UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DESARROLLO DE UN SISTEMA DE COTIZACIÓN Y EMISIÓN DE PÓLIZAS PARA UNA AGENCIA DE AUTOS.

TESIS

PARA OBTENER EL TITULO DE INGENIERO EN COMPUTACIÓN

PRESENTAN:

Gamiño Peón Ramón Moisés
Murillo González Rodrigo Salvador
Rojas Cortés Víctor
Rosas Ramírez Pedro
Tovar Pacheco Julio César

Directora: Ing. Lucila Patricia Arellano Mendoza.

Ciudad Universitaria. **México D.F. Octubre 2007**

AGRADECIMIENTOS

Dedico esta tesis a mis padres Rosa María Peón y Ramón Gamiño a quienes agradezco de todo corazón por su cariño, comprensión y apoyo sin condiciones ni medida, que siempre me impulsaron a seguir adelante y levantarme de cada tropiezo en mi camino, gracias por guiarme sobre el camino de la educación, gracias por ser unos padres tan excelentes.

A mis hermanos por sus consejos y su gran cariño que me mostraron durante este difícil camino y largo camino.

A mis cuatro amigos Pedro, Cesar, Victor y Rodrigo por su esmero y gran amistad que logro que esta tesis se realizará y se llegará a su culminación puesto que fue un gran esfuerzo de todos.

A Dios por permitirme llegar a esta meta tan importante en mi vida.

Gamiño Peón Ramón Moisés

A mi esposa e hija, los amores de mi vida

A mi padre, madre y hermano, fuentes de admiración, fortaleza, orgullo y respeto.

A mi nueva familia por todo su apoyo y confianza.

Y a mis verdaderos amigos, por permitirme hacerlos parte de mi vida.

Murillo González Rodrigo Salvador

Después de un esfuerzo enorme, llegó el momento de un logro más en mi vida, el cual quiero agradecer a:

Mi esposa:
Por todo su apoyo que me ha brindado para hacer mi trabajo de tesis.

Mis padres:
Por la confianza, el apoyo y motivación que me dieron, ya que sin ellos esto no hubiera sido posible.

A Dios:
Por la fuerza de voluntad que me proporciono para que esto fuera posible.

La Universidad:
Por abrirme sus puertas y la formación profesional que recibí.

A todos, gracias…

Rojas Cortés Victor

A mis Padres Pedro y Margarita:

Por darme la vida y la oportunidad de haber terminado una carrera gracias a sus desvelos, sacrificios y arduo trabajo, estoy muy orgullos de ustedes y los quiero con toda mi alma.

A mi Esposa Nathali:

Por haber dejado sus actividades a un lado para darme todo su amor y apoyo incondicional, te amo hermosa.

A mis Hermanos Neto, Evita, Fer, Raquelito y Silvita:

Por todo su cariño, comprensión, apoyo y protección como mis hermanos mayores que son, los quiero mucho.

A mis Sobrinos Caro, Chino, Opayo, Dianita, Adrianita y Beto:

Que les sirva como un ejemplo a seguir, los quiero mucho y cuentan con todo mi apoyo.

A Dios:

Por haberme dado la mejor de todas las familias y darme día a día la fuerza necesaria para conseguir mis objetivos y metas.

Rosas Ramírez Pedro

Agradezco primeramente a mis papas, por ser los mejores y estar conmigo incondicionalmente, gracias porque sin ellos y sus enseñanzas no estaría aquí ni sería quien soy ahora, a ellos les dedico esta tesis.

A mi abuelo y mis hermanos por ser la mejor familia que me pudo haber tocado.

A mi abuela, sé que me vez y estas orgullosa de mí.

Gracias a todos!!
Gracias por ayudarme a lograrlo.
Los quiero mucho

Tovar Pacheco Julio César

ÍNDICE

INTRODUCCIÓN

Esta tesis surge de la necesidad que actualmente tienen las agencias de autos para cotizar y emitir pólizas de una forma automática y en tiempo real.

De esta forma se propone un sistema que integre toda la información para ofrecer la cotización de pólizas de diferentes aseguradoras, procurando al cliente la mejor opción de acuerdo a su auto y necesidad, obteniendo como resultado la impresión de la póliza elegida.

Con este sistema se espera obtener una mejor administración del servicio de cotización y emisión de pólizas, reduciendo tiempo, costos y aumentando la productividad, además de garantizar la satisfacción del cliente.

Esta tesis se desarrolla en 4 capítulos donde se hará mención de conocimientos teóricos adquiridos en la facultad, así como también mencionaremos conocimientos de experiencia laboral.

En el primer capítulo se mencionan los fundamentos teóricos que respaldan el desarrollo del sistema, con el fin de tener las bases necesarias para poder determinar los alcances del sistema y las herramientas a utilizar.

En el segundo capítulo se describen las distintas formas en que las agencias de autos cotizan y emiten pólizas, identificando las deficiencias del servicio de este servicio y partiendo de esto se propone un sistema que solucione esta problemática, así mismo se establecen los requerimientos de dicho sistema y los objetivos que se persiguen al momento de desarrollarlo.

En el tercer capítulo se evalúan herramientas de desarrollo de software, eligiendo la que mejor se adapte a los requerimientos del sistema para su elaboración y por consiguiente tener un sistema actualizado.

En el cuarto y último capítulo se elabora el sistema de acuerdo a la metodología RUP. Se describen los pasos uno por uno, los cuales son análisis, diseño, codificación y pruebas.

Esta tesis aparte de solucionar un problema y cubrir una necesidad, se busca que sirva de ejemplo para proyectos posteriores y que se utilice como una referencia o guía para poder cubrir otras necesidades de la vida real.

CAPÍTULO 1. MARCO TEÓRICO

Dentro de este capítulo describiremos la metodología RUP, el análisis, diseño y programación orientada a objetos, el modelo entidad-relación y la tecnología .NET que son los conocimientos previos necesarios para poder abordar nuestro problema.

1.1 METODOLOGÍA RUP

Rational Unified Process - RUP (Proceso Unificado Racional) uno de los procesos más generales de modelo de software, cuyo objetivo es producir software de alta calidad, cumpliendo con los requerimientos de los usuarios dentro de una planificación y presupuesto establecidos.

Aunque RUP se concibió en gran medida para el desarrollo de sistemas basados en programación orientada a objetos, por ejemplo se suele emplear en proyectos de programación en lenguajes como Java, Visual Basic.NET, etc., y el cual es una gran herramienta puesto que trabajaremos con lenguajes orientados a objetos.

RUP tiene tres características esenciales: está dirigido por los Casos de Uso, está centrado en la arquitectura, y es iterativo e incremental.

1.1.1 Proceso Dirigido por Caso de Usos.

En RUP los Casos de Uso no son sólo una herramienta para especificar los requisitos del sistema. También guían su diseño, implementación y prueba. Los Casos de Uso constituyen un elemento integrador y una guía del trabajo.

Como se muestra en la Figura 1 los Casos de Uso no sólo inician el proceso de desarrollo sino que nos proporcionan un hilo conductor, permitiéndonos establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.

Figura 1. Los Casos de Uso integran el trabajo

Basándose en los Casos de Uso se crean los modelos de análisis y diseño, luego la implementación que los lleva a cabo, y se verifica que efectivamente el producto implemente adecuadamente cada Caso de Uso. Todos los modelos deben estar sincronizados con el modelo de Casos de Uso como se ejemplifica en la Figura 2

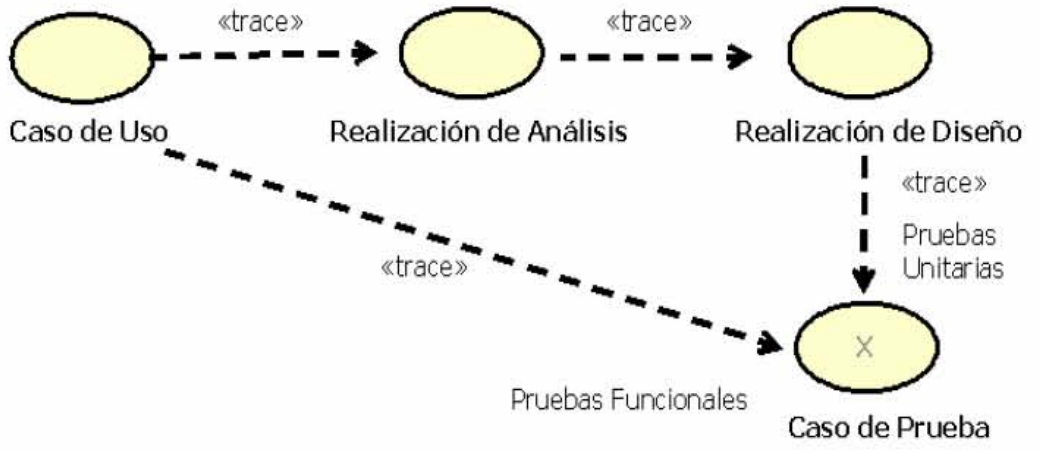


Figura 2. Trazabilidad a partir de los Casos de Uso.

