></tr>
                        <tr><td align="right">Gravado IVA
Tarifa 12% $</td><td width="80">1</td></tr>
                        <tr><td align="right">Importe del IVA
$</td><td width="80">1</td></tr>
                        <tr><td align="right">TOTAL $</td><td
width="80">1</td></tr>
                    </table>
                </tr>
            </tbody>
        </table>
    </body>
</html>
```

Ingresar Cliente.

Figura 49. Ingresar cliente.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Ingreso Empleado</title>
    <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <form action="config/ClienteProcesaI.php"
method="post">
            <table border="0" align="center" width="700">
                <tbody>
                    <tr align="center">
                        <td colspan="2" class="texto">Ingreso
de Cliente</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
```

```
                    </tr>
                    <tr>
                        <td width="90">C&eacute;dula/RUC</td>
                        <td>
                        <span id="cedula">
                        <label>
                          <input type="text" name="txtCedula"
id="ci" maxlength="13">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese la c&eacute;dula.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
&aacute;lido.</span></span>
                      </td>
                      <td rowspan="8" width="300"><img
src="imagenes/clientes.jpg" width="280" height="200"
alt="clientes"/></td>
                  </tr>
                    <tr>
                        <td>Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
                          <input type="text" name="txtNombre"
id="txtNombre">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Nombre.</span></span>
                      </td>
                    </tr>
                    <tr>
                        <td>Tel&eacute;fono</td>
                        <td>
                        <span id="fono">
                        <label>
                          <input type="text" name="txtTelefono"
id="txtTelefono" maxlength="10">
                        </label>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                      </td>
                    </tr>
                    <tr>
                        <td>E-mail</td>
                        <td>
                        <span id="mail">
                        <label>
                          <input type="text" name="txtEmail"
id="txtEmail">
                        </label>
```

```
<span class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                      </td>
                    </tr>
<tr>
                        <td>Direcci&oacute;n</td>
                        <td>
                        <textarea name="txtDireccion" cols="25"
rows="2"></textarea>
</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/ingresar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
    <script type="text/javascript">
<!--
var sprytextfield1 = new
Spry.Widget.ValidationTextField("cedula", "custom",
{validateOn:["blur"], pattern:"0000000000",
useCharacterMasking:true});
var sprytextfield2 = new
Spry.Widget.ValidationTextField("fono", "integer",
{isRequired:false, validateOn:["blur"]});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("mail", "email",
{isRequired:false, validateOn:["blur"]});
var sprytextfield4 = new Spry.Widget.ValidationTextField("nom",
"custom", {validateOn:["blur"]});
//-->
    </script>
    </body>
</html>
```

Ingresar Cuenta.

Figura 50. Ingresar cuenta

```
<html>
    <head>
```

```
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Ingreso Cuenta</title>
    <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <form action="config/CuentaProcesaI.php" method="post">
            <table width="700" border="0" align="center">
                <tbody>
                    <tr align="center">
                        <td colspan="2" class="texto">Ingreso
de Cuenta</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
                    </tr>
                    <tr>
                        <td width="80">Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
                          <input type="text" name="txtNombre"
id="txtNombre">
                        </label>
                        <span class="textfieldRequiredMsg">Se
necesita un valor.</span></span>
                      </td>
                      <td rowspan="6" width="300"><img
src="imagenes/cuentas2.jpg" width="200" height="125"
alt="cuentas2"/>
                      </td>
                    </tr>
                    <tr>
                        <td>Descripci&oacute;n</td>
                        <td>
                        <textarea name="txtDescripcion"
cols="20" rows="2"></textarea>
                        </td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/ingresar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
    <script type="text/javascript">
<!--
```

```
var sprytextfield1 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
//-->
        </script>
    </body>
</html>
```

Ingresar Cuentabanco.

Figura 51. Ingresar cuentabanco.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Ingreso Cuenta</title>
    <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <form action="config/CuentaBancoProcesaI.php"
method="post">
            <table width="750" border="0" align="center">
                <tbody>
                    <tr align="center">
                        <td colspan="2" class="texto">Ingreso
de Cuenta Banco</td>
                    </tr>
                    <tr>
```

```
                        <td width="60">N&uacute;mero</td>
                        <td>
                        <span id="num">
                        <label>
                          <input type="text" name="txtNumero"
id="txtNumero">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el N&uacute;mero de
Cuenta.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                        </td>
                        <td rowspan="8" width="300"><img
src="imagenes/money.jpg" width="180" height="130" alt="money"/>
                        </td>
                    </tr>
                    <tr>
                        <td>Banco</td>
                        <td>
                        <span id="banco">
                        <label>
                          <input type="text" name="txtBanco"
id="txtBanco">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el
Banco.</span></span></td>
                    </tr>
                    <tr>
                        <td width="70">Principal</td>
                        <td>
                        <span id="principal">
                        <label>
                          <input type="text"
name="txtPrincipal" id="txtPrincipal">
                        </label>
                        <span class="textfieldRequiredMsg">Se
necesita un valor.</span></span>
                        </td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/ingresar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("num",
"integer", {validateOn:["blur"]});
```