1.1.2 Proceso Centrado en la Arquitectura.

En RUP además de utilizar los Casos de Uso para guiar el proceso se presta especial atención al establecimiento temprano de una buena arquitectura que no se vea fuertemente impactada ante cambios posteriores durante la construcción y el mantenimiento. Existe una interacción entre los Casos de Uso y la arquitectura, los Casos de Uso deben encajar en la arquitectura cuando se llevan a cabo y la

arquitectura debe permitir el desarrollo de todos los Casos de Uso requeridos, actualmente y en el futuro. Esto provoca que tanto arquitectura como Casos de Uso deban evolucionar en paralelo durante todo el proceso de desarrollo de software.

En la Figura 3 se ilustra la evolución de la arquitectura durante las fases de RUP. Se tiene una arquitectura más robusta en las fases finales del proyecto. En las fases iniciales lo que se hace es ir consolidando la arquitectura por medio de baselines y se va modificando dependiendo de las necesidades del proyecto.

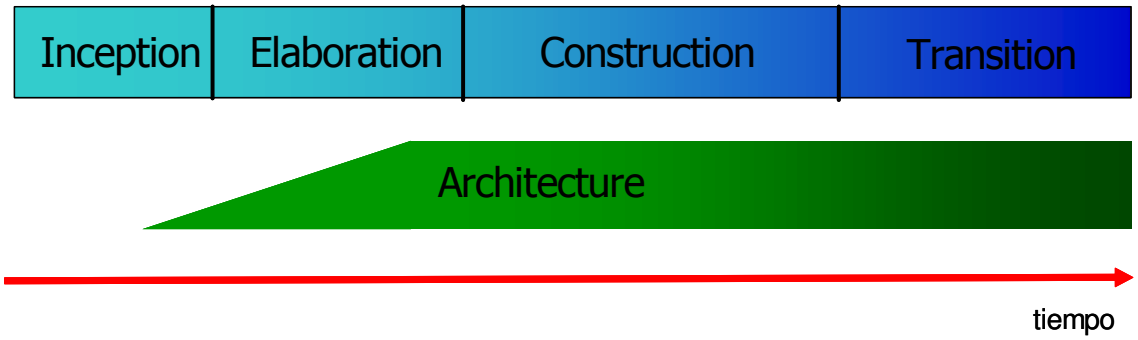


Figura 3. Evolución de la arquitectura del sistema.

Es conveniente ver el sistema desde diferentes perspectivas para comprender mejor el diseño por lo que la arquitectura se representa mediante varias vistas que se centran en aspectos concretos del sistema, abstrayéndose de los demás. Para RUP, todas las vistas juntas forman el llamado modelo 4+1 de la arquitectura, el cual recibe este nombre porque lo forman las vistas lógica, de implementación, de proceso y de despliegue, más la de Casos de Uso que es la que da cohesión a todas.

1.1.3 Proceso Iterativo o Incremental.

La estrategia que se propone en RUP es tener un proceso iterativo e incremental en donde el trabajo se divide en partes más pequeñas o mini proyectos. Permitiendo que el equilibrio entre Casos de Uso y arquitectura se vaya logrando durante cada mini proyecto, así durante todo el proceso de desarrollo. Cada mini proyecto se puede ver como una iteración (un recorrido más o menos completo a lo largo de todos los flujos de trabajo fundamentales) del cual se obtiene un incremento que produce un crecimiento en el producto.

Una iteración puede realizarse por medio de una cascada como se muestra en la Figura 4. Se pasa por los flujos fundamentales (Requisitos, Análisis, Diseño, Implementación y Pruebas), también existe una planificación de la iteración, un análisis de la iteración y algunas actividades específicas de la iteración. Al finalizar se realiza una integración de los resultados con lo obtenido de las iteraciones anteriores.

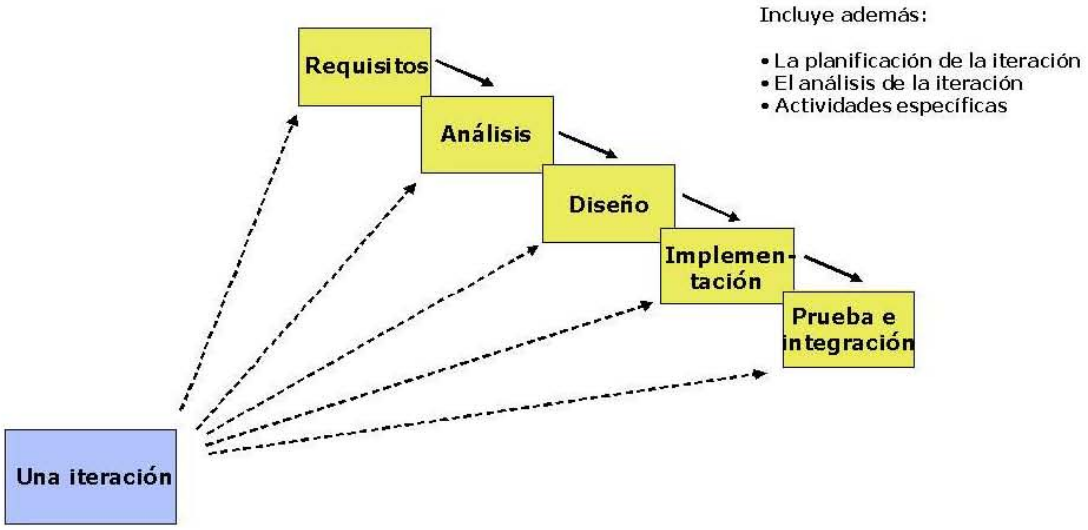


Figura 4. Una iteración RUP.

RUP divide el proceso en cuatro fases, dentro de las cuales se realizan varias iteraciones en número variable según el proyecto y en las que se hace un mayor o menor hincapié en los distintas actividades.

1. Concepción
2. Elaboración (definición, análisis, diseño)
3. Construcción (implementación)
4. Transición (fin de proyecto y puesta en marcha)

Cada fase se concluye con puntos de revisión como se muestra en la Figura 5. Es importante resaltar que la inclusión de puntos de revisión, es sumamente importante y en estos puntos se revisan los requerimientos establecidos para cada fase, basados en los controles de calidad. De esta manera, si un producto o proceso no pasa el punto de revisión de calidad, se rediseña o se cancela, evitando así, los problemas de costo extra, de reconstrucción, y de productos de mala calidad.

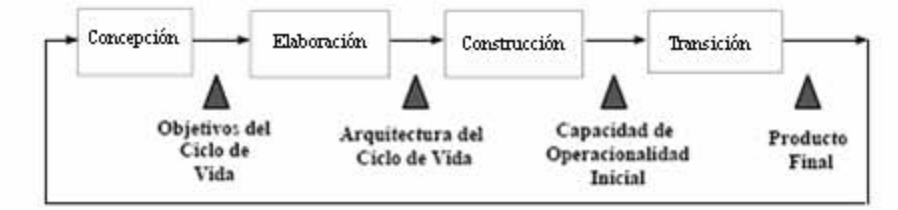


Figura 5. Fases de Relational Unified Process (RUP).

Durante la fase de inicio se define el modelo del negocio y el alcance del proyecto. Se identifican todos los actores y Casos de Uso, y se diseñan los Casos de Uso más esenciales (aproximadamente el 20% del modelo completo). Se desarrolla, un plan de negocio para determinar que recursos deben ser asignados al proyecto.

En la fase de elaboración, se construye un prototipo de la arquitectura, que debe evolucionar en iteraciones sucesivas hasta convertirse en el sistema final. Este prototipo debe contener los Casos de Uso críticos identificados en la fase de inicio. También debe demostrarse que se han evitado los riesgos más graves.

En la fase de construcción, la finalidad principal es alcanzar la capacidad operacional del producto de forma incremental a través de las sucesivas iteraciones. Durante esta fase todos los componentes, características y requisitos deben ser implementados, integrados y probados en su totalidad, obteniendo una versión aceptable del producto.

En la fase de transición la finalidad es poner el producto en manos de los usuarios finales, para lo que se requiere desarrollar nuevas versiones actualizadas del producto, completar la documentación, entrenar al usuario en el manejo del producto, y en general tareas relacionadas con el ajuste, configuración, instalación y facilidad de uso del producto.