```
var sprytextfield2 = new
Spry.Widget.ValidationTextField("banco", "none",
{validateOn:["blur"]});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("principal", "none",
{validateOn:["blur"]});
//-->
    </script>
    </body>
</html>
```

Ingresar Empleado.

Figura 52. Ingresar empleado.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Ingreso Empleado</title>
    <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <form action="config/EmpleadoProcesaI.php"
method="post">
            <table border="0" align="center" width="700">
                <tbody>
                    <tr align="center">
```

```
                        <td colspan="2" class="texto">Ingreso
de Empleados</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
                    </tr>
                    <tr>
                        <td>C&eacute;dula</td>
                        <td>
                        <span id="ci">
                        <label>
                          <input type="text" name="txtCedula"
id="txtCedula">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese la c&eacute;dula.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                      </td>
                      <td rowspan="8" width="300"><img
src="imagenes/empleados.jpg" width="265" height="213"
alt="empleados%20grupo%20peq"/>
                      </td>
                    </tr>
                    <tr>
                        <td>Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
                          <input type="text" name="txtNombre"
id="txtNombre">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Nombre.</span></span>
                        </td>
                    </tr>
                    <tr>
                        <td>Tel&eacute;fono</td>
                        <td>
                        <span id="fono">
                        <label>
                          <input type="text" name="txtTelefono"
id="txtTelefono" maxlength="10">
                        </label>
<span class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                      </td>
                    </tr>
                    <tr>
                        <td>E-mail</td>
                        <td>
                        <span id="mail">
```

```
                        <label>
                          <input type="text" name="txtEmail"
id="txtEmail">
                        </label>
<span class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span></td>
                    </tr>
                    <tr>
                        <td>Clave</td>
                        <td>
                        <span id="clave">
                        <label>
                          <input type="password"
name="txtClave" id="txtClave">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese la
Clave.</span></span></td>
                    </tr>
                    <tr>
                        <td width="70">Direcci&oacute;n</td>
                        <td><textarea
name="txtDireccion"></textarea></td>
                    </tr>
                    <tr align="center">
                        <td colspan="2">
                          <input type="image"
src="imagenes/ingresar.png" align="middle" />
                        </td>
                    </tr>
                </tbody>
            </table>
        </form>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("ci",
"custom", {pattern:"0000000000", validateOn:["blur"],
useCharacterMasking:true});
var sprytextfield2 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("fono", "integer",
{validateOn:["blur"], isRequired:false});
var sprytextfield4 = new
Spry.Widget.ValidationTextField("mail", "email",
{isRequired:false});
var sprytextfield5 = new
Spry.Widget.ValidationTextField("clave", "none",
{validateOn:["blur"]});
//-->
    </script>
    </body>
</html>
```

Ingresar Producto.

Figura 53. Ingresar producto.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Ingreso Empleado</title>
    <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <form action="config/ProductoProcesaI.php"
method="post">
            <table width="700" border="0" align="center">
                <tbody>
                    <tr align="center">
                        <td colspan="2" class="texto">Ingreso
de Producto</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
                    </tr>
                    <tr>
                        <td width="80">Nombre</td>
                        <td>
                        <span id="nom">
                        <label>

                          <input type="text" name="txtNombre"
id="txtNombre">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Nombre.</span></span>
                        </td>
                        <td rowspan="7" width="300"><img
src="imagenes/producto.JPG" width="200" height="180"
alt="producto"/>
                        </td>
```

```
                    </tr>
                    <tr>
                        <td>Costo</td>
                        <td>
                        <span id="costo">
                        <label>
                          <input type="text" name="txtCosto"
id="txtCosto">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Costo.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span></td>
                    </tr>
                    <tr>
                        <td>PVP</td>
                        <td>
                        <span id="pvp">
                        <label>
                          <input type="text" name="txtPVP"
id="txtPVP">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el PVP.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span></td>
                    </tr>
                    <tr>
                        <td>Stock</td>
                        <td>
                        <span id="stock">
                        <label>
                          <input type="text" name="txtStock"
id="txtStock">
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el
Stock.</span></span></td>
                    </tr>
                    <tr>
                        <td>Descripci&oacute;n</td>
                        <td><textarea
name="txtDescripcion"></textarea></td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/ingresar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
```

```
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
var sprytextfield2 = new
Spry.Widget.ValidationTextField("costo", "real",
{validateOn:["blur"]});
var sprytextfield3 = new Spry.Widget.ValidationTextField("pvp",
"real", {validateOn:["blur"]});
var sprytextfield4 = new
Spry.Widget.ValidationTextField("stock", "none",
{validateOn:["blur"]});
//-->
    </script>
    </body>
</html>
```

Listar Clientes.

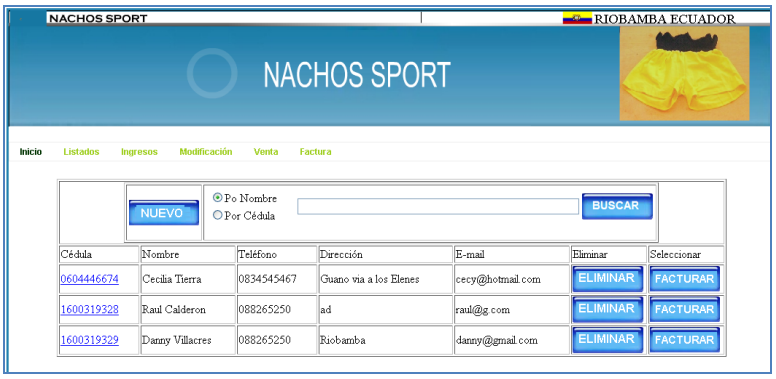


Figura 54. Listar clientes.