Como se puede observar en cada fase participan todas las disciplinas, pero que dependiendo de la fase el esfuerzo dedicado a una disciplina varía. RUP toma en cuenta las mejores prácticas en el modelo de desarrollo de software en particular las siguientes:

- Desarrollo de software en forma iterativa (repite una acción).
- Administración de requerimientos.
- Utiliza arquitectura basada en componentes.
- Modela el software visualmente (Modela con UML)
- Verifica la calidad del software.
- Controla los cambios.

La duración y esfuerzo dedicado en cada fase es variable dependiendo de las características del proyecto. Sin embargo, la Tabla 1.1.1 ilustra porcentajes frecuentes al respecto.

	Inicio	Elaboración	Construcción	Transición
Esfuerzo	5 %	20 %	65 %	10%
Tiempo Dedicado	10 %	30 %	50 %	10%

Tabla 1. Distribución típicas de esfuerzo y tiempo.

Por tal motivo la metodología RUP es un proceso que nos facilita el modelado y desarrollo de proyectos de software con lenguajes orientados a objetos.

1.2. ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS.

1.2.1 Introducción a la tecnología de objetos.

El diseño orientado a objetos modela el software en términos similares a los que utilizan las personas para describir objetos del mundo real. Este diseño aprovecha las relaciones entre las clases, en donde los objetos de cierta clase tienen las mismas características. También aprovecha las relaciones de herencia, e incluso las relaciones de herencia múltiple, en donde las nuevas clases de objetos se derivan absorbiendo las características de las clases existentes y agregando sus propias características únicas.

El Diseño orientado a objetos (DOO) ofrece una manera más natural e intuitiva de ver el proceso de diseño: a saber, modelando los componentes de software de igual forma que como describimos los objetos del mundo real (por sus atributos y comportamientos).
El DOO también modela la comunicación entre los objetos. El DOO encapsula los atributos y las operaciones (comportamiento) en los objetos; los atributos y las operaciones de un objeto se enlazan íntimamente entre sí.

Reutilizar las clases existentes cuando se crean nuevas clases y programas es un proceso que ahorra tiempo y esfuerzo. La reutilización también ayuda a los programadores a crear sistemas más confiables, ya que las clases y componentes existentes a menudo han pasado por un proceso extenso de prueba y depuración.

Evidentemente, con la tecnología de objetos podemos crear la mayor parte del software que necesitaremos mediante la combinación de clases, que son "partes intercambiables.

1.2.2 Introducción al análisis y diseño orientados a objetos (A/DOO).

Para crear las mejores soluciones de análisis y diseño de software, se debe seguir un proceso detallado para obtener un análisis de los requerimientos de su

proyecto, y para desarrollar un diseño que cumpla con esos requerimientos. Idealmente, un programador pasaría por este proceso y revisaría el diseño antes de escribir el código para su proyecto. Si este proceso implica analizar y diseñar su sistema desde un punto de vista orientado a objetos, lo llamamos un proceso de análisis y diseño orientado a objetos (A/DOO).

Los programadores experimentados saben que, sin importar qué tan simple parezca un problema, el análisis y el diseño pueden ahorrar innumerables horas que podrían perderse al abandonar, en plena implementación, un método de desarrollo de un sistema mal planeado.

A/DOO es el término genérico para el proceso de analizar un problema y desarrollar un método para resolverlo. Los pequeños problemas no requieren de un proceso exhaustivo. Podría ser suficiente con escribir pseudocódigo antes de empezar a escribir el código. El pseudocódigo es un medio informal de expresar el código de un programa. En realidad no es un lenguaje de programación, pero podemos usarlo como un tipo de "bosquejo" para guiamos a medida que escribimos nuestro código.

El pseudocódigo puede ser suficiente para los problemas pequeños, pero a medida que éstos y los grupos de personas que los resuelven se incrementan en tamaño, los métodos de A/DOO se vuelven más necesarios. Idealmente, un grupo debería acordar un proceso estrictamente definido para resolver su problema, y acordar también una manera uniforme para que los miembros del grupo se comuniquen los resultados de ese proceso entre sí. Aunque existen muchos procesos de A/DOO distintos, existe un lenguaje gráfico para comunicar los resultados de cualquier proceso A/DOO que se ha vuelto muy popular. Este lenguaje se conoce como Lenguaje Unificado de Modelado (UML). UML se desarrolló a mediados de la década de los noventa, bajo la dirección inicial de tres metodologistas de software: Grady Booch, James Rumbaugh e Ivar Jacobson.

1.2.3 Lenguaje Unificado de Modelado.

El UML de sus siglas Unified Modeling Language (Lenguaje Unificado de Modelado) es un lenguaje gráfico que permite a las personas que crean sistemas (por ejemplo, arquitectos de software, ingenieros de sistemas, programadores, etcétera.) representar sus requerimientos, análisis y diseños orientados a objetos mediante el uso de una notación común.

El UML es ahora el esquema de representación gráfica más utilizado para modelar sistemas orientados a objetos. Evidentemente ha unificado los diversos esquemas de notación populares. Aquellos quienes diseñan sistemas utilizan el lenguaje (en forma de diagramas) para modelar sus sistemas.

Una de las características más atractiva de UML, es su flexibilidad. UML, es extensible e independiente de los diversos procesos de A/DOO, esto aunado con la Programación Orientada a Objetos podemos obtener buenos resultados.

1.3 PROGRAMACIÓN ORIENTADA A OBJETOS.

1.3.1 Características de la Programación Orientada a Objetos.

La programación Orientada a Objetos (POO) es una forma especial de programar, más cercana a como expresaríamos las cosas en la vida real que otros tipos de programación como la programación estructurada que utiliza un número de estructuras de control determinado que hacen rebuscada la programación y lenta.

Actualmente una de las áreas más utilizadas en la industria y en el ámbito académico es la orientación a objetos. La orientación a objetos promete mejoras de amplio alcance en la forma de diseño, desarrollo y mantenimiento del software ofreciendo una solución a largo plazo a los problemas y preocupaciones que han existido desde el comienzo en el desarrollo de software: la falta de portabilidad del código y reusabilidad, código que es difícil de modificar, ciclos de desarrollo largos y técnicas de codificación no intuitivas. La POO no es difícil, pero es una manera especial de pensar, a veces subjetiva de quien la programa, de manera que la forma de hacer las cosas puede ser diferente según el programador. Aunque podamos hacer los programas de formas distintas, no todas ellas son correctas, lo difícil no es programar orientado a objetos sino programar bien y solo así podemos aprovechar todas las ventajas de la Programación Orientada a Objetos.

En la POO se definen los programas en términos de clases de objetos, los cuales que son entidades que combinan estado, comportamiento e identidad, es decir datos, procedimientos o métodos y propiedad del objeto que lo diferencia del resto.

La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. De esta forma nuestro sistema de emisión y cotización de pólizas quedará perfectamente identificada cada tarea con un clase correspondiente a su objeto que pertenezca

permitiendo que los módulos del sistema en general sean más fáciles de escribir, mantener y reutilizar.

De esta forma, un objeto que si bien contiene toda la información, y por medio de sus atributos permite definirlo e identificarlo frente a otros objetos pertenecientes a otras clases e incluso entre objetos de la misma clase, al poder tener valores bien diferenciados en sus atributos. A su vez, dispone de mecanismos de interacción por medio de los llamados métodos que favorecen la comunicación entre objetos ya sean de una misma clase o de distintas, y en consecuencia, el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separa la información del procesamiento.

Los lenguajes orientados a objetos primero deben definirse los objetos con datos y métodos y después envían mensajes a los objetos diciendo que realicen esos métodos por si mismos a diferencia de los lenguajes de programación imperativos y estructurados que piensan todo en términos de procedimientos y estructuras, escriben funciones y después les pasan datos de entrada con el único objetivo de obtener datos de salida.

1.3.2 Ventajas de la Programación Orientada a Objetos.

Las principales diferencias que hacen la programación orientada a objetos la mejor de las opciones son las siguientes:

- La programación orientada a objetos es más moderna, es una evolución de la programación imperativa plasmada en el diseño de una familia de lenguajes conceptos que existían previamente, con algunos nuevos.
- La programación orientada a objetos se basa en lenguajes que soportan sintáctica y semánticamente la unión entre los tipos abstractos de datos y sus operaciones.

- La programación orientada a objetos incorpora en su entorno de ejecución mecanismos tales como el polimorfismo y el envío de mensajes entre objetos.

La POO se adapta fácilmente para al modelo entidad relación el cual se adoptará para diseñar la base de datos de nuestro sistema.

1.4 Base de Datos.

El planteamiento de un esquema conceptual correcto para el alojamiento de la información manejada por el sistema dentro de una base de datos es positiva para el desarrollo de una aplicación funcionalmente viable, por lo que es importante tener un buen modelado de nuestros datos que nos permita transformar los parámetros del mundo real en abstracciones que permiten entender los datos sin tener en cuenta la física de los mismos.

En si el modelo conceptual se encargará de reflejar las relaciones entre los datos y relaciones en la base de datos. Dentro de los diferentes modelos conceptuales (E/R, RM/T y semántico), tomaremos la decisión de hacer uso del E/R, ya que es la base para pasar al modelo relacional de Frank Codd .

1.4.1 Modelo Entidad-Relación.

El modelo Entidad-Relación (E/R), es también conocido como Entidad-Asociación o Entidad-Relación-Atributo y propone mediante el uso de conceptos abstractos como entidad, relación y atributos permitir describir un conjunto de datos relativos a un dominio definido dentro de una base de datos.

Los componentes que formarán nuestro modelo E/R serán los siguientes:

- Entidad: Es cualquier elemento u objeto tanto abstracto como real, poseedor de múltiples propiedades y del cual se pueda almacenar información en la base de datos.
- Relaciones: Es la representación de las relaciones entre las entidades, permitiéndonos relacionar los datos del modelo.
- Atributos: Es la descripción de las entidades y relaciones, es decir lo que le da algún calificativo a una entidad. Alguno de los atributos que puede fungir como identificador ya que sus valores son únicos en cada ejemplar de una entidad.

- Cardinalidad: Es el número de entidades a las que se puede asociase una entidad a través de un conjunto de relaciones y pueden ser de uno a uno, de uno a varios o varios a varios.

Como habíamos comentado anteriormente, después de obtener el diagrama E/R ya podemos estar listos para la obtención del modelo relacional de nuestra base de datos.

Un modelo relacional considera a una base de datos como una colección de relaciones, cada relación representa una tabla constituidas por filas y columnas. En las tablas cada línea nos representa una colección de datos que nos describen a una entidad del mundo real

Dentro del modelo relacional se utiliza los siguientes conceptos generales:

- Relación. Elemento fundamental en el modelo relacional, mejor conocido como tabla.
- Tupla. Cada fila de la tabla (cada ejemplar que la tabla representa).
- Atributo. Cada columna de la tabla.
- Grado. Número de atributos de la tabla.
- Cardinalidad. Número de filas en una tabla.

Estos conceptos tienen distintos sinónimos utilizados de acuerdo a la nomenclatura designada, las cuales pueden ser (ver Tabla 2):

Nomenclatura Relacional	Nomenclatura Tabla	Nomenclatura Ficheros
Relación	Tabla	Fichero
Tupla	Fila	Registro
Atributos	Columna	Campo
Grado	Número de columnas	Número de campos
Cardinalidad	Número de filas	Número de registros

Tabla 2. Diferentes nomenclaturas utilizadas en un modelo relacional.

Para pasar de nuestro modelo E/R a un modelo relacional tomaremos en cuenta lo siguiente:

- Las entidades pasan a ser tablas.
- Los atributos pasan a ser columnas.
- Los identificadores principales pasan a ser claves primarias.
- Los identificadores candidatos pasan a ser claves candidatas.

Ya obtenido el esquema relacional a partir del modelo E/R que representaba la base de datos normalmente se tiene una buena base de datos, pero en ocasiones debido a fallas en el diseño y problemas indetectables en la fase del diseño, tendremos un modelo que tenga los siguientes problemas:

- Redundancia. Se llama así a los datos que se repiten continua e innecesariamente por las tablas de las bases de datos.
- Ambigüedades. Datos que no clarifican suficientemente el registro al que representan.
- Problemas en el mantenimiento de los datos actualizándolos, insertándolos y borrándolos.

- Dificultad para reestructurar o reorganizar los datos cuando surjan nuevas aplicaciones.

La manera para deshacernos de estos problemas, es mediante la normalización, la cual se encarga de simplificar la relación entre los campos de un registro.

1.4.2 Normalización del Modelo Entidad-Relación.

El procedimiento de normalización consiste en someter a las tablas que representan entidades a un análisis formal para ver si cumplen, o no, las restricciones necesarias que aseguren evitar los problemas citados con anterioridad. A mayor nivel de normalización, mayor calidad en la organización de los datos y menor peligro para la integridad de los datos. Este procedimiento consiste en ir alcanzando formas normales

Normalizar la información es un proceso complejo que bien logrado brinda beneficios importantes al desempeño de consultas de datos y al aprovechamiento del espacio disponible en disco.

El proceso de normalización se lleva a partir del nivel de normalización cero porque ninguna de nuestras reglas de normalización ha sido aplicada, hasta el quinto nivel de normalización, esto dependiendo de la funcionalidad que le queramos de dar a nuestra estructura de datos o aplicación.

Otra estructura importante dentro de nuestro modelo relacional son las llaves primarias y foráneas dentro de nuestras relaciones que conforman nuestro modelo. La llave primaria es un conjunto de atributos que nos permiten identificar unívocamente una tupla en una relación. Las llaves foráneas son una combinación de atributos de una relación que son, a su vez, una llave primaria para otra relación, algo que caracteriza a esta llave es que de los valores que presenta a no ser que sean null, tienen que corresponder a valores existentes en la llave primaria de la relación a la que se refieren.

1.5 TECNOLOGÍA .NET

Con la tecnología .NET la información y las aplicaciones, ya no estarán en nuestra PC, sino en la Red.

1.5.1 Plataforma .NET.

La idea central de la plataforma .NET es la de software como servicio y de cómo construir, instalar, consumir, integrar o agregar estos servicios para que puedan ser accedidos mediante Internet. Esto es posible debido a que tenemos la infraestructura de comunicación que es Internet cada vez mas rápida y a un costo cada vez menor y además, a la capacidad de los procesadores que continúa incrementándose año tras año.

En la figura 6 Podemos visualizar como están relacionados los componentes que integran la plataforma .NET

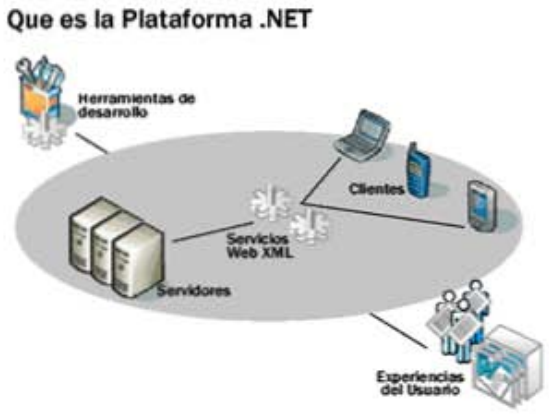


Figura 6. Componentes de .NET.

Los componentes que integran esta plataforma se mencionan en la tabla 3.

Componentes	Descripción	Ejemplo
Herramientas de desarrollo	Interfaces de programación y herramientas para diseñar, crear, ejecutar e instalar soluciones basadas en .NET.	.NET Framework. Visual Studio .NET
Servidores	Infraestructura para construir, instalar y operar la plataforma .NET	Windows 2000 Server
Servicios Web XML	Servicios que implementan tareas que son accedidos desde Internet.	.NET Passport
Clientes	Dispositivos físicos que corren en sistemas operativos que integran e interactúan con otros elementos .NET.	Windows CE. Windows XP para PCs
Experiencia del usuario	Software tradicional que se integra con los servicios Web para presentar todo lo que el usuario necesita.	El sitio Microsoft MSN

Tabla 3. Descripción de componentes .NET.

1.5.2 Arquitectura de la plataforma .Net.

La arquitectura .NET o mejor conocido como .NET Framework, es el modelo de programación para construir y ejecutar los servicios .NET, constituye la base de la plataforma .NET y denota la infraestructura sobre la que se reúne todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones. Mediante esta herramienta se ofrece un entorno de ejecución altamente distribuido, que permite crear aplicaciones robustas y escalables.

En la figura 7 se pueden apreciar las distintas partes que componen al .NET Framework las cuales son, entorno de ejecución de aplicaciones, el conjunto de bibliotecas de funcionalidad reutilizable y los compiladores y herramientas de desarrollo para los lenguajes .NET

Figura 7. Componentes de .NET Framework.

Todos estos componentes se montan por encima de la familia de sistemas operativos Windows.

1.5.3 Common Language Runtime (CLR).

El CLR es el núcleo del .Net Framework, es el entorno de ejecución en el que se cargan las aplicaciones desarrolladas en los distintos lenguajes, ampliando el conjunto de servicios que ofrece el sistema operativo estándar Win32.

En la figura 8 podemos apreciar los distintos componentes internos que constituyen el CLR, cada uno de los cuales tiene un propósito específico.