```
<?
    include 'config/AccesoDatos/Componentes/Cliente.php';
    include 'config/AccesoDatos/clsCliente.php';
    include 'config/AccesoDatos/DB_SQLserver.php';
?>
<html>
    <head>
```

```
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Lista Clientes</title>
    </head>
    <body>
        <table width="918" border="1" align="center">
          <tr>
            <td colspan="7">
                <table border="1" align="center">
                  <tr>
                    <td width="100" align="center"><a
href="IngresarCliente.php"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                    <td>
                        <form id="form1" name="form1"
method="post" action="ListaClientes.php" >
                            <table border="0">
                                <tr>
                                    <td>
                                        <table border="0"
width="120">
                                            <tr><td><input
type="radio" name="op" value="1" checked="checked" />Po
Nombre</td></tr>
                                            <tr><td><input
type="radio" name="op" value="2" />Por C&eacute;dula</td></tr>
                                        </table>
                                    </td>
                                    <td>
                                      <input name="pnombre"
type="text" id="pnombre" size="60" />
                                    </td>
                                    <td>
                                      <input type="image"
src="imagenes/buscar.png" name="Buscar" id="Buscar"/>
                                    </td>
                                </tr>
                            </table>
                        </form>
                    </td>
                  </tr>
                </table>
            </td>
          </tr>
    <tr>
        <td>C&eacute;dula</td>
        <td>Nombre</td>
        <td>Tel&eacute;fono</td>
        <td>Direcci&oacute;n</td>
        <td>E-mail</td>
        <td width="100">Eliminar</td>
        <td width="100">Seleccionar</td>
    </tr>
```

```
    <?php
            $objAcceso = new clsCliente();
            if (isset($_POST['pnombre']))
            {
                if ($_POST['op']==1)
                {
                    $resultado = $objAcceso-
>BuscarClienteNombre($_POST["pnombre"]);
                }
                else
                {
                    $resultado = $objAcceso-
>BuscarCliente($_POST["pnombre"]);
                }
            }
            else
            {
                $resultado = $objAcceso->DatosCliente();
            }

        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {
               echo '<tr>';
                   echo '<td><a href="ModificaCliente.php?id='
. $resultado[$i][0] . '">'. $resultado[$i][0] .'</a></td>';
                   echo '<td>'. $resultado[$i][1] .'</td>';
                   echo '<td>'. $resultado[$i][2] .'</td>';
                   echo '<td>'. $resultado[$i][3] .'</td>';
                   echo '<td>'. $resultado[$i][4] .'</td>';
                   echo '<td><a
href="config/ClienteProcesaE.php?id=' . $resultado[$i][0] .
'"><input type="image" src="imagenes/eliminar.png"
/></a></td>';
                   echo '<td><a
href="config/FacturaProcesaI.php?id=' . $resultado[$i][0] .
'"><input type="image" src="imagenes/facturar.png"
/></a></td>';
               echo '</tr>';
            }
        }
    ?>

</table>
</body>
</html>
```

Listar Cuentas.

Figura 55. Listar cuentas.

```
<?
    include 'config/AccesoDatos/Componentes/Cuenta.php';
    include 'config/AccesoDatos/clsCuenta.php';
    include 'config/AccesoDatos/DB_SQLserver.php';
?>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Lista de Cuentas</title>
    </head>
    <body>
        <table width="918" border="1" align="center">
          <tr>
            <td colspan="6">
                <table border="1" align="center">
                  <tr>
                    <td width="100" align="center"><a
href="IngresarCuenta.php"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                    <td>
                        <form id="form1" name="form1"
method="post" action="ListaCuentas.php">
                            <table border="0">
                                <tr>
                                    <td>
                                        <table border="0"
width="120">
                                            <tr><td><input
type="radio" name="op" value="1" checked="checked" />Por
Nombre</td></tr>
                                        </table>
                                    </td>
                                    <td>
                                        <input
name="pnombre" type="text" id="pnombre" size="60" />
                                    </td>
```

```
                                        <td>
                                           <input type="image"
src="imagenes/buscar.png" name="Buscar" id="Buscar"/>
                                        </td>
                                    </tr>
                                </table>
                            </form>
                        </td>
                    </tr>
                </table>
            </td>
           </tr>
    <tr>
        <td>Nombre</td>
        <td>Descripci&oacute;n</td>
        <td width="100">Eliminar</td>
    </tr>
    <?php
            $objAcceso = new clsCuenta();
            if (isset($_POST['pnombre']))
            {
                if ($_POST['op']==1)
                {
                    $resultado = $objAcceso-
>BuscarCuentaNombre($_POST["pnombre"]);
                }
                else
                {
                    $resultado = $objAcceso-
>BuscarCuenta($_POST["pnombre"]);
                }
            }
            else
            {
                $resultado = $objAcceso->DatosCuentas();
            }

        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {
               echo '<tr>';
                   echo '<td><a href="ModificaCuenta.php?id=' .
$resultado[$i][0] . '">'. $resultado[$i][1] .'</a></td>';
                   echo '<td>'. $resultado[$i][2] .'</td>';
                   echo '<td><a
href="config/CuentaProcesaE.php?id=' . $resultado[$i][0] .
```

```
'"><input type="image" src="imagenes/eliminar.png"
/></a></td>';
                echo '</tr>';
            }
        }
    ?>

</table>
</body>
</html>
```

Listar Cuentas Bancos.

Figura 56. Listar cunetas bancos.