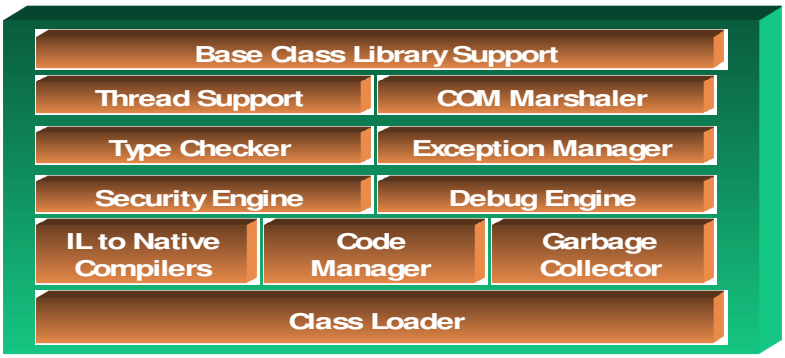


Figura 8. Elementos de CLR.

En la Tabla 4 se describen las características de los elementos del CLR.

Elemento	Descripción
Class Loador	Permite cargar en memoria las clases.
IL to Native Compilers	Transforma código intermedio de alto nivel independiente del hardware que lo ejecuta a código de máquina propio del dispositivo que lo ejecuta.
Code Manager	Coordina toda la operación de los distintos subsistemas del CLR.
Garbage Collector	Elimina de memoria objetos no utilizados.
Security Engine	Administra la seguridad del código que se ejecuta.
Debug Engine	Permite hacer un seguimiento de la ejecución del código aún cuando se utilicen lenguajes distintos.
Type Checker	Controla que las variables de la aplicación usen el área de memoria que tiene asignado.
Exception Manager	Maneja los errores que se producen durante la ejecución del código.
Thread Support	Permite ejecutar código en forma paralela.
COM Marshaler	Coordina la comunicación con los componentes COM para que puedan ser usados por el .NET Framework.
Base Class Library Support	Interfaz con las clases base del .NET Framework.

Tabla 4. Descripción de los elementos del CLR.

1.5.4 NET Framework Class Library.

Dentro del conjunto de la .NET Framework Class Library se distinguen 4 sub-componentes principales:

- Base Class Library: contiene la funcionalidad más comúnmente utilizada para el desarrollo de todo tipo de aplicaciones. Algunos ejemplos son el manejo de colecciones, cadenas de texto, entrada/salida, threading, operaciones matemáticas y dibujos 2D.

- ADO.NET: contiene un conjunto de clases que permiten interactuar con bases de datos relacionales y documentos XML como repositorios de información persistente.
- ASP.NET: constituye la tecnología para construir aplicaciones con interfaz de usuario Web.
- Windows Forms: permite crear aplicaciones con interfaz de usuario basada en formularios y ventanas Windows de funcionalidad eficiente y que se ejecutan directamente en los clientes.

1.5.5 Common Language Specification (CLS)

Uno de los objetivos de diseño de la plataforma .NET fue el ser independiente del lenguaje de programación elegido para el desarrollo de aplicaciones. Para lograr esto es que se creó la especificación de lenguaje común, que define y estandariza un subconjunto de todas las características soportadas por el CLR y que son necesarias en la mayoría de las aplicaciones. Todos los componentes desarrollados y compilados de acuerdo con la especificación CLS pueden interactuar entre si, independientemente del lenguaje de programación de alto nivel en el que fueron escritos.

1.5.6 Lenguajes

.Net Framework soporta múltiples lenguajes de programación y aunque cada lenguaje tiene sus características propias, es posible desarrollar cualquier tipo de aplicación con cualquiera de estos lenguajes. Existen más de 30 lenguajes adaptados a .Net, desde los más conocidos como C# (C Sharp), Visual Basic o C++ hasta otros lenguajes menos conocidos como Perl o Cobol.

1.5.7 Modelo de ejecución de .NET

El desarrollo de una aplicación .NET comienza con la escritura de su código fuente en alguno de los lenguajes de alto nivel soportados por la plataforma. Posteriormente es compilado obteniéndose un ejecutable o una biblioteca. A estos

componentes .NET resultantes del proceso de compilación se los denomina genéricamente Assemblies, o Ensamblados.

Los assemblies contienen un código denominado MSIL (Microsoft Intermediate Language). EL MSIL es un conjunto de instrucciones independientes de cualquier CPU existente y que puede ser convertido a código nativo muy eficiente. MSIL incluye instrucciones para cargar, almacenar, inicializar e interactuar con objetos y sus atributos y métodos, así como también instrucciones aritméticas y lógicas, control de flujo, acceso directo a memoria, manejo de errores y otras operaciones. Antes de que el código MSIL pueda ser ejecutado debe convertirse a código nativo específico para un CPU y Sistema Operativo, tarea a cargo de los compiladores JIT incluidos en el CLR. Lo anterior descrito se observa en la figura 9.

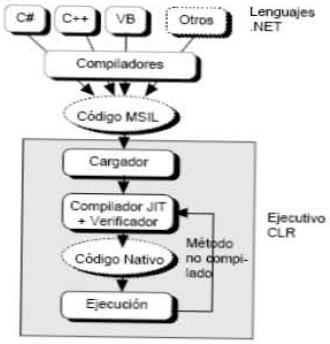


Figura 9. Proceso de compilación.

1.5.8 Ensamblados

El .Net Framework maneja un nuevo concepto denominado ensamblado. Los ensamblados son ficheros con forma de EXE o DLL que contienen toda la funcionalidad de la aplicación de forma encapsulada. Por tanto la solución al problema puede ser tan fácil como copiar todos los ensamblados en el directorio de la aplicación.

Con los ensamblados ya no es necesario registrar los componentes de la aplicación. Esto se debe a que los ensamblados almacenan dentro de si mismos

toda la información necesaria en lo que se denomina el manifiesto del ensamblado. El manifiesto recoge todos los métodos y propiedades en forma de meta-datos junto con otra información descriptiva, como permisos, dependencias, etc.

En el siguiente capítulo veremos los requerimientos del sistema comenzando por el planteamiento del problema, posteriormente propondremos una solución y en base a esto plantearemos los objetivos generales y particulares.

CAPÍTULO 2. REQUERIMIENTOS DEL SISTEMA

En el desarrollo de este capítulo se hará mención de los problemas que se presentan en una agencia de autos para ofrecer al cliente la emisión y cotización de una póliza de seguros, y nuestra propuesta tecnológica para dar solución a esos problemas, considerando la situación actual y el alcance que puede tener nuestro desarrollo.

2.1 PLANTEAMIENTO DEL PROBLEMA.

Uno de los problemas fundamentales con los que se enfrentan las agencias de venta de automóviles es que se ven en la necesidad de obtener la cotización de un seguro de automóvil por parte de varias aseguradoras para ofrecer al cliente una mayor diversidad de tarifas.

Ya existen sistemas que realizan la función de cotizar y emitir pólizas de seguros, uno diferente en cada agencia o punto de venta, y la forma de obtención y generación de la información no siempre resulta la más adecuada.
El mantenimiento y actualización de la información para cada punto de venta o agencia es una tarea muy complicada y que se tiene que llevar a cabo periódicamente para poder ofrecer al cliente información veraz a su solicitud.

Cuando no se manejan sistemas de este tipo en las agencias o puntos de venta, la cotización y emisión de pólizas, se realiza de la manera más tardada, mediante intermediarios o agentes de las agencias aseguradoras.

Todos estos movimientos provocan: retrasos en el tiempo de respuesta del cliente para obtener la cotización y emisión de su seguro, o que los datos que le fueron proporcionados al cliente resulten no estar actualizados, o la pérdida o corrupción de la información. Todos estos problemas se traducen en una insatisfacción del cliente con respecto al servicio.

2.1.1 Situación actual.

Hoy en día, existen sistemas que realizan la cotización y emisión de pólizas de seguros, pero la generación de información se hace de manera independiente y esto propicia manejar una base de datos por cada agencia, lo cual hace que la administración de la información se vuelva complicada y laboriosa. Para que una agencia pueda hacer uso de este sistema, es necesario implementárselo en su negocio, por lo que cada vez que hay un cambio al mismo, hay que acudir a actualizarlo.

Todo esto ha provocado el aumento de costos en la actualización del sistema en las agencias, y como consecuencia, ha generado la insatisfacción de los clientes por el tiempo de respuesta a su solicitud.

Además, en algunos de los casos, la cotización de ventas de seguros se hace de la forma tradicional, llenando a mano la papelería y trasladando la documentación a las centrales donde se lleva a cabo la administración de seguros; así también, se puede hacer de manera telefónica. Esto ocasiona una mala administración de la información, pérdida de la misma en el traslado y errores de captura y ortográficos que causan pérdida de tiempo y mayor número de incidencias. Con lo anterior señalado, el sistema se vuelve complicado y requiere de un mayor esfuerzo para lograr incrementar los niveles de satisfacción del cliente.

Las anteriores problemáticas podemos observarlas en la Figura 10.

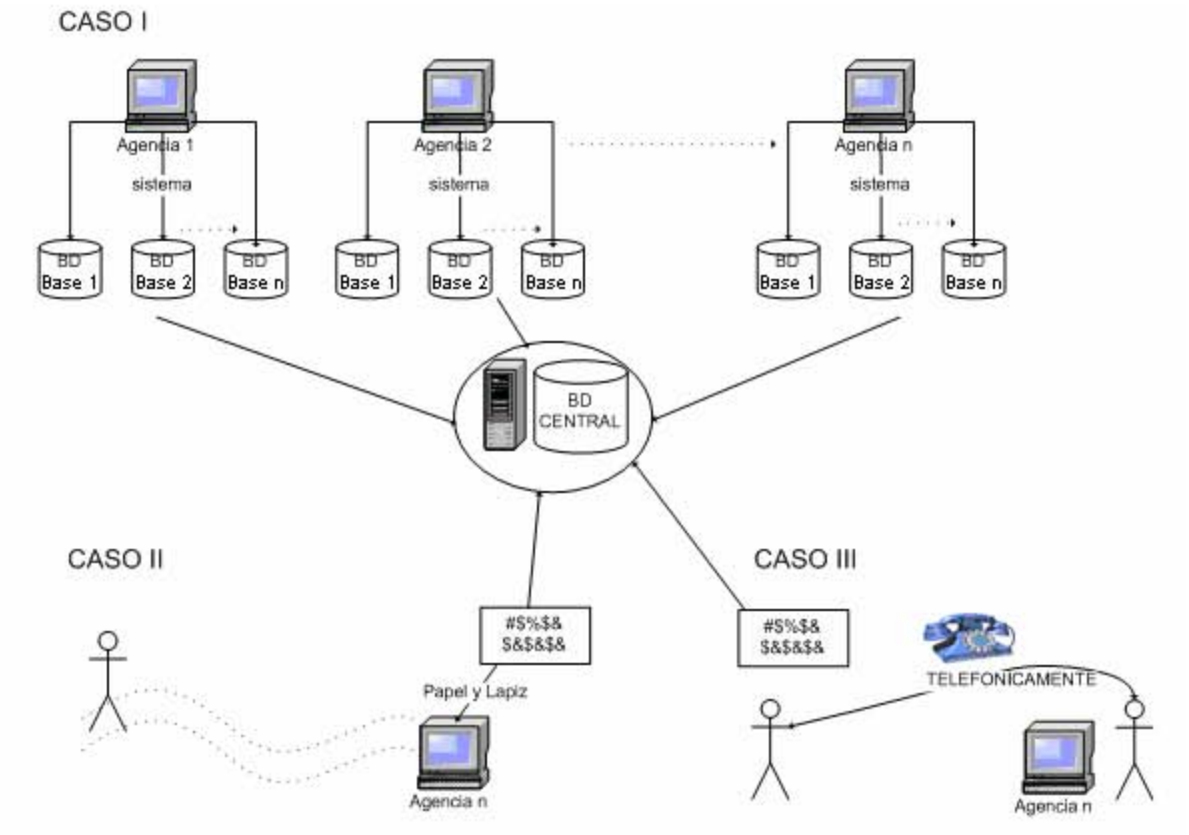


Figura 10. Problemática Actual

2.2 PROPUESTA DE SOLUCIÓN

De acuerdo a nuestra problemática actual y considerando el extenso mercado que tienen los seguros de autos, se propone desarrollar un sistema vía Web con una base de datos central que administre las pólizas de diferentes aseguradoras y agencias, evitando sistemas independientes y pérdida de información. De esta forma si una agencia o punto de venta requiere la cotización de una póliza para determinado automóvil y modelo, utilizara Internet para obtener acceso a la Base de Datos Central y consultar la información necesaria sin necesidad de acudir personalmente y de hacer esperar mucho tiempo al cliente, esta propuesta se ilustra en la Figura 11.

Figura 11. Propuesta de solución.

Con esta propuesta de solución las agencias o puntos de venta tendrán grandes ventajas, empezando por omitir el llenado de las solicitudes que tradicionalmente se utilizan, el ahorro en llamadas realizadas diariamente a las agencias de otra ciudad y aseguradoras, pero el ahorro más significativo será de tiempo que traducido esto en productividad, al no demorar tanto en la cotización de una póliza por consiguiente se captará mayor número de solicitantes diariamente.

2.2.1 Necesidades de un nuevo sistema.

En la actualidad existen aproximadamente en el mercado de seguros de México 38 Compañías aseguradoras, esto hace que el cliente debe consultar vía telefónica o personalmente cada una de las compañías que le interese. La idea de desarrollar un nuevo sistema capaz de cotizar y emitir pólizas en un tiempo muy corto y de una forma sencilla es debido a que los innumerables agentes de seguros hacen hasta lo imposible por atraer el dinero a la cuenta de la aseguradora para la que trabajen, sin importar las vueltas que hacen dar a las personas para acudir una y otra vez a consultar el estado de su cotización y de su póliza.

Ya existen algunas aseguradoras, bancos nacionales y sitios de Internet que han puesto a disposición de sus clientes el servicio de cotización de pólizas para ahorrar tiempo y darle entera libertad de decidirse por el seguro que más le convenga, sin embargo muchos clientes desconocen el manejo de Internet o tienen mucha inseguridad de ingresar datos personales y sean interceptados por personas ajenas a los sistemas, debido a ello siempre es necesario de que un asesor y que mejor la persona que les vende el carro les pueda cotizar el seguro más adecuado a sus necesidades.

Partiendo de esta premisa nuestro sistema permitirá conocer el monto del contrato que asegurará un vehículo, de una manera muy sencilla, el asesor deberá entrar al sitio Web autentificándose con usuario y contraseña válidos, una vez dentro del mismo podrá ingresar los datos como la marca, submarca, modelo, y plazo por el que requiere la cobertura entre otros datos más del vehículo o de la ciudad donde circula, posteriormente se solicitará la cotización de la póliza y en ese mismo momento se le otorgara al cliente para su análisis, si decide ahí mismo sobre alguna de las opciones, se procederá hacer la emisión respectiva.

2.2.2 Planteamiento del sistema.

El sistema tendrá como filosofía "desde la perspectiva del cliente". Este se hará con una aplicación que podrá ser accedido mediante un navegador para Internet y hará uso de una base de datos central, como se muestra en la figura 12.

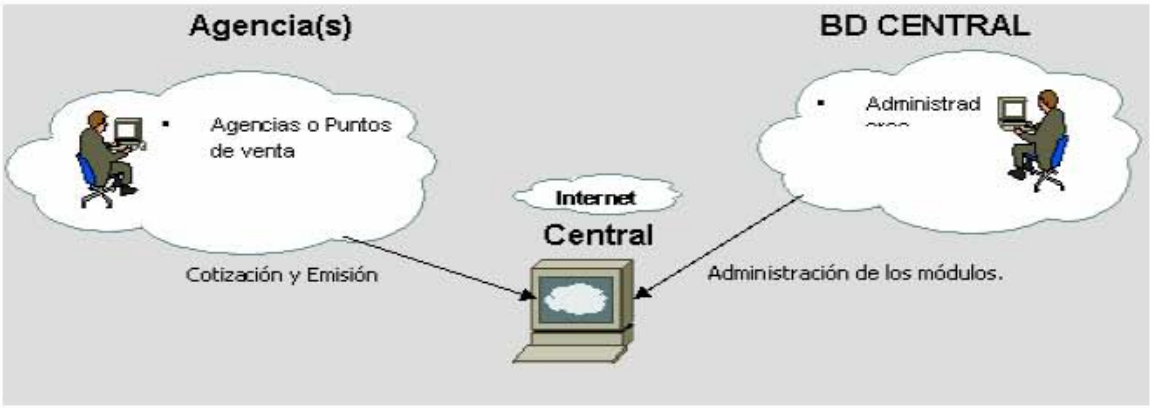


Figura 12. Solución Planteada.

Diagrama de flujo.

Los procesos generales que se tienen que hacer son:

a) Validar Usuario
b) Cotizar vehículos
c) Emitir póliza seleccionada
d) Imprimir póliza.
e) Reimprimir póliza
f) Consultar póliza emitida.

Estos procesos se muestran en la figura 13.