```
<?
    include 'config/AccesoDatos/Componentes/CuentaBanco.php';
    include 'config/AccesoDatos/clsCuentaBanco.php';
    include 'config/AccesoDatos/DB_SQLserver.php';
?>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Lista Cuentas Banco</title>
    </head>
    <body>
        <table width="918" border="1" align="center">
          <tr>
            <td colspan="6">
                <table border="1" align="center">
                  <tr>
                    <td width="100" align="center"><a
href="IngresarCuentaBanco.php"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                    <td>
                        <form id="form1" name="form1"
method="post" action="ListaCuentasBancos.php">
                            <table border="0">
```

```
                                <tr>
                                    <td>
                                        <table border="0"
width="120" >
                                            <tr><td><input
type="radio" name="op" value="1" checked="checked" />Por
Principal</td></tr>
                                            <tr><td><input
type="radio" name="op" value="2" />Por Cuenta</td></tr>
                                        </table>
                                    </td>
                                    <td>
                                        <input
name="pnombre" type="text" id="pnombre" size="60" />
                                    </td>
                                    <td>
                                        <input type="image"
src="imagenes/buscar.png" name="Buscar" id="Buscar"/>
                                    </td>
                                </tr>
                            </table>
                        </form>
                    </td>
                </tr>
            </table>
        </td>
    </tr>
    <tr>
        <td>N&uacute;mero</td>
        <td>Banco</td>
        <td>Principal</td>
        <td width="100">Eliminar</td>
    </tr>
    <?php
            $objAcceso = new clsCuentaBanco();
            if (isset($_POST['pnombre']))
            {
                if ($_POST['op']==1)
                {
                    $resultado = $objAcceso-
>BuscarCuentaBancoNombre($_POST["pnombre"]);
                }
                else
                {
                    $resultado = $objAcceso-
>BuscarCuentaBanco($_POST["pnombre"]);
                }
            }
            else
            {
                $resultado = $objAcceso->DatosCuentasBancos();
            }
```

```
        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {
               echo '<tr>';
                   echo '<td><a
href="ModificaCuentaBanco.php?id=' . $resultado[$i][0] . '">'.
$resultado[$i][0] .'</a></td>';
                   echo '<td>'. $resultado[$i][1] .'</td>';
                   echo '<td>'. $resultado[$i][2] .'</td>';
                   echo '<td><a
href="config/CuentaBancoProcesaE.php?id=' . $resultado[$i][0] .
'"><input type="image" src="imagenes/eliminar.png"
/></a></td>';
               echo '</tr>';
            }
        }
    ?>

</table>
</body>
</html>
```

Listar Empleados.

Figura 57. Listar empleados.

```
<?
    include 'config/AccesoDatos/Componentes/Empleado.php';
    include 'config/AccesoDatos/clsEmpleado.php';
    include 'config/AccesoDatos/DB_SQLserver.php';
?>

<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Lista Usuarios</title>
    </head>
    <body>
        <table width="918" border="1" align="center">
          <tr>
            <td colspan="6">
                <table border="1" align="center">
                  <tr>
                    <td width="100" align="center"><a
href="IngresarEmpleado.php"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                    <td>
                        <form id="form1" name="form1"
method="post" action="ListaEmpleados.php">
                            <table border="0">
                                <tr>
                                    <td>
                                        <table border="0"
width="120">
                                            <tr><td><input
type="radio" name="op" value="1" checked="checked" />Po
Nombre</td></tr>
                                            <tr><td><input
type="radio" name="op" value="2" />Por C&eacute;dula</td></tr>
                                        </table>
                                    </td>
                                    <td>
                                        <input
name="pnombre" type="text" id="pnombre" size="60" />
                                    </td>
                                    <td>
                                        <input type="image"
src="imagenes/buscar.png" name="Buscar" id="Buscar"/>
                                    </td>
                                </tr>
                            </table>
                        </form>
                    </td>
                </tr>
```

```
                </table>
            </td>
          </tr>
    <tr>
        <td>C&eacute;dula</td>
        <td>Nombre</td>
        <td>Tel&eacute;fono</td>
        <td>Direcci&oacute;n</td>
        <td>E-mail</td>
        <td width="100">Eliminar</td>
    </tr>
    <?php
        $objAcceso = new clsEmpleado();

            if (isset($_POST['pnombre']))
            {
                if ($_POST['op']==1)
                {
                    $resultado = $objAcceso-
>BuscarEmpleadoNombre($_POST["pnombre"]);
                }
                else
                {
                    $resultado = $objAcceso-
>BuscarEmpleado($_POST["pnombre"]);
                }
            }
            else
            {
                $resultado = $objAcceso->DatosEmpleado();
            }

        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {
               echo '<tr>';
                   echo '<td><a href="ModificaEmpleado.php?id='
. $resultado[$i][0] . '">'. $resultado[$i][0] .'</a></td>';
                   echo '<td>'. $resultado[$i][1] .'</td>';
                   echo '<td>'. $resultado[$i][2] .'</td>';
                   echo '<td>'. $resultado[$i][3] .'</td>';
                   echo '<td>'. $resultado[$i][4] .'</td>';
                   echo '<td><a
href="config/EmpleadoProcesaE.php?id=' . $resultado[$i][0] .
'"><input type="image" src="imagenes/eliminar.png"
/></a></td>';
               echo '</tr>';
```

```
            }
        }
    ?>
</table>
</body>
</html>
```

Lista Productos.

Figura 58. Listar Productos.