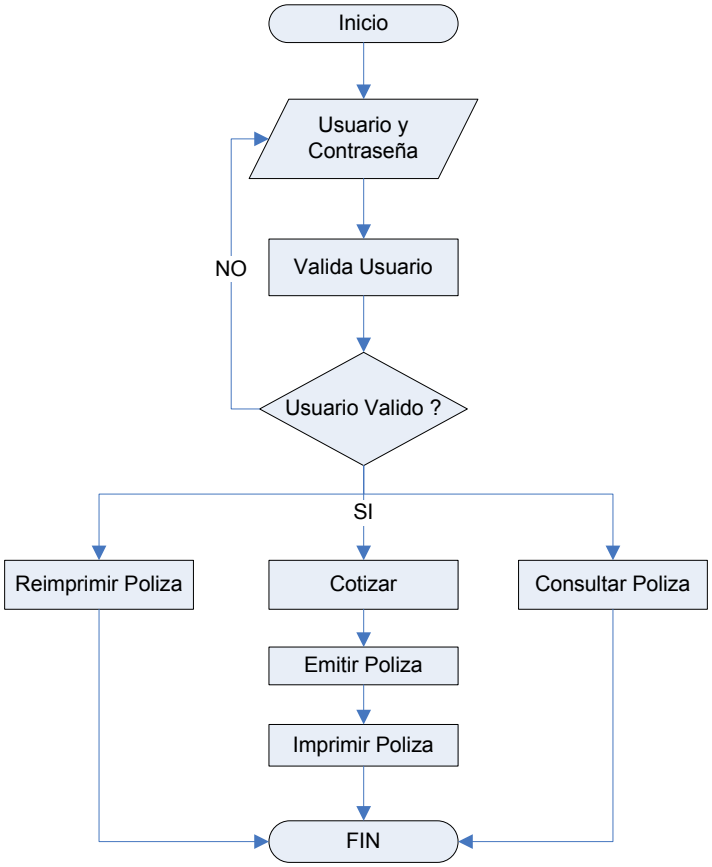


Figura 13. Diagrama de flujo del sistema a desarrollar.

2.2.3 Requerimientos del sistema.

- El sistema se debe acceder desde Internet.
- Debe tener una interfaz amigable.
- Para poder hacer uso del sistema los usuarios deben de firmarse, es decir con su respectivo usuario y contraseña, con esto tendremos una mayor seguridad para poder hacer uso de la aplicación.
- Tratar que el proceso se desarrolle en 3 pasos. Cotizar, emitir e imprimir póliza.
- Utilizar dos pantallas de captura de datos para facilitar el proceso.
- Los usuarios deben de formar parte de un perfil.

- Los módulos que debe de contener el sistema son: Cotización, dentro de éste estará contemplado la cotización y emisión respectivamente.
- Consultas y reimpresión de pólizas.
- De acuerdo al perfil del usuario se debe de mostrar los módulos y submódulos respectivos.
- Se debe mostrar el nombre del usuario que esta accediendo, así como la agencia a la cual pertenece.
- Un botón de salir en caso de que ya no se quiera hacer uso del sistema, para eliminar la sesión del usuario y tener mayor seguridad.
- La aplicación debe de conservar sesiones activas que no sobrepasen más de 20 minutos, de estar inactiva, si no es así el usuario debe de registrarse de nueva cuenta.
- El sistema debe de concentrar la información de tal manera que ésta este disponible desde cualquier lugar y en cualquier momento ya sea para emitir, consultar o reimprimir una póliza.
- El sistema sea flexible para actualizar información de vehículos.
- Mostrar en una tabla las combinaciones de cotización, por forma de pago, por paquete y por aseguradora.
- En cada cotización se debe de mostrar las coberturas que cubren dicha cotización.
- Que puedan elegir sólo una opción de las cotizaciones presentadas.
- Posibilidad de imprimir un formato de cotización personalizado.
- Introducir sumas aseguradas correctas.
- No permitir que se dupliquen pólizas emitidas.
- Respaldar la base de datos por lo menos una vez a la semana.
- Garantizar cálculos correctos para las primas totales de las pólizas.
- Para los campos de fechas mostrar un calendario para elegir el día requerido
- Los campos no deben de permitir el ingreso de caracteres especiales, sólo letras y números
- Validar la información que se introdujo en los campos.

La impresión de la póliza contendrá:

- Datos del cliente.
- Datos del vehículo.
- Datos de la póliza
- Datos de las coberturas.
- Datos de la aseguradora.
- Datos de las primas totales.

Considerando la problemática planteada, la infraestructura que va a tener el sistema es una aplicación Web de arquitectura cliente-servidor, con un modelo de aplicación de servicio como se muestra en la figura 14.

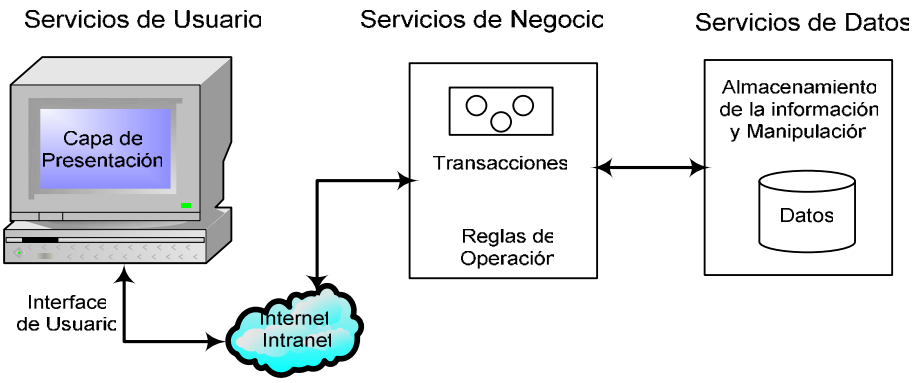


Figura 14. Aplicación de servicios.

El término aplicación de servicio significa que la funcionalidad de la aplicación está definida como un conjunto de servicios (usuario, negocio, datos) que cubren las necesidades del usuario.

Los beneficios de usar el modelo de aplicación de servicios son los siguientes:

1. Desarrollo de metas claras y consistentes: Dividir los servicios dentro un sitio WEB, permite que el desarrollo sea más fácil. La funcionalidad de cada servicio se define claramente.

2. Mejor Manejo en los cambios: Debido a que los servicios dividen la funcionalidad del sitio WEB en distintas tareas, cualquier cambio en la aplicación de un servicio no introduce cambios a los componentes del servicio.
3. Aislamiento de funcionalidad: La funcionalidad de un servicio específico esta encapsulada, si hay un error, se puede tratar fácilmente el componente correspondiente de la aplicación del servicio.
4. División de labores: Identificar servicios, permite determinar cuáles miembros del desarrollo WEB se tienen que construir bien y por completo.

El modelo de servicios a utilizar será una aplicación WEB de tres capas, como se muestra en la figura 15, ya que es la que mejor se adapta a las prácticas orientadas a objetos.

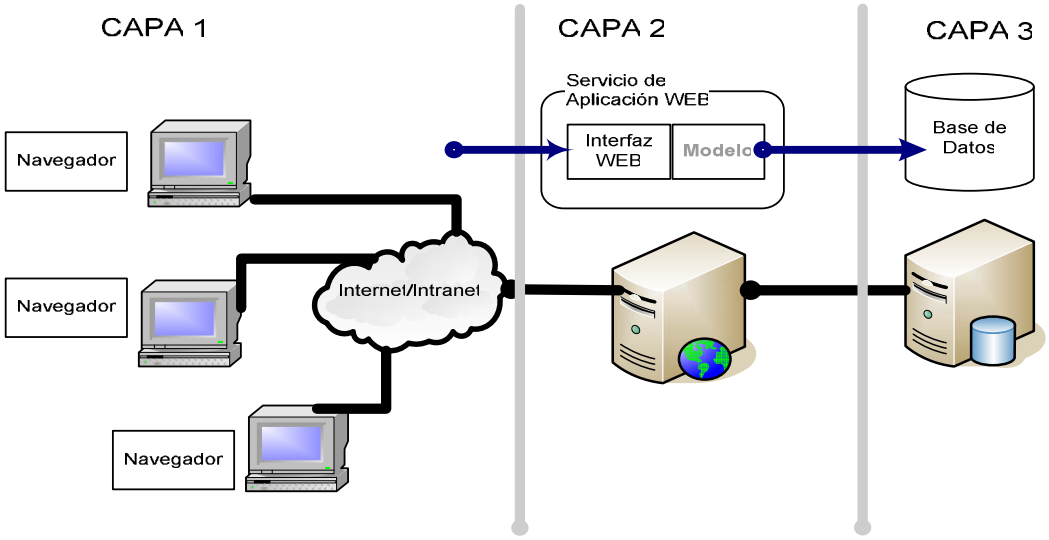


Figura 15. Arquitectura de 3 capas.

La capa 1 del modelo de aplicación de servicios la constituye el cliente que es propiamente el usuario que cuenta con una interfaz Web. El que se encarga de definir el formato de los datos es el servidor de la aplicación. Éste construye una página Web y se la entrega al cliente por medio de su navegador.