```
<?
    include 'config/AccesoDatos/Componentes/Producto.php';
    include 'config/AccesoDatos/clsProducto.php';
    include 'config/AccesoDatos/DB_SQLserver.php';
?>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title>Lista Productos</title>
    </head>
    <body>
        <table width="918" border="1" align="center">
          <tr>
            <td colspan="6">
                <table border="1" align="center">
                  <tr>
                    <td width="100" align="center"><a
href="IngresarProducto.php"><input type="image"
src="imagenes/nuevo.png" /></a></td>
                    <td>
                        <form id="form1" name="form1"
method="post" action="ListaProductos.php">
                            <table border="0">
                                <tr>
                                    <td>
```

```
                                        <table border="0"
width="120">
                                            <tr>
                                            <td><input
type="radio" name="op" value="1" checked="checked" />Por
Nombre</td>
                                            </tr>
                                            <tr><td><input
type="radio" name="op" value="2" />Por C&oacute;digo</td></tr>
                                        </table>
                                    </td>
                                    <td>
                                            <input
name="pnombre" type="text" id="pnombre" size="60" />
                                    </td>
                                    <td>
                                        <input type="image"
src="imagenes/buscar.png" name="Buscar" id="Buscar"/>
                                    </td>
                                </tr>
                            </table>
                        </form>
                    </td>
                </tr>
            </table>
        </td>
       </tr>
    <tr>
        <td>Nombre</td>
        <td>Costo</td>
        <td>PVP</td>
        <td>Descripci&oacute;n</td>
        <td>Stock</td>
        <td width="100">Eliminar</td>
    </tr>
    <?php
            $objAcceso = new clsProducto();
            if (isset($_POST['pnombre']))
            {
                if ($_POST['op']==1)
                {
                    $resultado = $objAcceso-
>BuscarProductoNombre($_POST["pnombre"]);
                }
                else
                {
                    $resultado = $objAcceso-
>BuscarCliente($_POST["pnombre"]);
                }
            }
            else
            {
                $resultado = $objAcceso->DatosProducto();
```

```
            }

        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {
               echo '<tr>';
                   echo '<td><a href="ModificaProducto.php?id='
. $resultado[$i][0] . '">'. $resultado[$i][1] .'</a></td>';
                   echo '<td>'. $resultado[$i][2] .'</td>';
                   echo '<td>'. $resultado[$i][3] .'</td>';
                   echo '<td>'. $resultado[$i][4] .'</td>';
                   echo '<td>'. $resultado[$i][5] .'</td>';
                   echo '<td><a
href="config/ProductoProcesaE.php?id=' . $resultado[$i][0] .
'"><input type="image" src="imagenes/eliminar.png"
/></a></td>';
               echo '</tr>';
            }
        }
    ?>

</table>
</body>
</html>
```

Modificar Cliente.

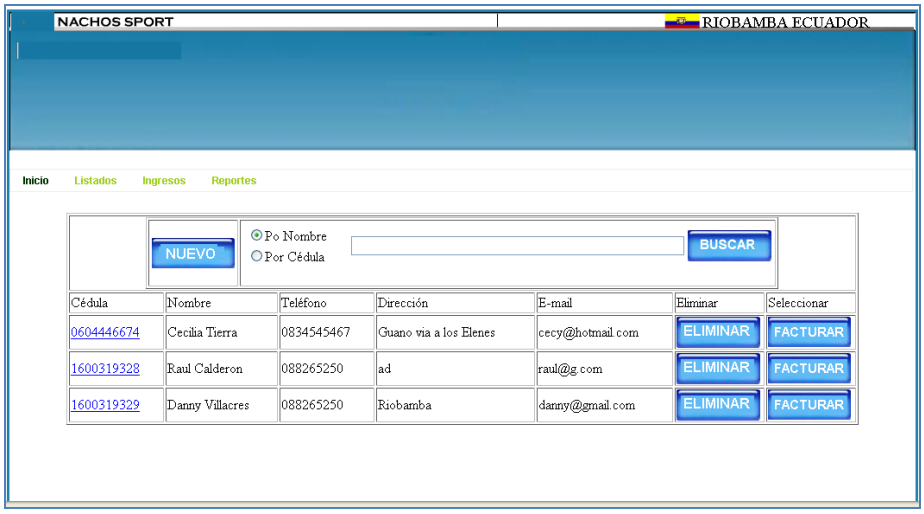


Figura 59. Modificar cliente.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
            <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <?php

        include 'config/AccesoDatos/Componentes/Cliente.php';
        include 'config/AccesoDatos/clsCliente.php';
        include 'config/AccesoDatos/DB_SQLserver.php';

        $cedula = $_REQUEST["id"];
        $objAcceso = new clsCliente();
        $resultado = $objAcceso->BuscarCliente($cedula);
        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {?>
    <form action="config/ClienteProcesaM.php" method="post">
                    <table width="355" border="0"
align="center">
```

```
                    <tbody>
                    <tr align="center">
                      <td colspan="2" class="texto">Actualizar
Cliente</td>
                    </tr>
                    <tr align="center">
                      <td colspan="2"><br></td>
                    </tr>
                    <tr>
                        <td>C&eacute;dula</td>
                        <td>
<?php echo'              <input type="text" name="txtCedula"
value=' . $resultado[$i][0] . ' readonly />';?>
                        </td>
                    </tr>
                    <tr>
                        <td width="75">Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
<?php echo'             <input type="text" name="txtNombre"
id="txtNombre" value=' . $resultado[$i][1] . '>';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Nombre.</span></span>
                      </td>
                    </tr>
                    <tr>
                        <td>Tel&eacute;fono</td>
                        <td>
                        <span id="fono">
                        <label>
<?php echo'              <input type="text" name="txtTelefono"
id="txtTelefono" value=' . $resultado[$i][2] . ' maxlength="10"
>';?>
                        </label>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido</span></span>
                        </td>
                    </tr>
                    <tr>
                        <td>E-mail</td>
                        <td>
                        <span id="mail">
                        <label>
<?php echo'              <input type="text" name="txtEmail"
id="txtEmail" value=' . $resultado[$i][4] . ' >';?>
                        </label>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                        </td>
```

```
                    </tr>
                    <tr>
                        <td>Direcci&oacute;n</td>
                        <td>
<?php echo'               <textarea name="txtDireccion">' .
$resultado[$i][3] . '</textarea>';?>
                        </td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/actualizar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
 <?php    }
        }
        ?>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
var sprytextfield2 = new
Spry.Widget.ValidationTextField("fono", "integer",
{validateOn:["blur"], isRequired:false});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("mail", "email",
{validateOn:["blur"], isRequired:false});
//-->
    </script>
    </body>
</html>
```

Modificar Cuenta

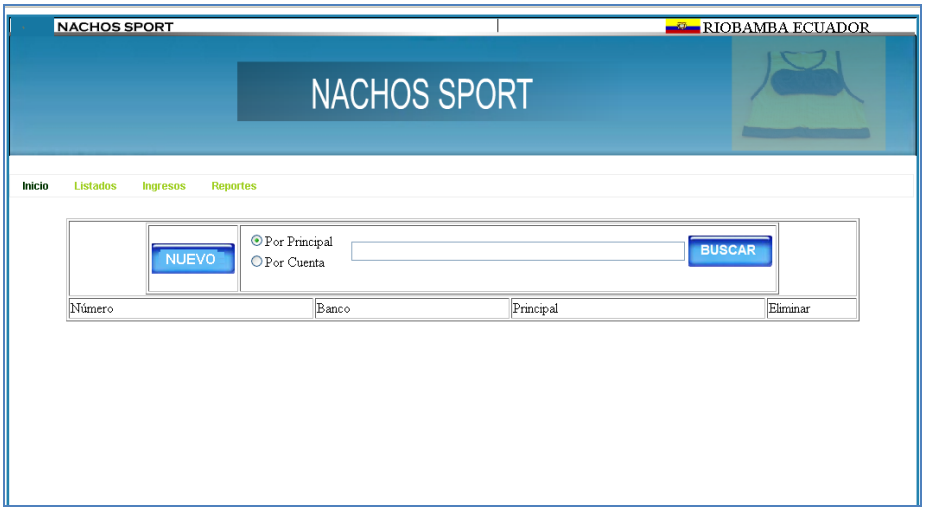


Figura 60. Modificar cuenta.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
            <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
    </head>
    <body>
        <?php

        include 'config/AccesoDatos/Componentes/Cuenta.php';
        include 'config/AccesoDatos/clsCuenta.php';
        include 'config/AccesoDatos/DB_SQLserver.php';

        $codigo = $_REQUEST["id"];
        $objAcceso = new clsCuenta();
        $resultado = $objAcceso->BuscarCuenta($codigo);
        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);

            for($i = 0; $i < $total; $i++)
```

```
            {
               ?>
                     <form action="config/CuentaProcesaM.php"
method="post">
                     <table width="350" border="0"
align="center">
                     <tbody>
                     <tr align="center" class="texto">
                     <td colspan="2">Actualizar Cuenta</td>
                     </tr>
                     <tr align="center">
                         <td colspan="2"><br></td>
                     </tr>
                     <tr>
                         <td><input type="hidden"
name="txtCodigo" value=' . $resultado[$i][0] . ' /></td>
                     </tr>
                     <tr>
                         <td width="80">Nombre</td>
                         <td>
                                   <span id="nom">
        <label>
          <?php echo '<input type="text" name="txtNombre"
id="txtNombre" value=' . $resultado[$i][1] . ' >' ;?>
        </label>
        <span class="textfieldRequiredMsg">Ingrese el
Nombre.</span></span>
                        </td>
                     </tr>
                     <tr>
                       <td>Descripci&oacute;n</td>
                       <td>
                       <?php echo'             <textarea
name="txtDescripcion" value=' . $resultado[$i][2] . '
></textarea>';?>
                                   </td>
                     </tr>
                     <tr align="center">
                         <td colspan="2"><input type="image"
src="imagenes/actualizar.png" align="middle" /></td>
                     </tr>
                  </tbody>
             </table>
         </form>
           <?php   }
         }
         ?>
    <script type="text/javascript">
    var sprytextfield1 = new
Spry.Widget.ValidationTextField("nom", "none",
{validateOn:["blur"]});
    </script>
    </body>
```

```
</html>
```

Modifica Cuentabanco

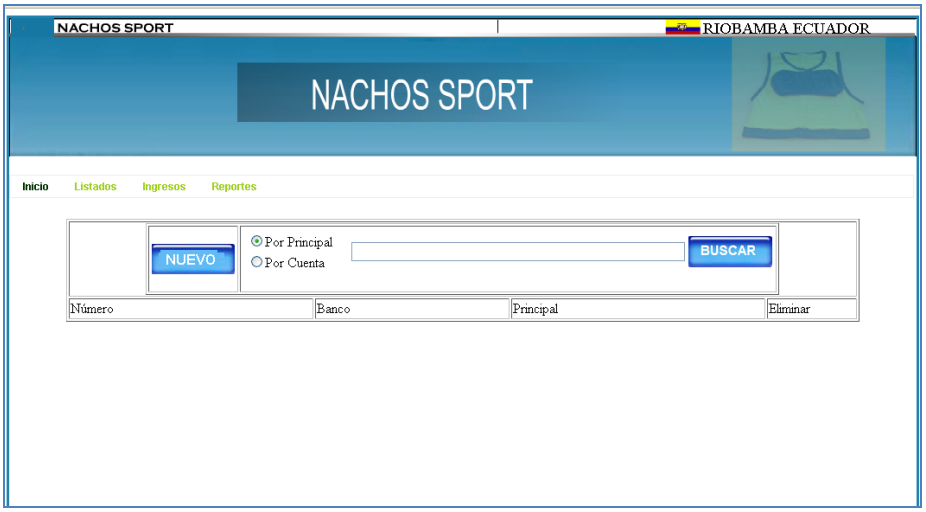


Figura 61. Modificar cuentabanco.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
            <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
            echo 0;
        }
        else
        {
            $total = count($resultado);

            for($i = 0; $i < $total; $i++)
            {?>
                    <form
action="config/CuentaBancoProcesaM.php" method="post">
                    <table width="352" border="0"
align="center">
                    <tbody>
                    <tr align="center">
```

```
                    <td colspan="2" class="texto">Actualizar
Cuenta Banco</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
                    </tr>
                    <tr>
                        <td width="65">N&uacute;mero</td>
                        <td>
                        <span id="num">
                        <label>
<?php echo'             <input type="text" name="txtNumero"
id="txtNumero" value=' . $resultado[$i][0] . ' >';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el # de Cuenta.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                                </td>
                    </tr>
                    <tr>
                        <td>Banco</td>
                        <td>
                        <span id="banco">
                        <label>
<?php echo'              <input type="text" name="txtBanco"
id="txtBanco" value=' . $resultado[$i][1] . '>';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Banco.</span></span>
                                </td>
                    </tr>
                    <tr>
                        <td>Principal</td>
                        <td>
                        <span id="Principal">
                        <label>
<?php echo'              <input type="text" name="txtPrincipal"
id="txtPrincipal" value=' . $resultado[$i][2] . '>';?>
                        </label>
                        <span class="textfieldRequiredMsg">Se
necesita un valor.</span></span>
                                </td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/actualizar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
  <?php   }
```

```
        }
        ?>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("num",
"integer", {validateOn:["blur"]});
var sprytextfield2 = new
Spry.Widget.ValidationTextField("banco", "none",
{validateOn:["blur"]});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("Principal", "none",
{validateOn:["blur"]});
//-->
    </script>
    </body>
</html>
```

Modificar Empleado.

Figura 62.

Modificar cliente.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
            <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <?php
```

```
        include 'config/AccesoDatos/Componentes/Empleado.php';
        include 'config/AccesoDatos/clsEmpleado.php';
        include 'config/AccesoDatos/DB_SQLserver.php';

        $cedula = $_REQUEST["id"];
        $objAcceso = new clsEmpleado();
        $resultado = $objAcceso->BuscarEmpleado($cedula);
        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
Ingrese la Cedula.</span><span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                    </tr>
                    <tr>
                        <td>Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
   <?php echo'            <input type="text" name="txtNombre"
id="txtNombre" value=' . $resultado[$i][1] . ' />';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Nombre.</span></span>
                        </td>
                    </tr>
                    <tr>
                        <td>Tel&eacute;fono</td>
                        <td>
<span id="fono">
                        <label>
<?php echo'             <input type="text" name="txtTelefono"
id="txtTelefono" value=' . $resultado[$i][2] . '
maxlength="10">';?>
                        </label>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                    </tr>
                    <tr>
                        <td>E-mail</td>
                        <td>
                        <span id="mail">
                        <label>
<?php echo'             <input type="text" name="txtEmail"
id="txtEmail" value=' . $resultado[$i][4] . '>';?>
                        </label>
```

```
                                    <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                        </td>
                    </tr>
                    <tr>
                        <td>Clave</td>
                        <td>
                          <span id="clave">
                          <label>
<?php echo'               <input type="password"
name="txtClave" id="txtClave" value=' . $resultado[$i][5] .
'>';?>
                          </label>
                        <span
class="textfieldRequiredMsg">Ingrese la
Clave.</span></span></td>
                    </tr>
                    <tr>
                        <td>Direcci&oacute;n</td>
                        <td>
<?php echo'             <textarea name="txtDireccion" cols="20"
rows="2">' . $resultado[$i][3] . '</textarea>';?>
                                    </td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><input type="image"
src="imagenes/actualizar.png" align="middle" /></td>
                    </tr>
                </tbody>
            </table>
        </form>
<?php    }
        }
        ?>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("ci",
"custom", {pattern:"0000000000", useCharacterMasking:true,
validateOn:["blur"]});
var sprytextfield2 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
var sprytextfield3 = new
Spry.Widget.ValidationTextField("fono", "integer",
{isRequired:false, validateOn:["blur"]});
var sprytextfield4 = new
Spry.Widget.ValidationTextField("clave", "none",
{validateOn:["blur"]});
var sprytextfield5 = new
Spry.Widget.ValidationTextField("mail", "email",
{isRequired:false});
//-->
    </script>
```

```
    </body>
</html>
```

Modificar Producto.

Figura 63. Modificar producto.

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">
        <title></title>
            <link href="hojaStilo/texto.css" rel="stylesheet"
type="text/css">
    <script src="SpryAssets/SpryValidationTextField.js"
type="text/javascript"></script>
    <link href="SpryAssets/SpryValidationTextField.css"
rel="stylesheet" type="text/css">
</head>
    <body>
        <?php

        include 'config/AccesoDatos/Componentes/Producto.php';
        include 'config/AccesoDatos/clsProducto.php';
        include 'config/AccesoDatos/DB_SQLserver.php';

        $codigo = $_REQUEST["id"];
        $objAcceso = new clsProducto();
        $resultado = $objAcceso->BuscarProducto($codigo);
        if($resultado == 0)
        {
            echo 0;
        }
        else
        {
            $total = count($resultado);
            for($i = 0; $i < $total; $i++)
            {?>
    <form action="config/ProductoProcesaM.php" method="post">
```

```
                    <table width="379" height="232" border="0"
align="center">
                    <tbody>
                    <tr align="center">
                    <td colspan="2" class="texto">Actualizar
Producto</td>
                    </tr>
                    <tr align="center">
                        <td colspan="2"><br></td>
                    </tr>
                    <tr>
<?php echo'          <td><input type="hidden" name="txtCodigo"
value=' . $resultado[$i][0] . ' /></td>';?>
                    </tr>
                    <tr>
                        <td>Nombre</td>
                        <td>
                        <span id="nom">
                        <label>
<?php echo'               <input type="text" name="txtNombre"
id="txtNombre" value=' . $resultado[$i][1] . '>';?>
                        </label>
                        <span class="textfieldRequiredMsg">Se
necesita un valor.</span></span>
                                </td>
                    </tr>
                    <tr>
                        <td>Costo</td>
                        <td>
                        <span id="costo">
                        <label>
<?php echo'              <input type="text" name="txtCosto"
id="txtCosto" value=' . $resultado[$i][2] . '>';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el Costo.</span>
                        <span
class="textfieldInvalidFormatMsg">Formato no
v&aacute;lido.</span></span>
                      </td>
                    </tr>
                    <tr>
                        <td>P.V.P.</td>
                        <td>
                        <span id="pvp">
                        <label>
<?php echo'              <input type="text" name="txtPVP"
id="txtPVP" value=' . $resultado[$i][3] . ' >';?>
                        </label>
                        <span
class="textfieldRequiredMsg">Ingrese el PVP.</span>

            </table>
```

```
        </form>
 <?php           }
        }
        ?>
    <script type="text/javascript">
<!--
var sprytextfield1 = new Spry.Widget.ValidationTextField("nom",
"none", {validateOn:["blur"]});
var sprytextfield2 = new
Spry.Widget.ValidationTextField("costo", "real",
{validateOn:["blur"]});
var sprytextfield3 = new Spry.Widget.ValidationTextField("pvp",
"real", {validateOn:["blur"]});
var sprytextfield4 = new Spry.Widget.ValidationTextField("stc",
"none", {validateOn:["blur"]});
//-->
    </script>
</body>
</html>

</html>
```

7. Al Usuario Final.

En este manual técnico se describen los componentes básicos, todo esto con el objetivo de que se pueda leer, interpretar y analizar las partes de que se conforma el sistema desarrollado a fin de que se le desee realizar modificaciones futuras o bien actualizaciones para mejorar su eficiencia y de ser posible sea base para algunos sistemas futuros a desarrollarse que sean afines a este.

En este manual podrá encontrar información referente a:

- Diseño de la Base de Datos.
- Tipos de Datos utilizados.
- Vista de las Pantallas
- Codificación de los botones de las pantallas.
- Modelo implementado a seguir para la realización del sistema

Con este documento se espera que sea claro, fácil de entender para los programadores, diseñadores, analistas de software que en si deseen, realizar alguna de las acciones descritas anteriormente al sistema.