La capa 2 contiene los servicios Web y de negocios. Esta capa interactúa entre la capa 3 de servicios de datos para obtener información y, la capa 1 que brinda servicios al cliente.

La capa 3 contiene los servicios de datos. Éste proporciona servicios y almacena la información utilizada en el sistema.

2.3 OBJETIVOS.

Durante el desarrollo de este trabajo tenemos que alcanzar un conjunto diverso de objetivos a cumplir, tanto generales como particulares.

El número de objetivos propuestos dependerán en gran medida de la funcionalidad que le logremos dar a nuestro sistema, así como del uso que le hayamos dado a los conocimientos académicos que se hayan aplicado durante el desarrollo de éste.

2.3.1 Objetivo General.

Desarrollar un sistema seguro, confiable, flexible, portable y eficaz que permita a las agencias de autos obtener una mejor administración para cotizar y emitir pólizas de las diferentes aseguradoras con las cuales tengan convenio, reduciendo tiempo, costos y aumentando la productividad, además de garantizar la satisfacción del cliente.

2.3.2 Objetivos Particulares.

Los objetivos que se desean alcanzar al terminó de este trabajo son:

1. Analizar los requerimientos en lo referente a la vinculación entre los servicios prestados por una Agencia de Autos y sus clientes, proponiendo el diseño y la implementación de un sistema que facilite y agilicé la cotización y emisión de pólizas de seguros para automóvil de dicha agencia.
2. Desarrollo de software que permita a la agencia de autos la facilidad y rapidez de cotizarle y emitir a sus clientes el seguro para su automóvil, mediante la diferente gama de aseguradoras con las que tiene convenio la agencia de autos, así como de sus servicios de cada uno sus costos, políticas de operación de las mismas y de los tramites que cada una realiza para la emisión de sus pólizas y así satisfacer las necesidades del cliente y de la agencia de autos.

3. Analizar las diferentes etapas del proceso de cotización y emisión de pólizas, y seleccionar el mejor método de modelado de procesos para el desarrollo del proyecto.
4. Analizar y seleccionar el lenguaje de programación que sea el mejor para el desarrollo del software, así como la base de datos y la plataforma en la que se trabajara dicho sistema y con el cual se haga un sistema seguro, confiable, robusto y de fácil manejo.

5. Analizar los resultados de las pruebas para comprobar que los resultados obtenidos del sistema de cotización y emisión de pólizas corresponda con los resultados esperados.
6. Elaboración de un informe con las recomendaciones pertinentes, para mejorar el modelado de usuario y las interfaces del sistema de cotización y emisión de pólizas, de acuerdo a los resultados obtenidos de las pruebas aplicadas.
7. Los conocimientos adquiridos durante la carrera puedan verse reflejados en el desarrollo del sistema.

Una vez que ha quedado clara la propuesta del sistema a desarrollar, así como los objetivos que se persiguen, el siguiente paso es realizar un análisis de las herramientas que actualmente se encuentran en el mercado y nos servirán para su elaboración, tratando de elegir las más apropiadas para nuestro objetivo.

CAPÍTULO 3. ANÁLISIS DE HERRAMIENTAS

De la gran diversidad de herramientas que actualmente se utilizan para el desarrollo de software se escogieron las que más se adecuaron a las necesidades y requerimientos del sistema, a continuación se muestran.

3.1 ENTORNO DE DESARROLLO (IDE).

De los diferentes Entornos Integrados de Desarrollo (IDE) que existen en el mercado seleccionamos a tres de los más usados y robustos que hay actualmente, esto con el fin de ver alguna de sus características y después de un análisis funcional decidir cual es el más conveniente para cumplir con los objetivos planteados en este trabajo.

IDE	**Características**
Visual Studio .NET	• Desarrollado por Microsoft para SO Windows. • Soporta los lenguajes .NET. • ASP.NET para aplicaciones Web. • Usa CIL como compilador. • Cuenta con un recolector de basura para liberar la memoria que no ocupemos. • Herramienta diseñada para crear aplicaciones conectadas a Microsoft.NET para Windows y web.
JDeveloper	• Desarrollado por Oracle para lenguaje Java. • JSP para aplicaciones Web. • Cuenta con un recolector de basura. • Se puede instalar en Windows, Linux y Mc OS. • Se requiere instalar su compilador JREJDK.
Delphi	• Desarrollado por Code Gear. • Entorno diseñado en especial para la programación visual. • Multiplataforma.

Tabla 5. Tabla comparativa de Entornos Integrados de Desarrollo (IDE).

Analizando los diferentes IDE de la Tabla 5, seleccionamos Visual Studio .NET, el cual soporta los nuevos lenguajes .NET: C#, Visual Basic .NET y Managed C++, además de C++. Visual Studio .NET puede utilizarse para construir aplicaciones dirigidas a Windows (utilizando Windows Forms), Web (usando ASP.NET y Servicios Web) y dispositivos portátiles (utilizando .NET Compact Framework).

En .NET se dice que el código está administrado. Puesto que lo que realmente se ejecuta es código intermedio, dicha ejecución esta *vigilada* de forma que no pueda realizar acciones inapropiadas.

En otras palabras Visual Studio .NET es una IDE creada para realizar aplicaciones para el sistema operativo Windows, el cual es el sistema operativo más comercial a nivel nacional y con el cual trabajan la mayoría de las agencias de autos.

Además .NET incorpora un recolector de basura. Esto significa que no deberemos preocuparnos de liberar manualmente la memoria que vayamos dejando de utilizar ya que el recolector de basura (Garbage Collector o GC en inglés) se encarga de monitorizar toda la memoria utilizada, decidir si dicha memoria es accesible de alguna forma y, en caso de no serlo, disponer de ella de la forma más adecuada.

También .NET incorpora un sistema llamado CAS (Code Access Security o Seguridad de Acceso a Código) que permite restringir los permisos de ejecución de un determinado código de una forma pormenorizada. Esto significa que podemos definir que un determinado segmento de código (asociado a un determinado rol) tan sólo tenga acceso de lectura a tres archivos completos.

Así mismo podremos también configurar que nuestra librería, que proporciona una serie de servicios proporcione distinta accesibilidad dependiendo del nivel de acceso del programa llamante (por ejemplo pudríamos evitar que una aplicación

llamante cree nuevos documentos en función de su nivel de acceso, o sólo pueda ver determinados documentos.

Con esta plataforma Microsoft incursiona de lleno en el campo de los Servicios Web y establece el XML como norma en el transporte de información en sus productos y lo promociona como tal en los sistemas desarrollados utilizando sus herramientas.

Todo esto nos facilita y nos permite trabajar mucho mejor en el desarrollo de cualquier proyecto, aunque no sea una IDE multilenguaje puede soportar y trabajar con la mayoría de los lenguajes de programación más actuales, así como de poder utilizar con facilidad las librerías de Windows y con esto poder adaptar perfectamente el sistema en el ambiente Windows sin ningún contratiempo.

La cual es una herramienta útil para el desarrollo de nuestro proyecto, puesto que el lenguaje de programación que se utilizará para el desarrollo del sistema es Visual Basic .NET y como plataforma para las aplicaciones Web usaremos ASP.NET.

3.2. LENGUAJES DE PROGRAMACIÓN.

Los lenguajes de programación para nuestro sistema se analizaron de acuerdo a la visión y métodos que utilizan para la construcción de un programa o subprograma, además que cubriera con las necesidades de nuestro sistema.

El sistema de emisión y cotización de pólizas requiere una actualización constante y dinámica, que los módulos estén perfectamente definidos y que las tareas sean específicas para cada uno de ellos además de existir una comunicación entre todos los módulos, con la finalidad de no repetir información y poder reutilizar el código de algún módulo en alguna otra parte del sistema, de esto se definió trabajar con la programación orientada a objetos.

Entre los lenguajes de programación orientados a objetos más importantes que se pueden mencionar, aparecen los siguientes:

Ada, C++, C# , VB.NET, Clarion, Delphi , Eiffel , Java ,Lexico, Objective-C, Ocaml, Oz, PHP, PowerBuilder, Pitón, Ruby,Smalltalk.

De estos lenguajes se analizaron dos de ellos para obtener el mejor que se adapta a las necesidades de nuestro sistema, Visual Basic.Net y Java.

3.2.1 Visual Basic.Net.

Visual Basic.NET es una versión de Visual Basic enfocada al desarrollo de aplicaciones .NET, es totalmente un lenguaje Orientado a Objetos sus creadores son Microsoft, sus características más importantes son:

- Diseño de controles de usuario para aplicaciones Windows y Web
- Programación de bibliotecas de clase.
- Envió de datos vía documentos XML.
- Generación de reportes basados en Crystal Reports a partir de información obtenida de orígenes de datos (archivos de texto, bases, etc.)

Para las aplicaciones Web de todo sistema programado en Visual Basic.Net una tecnología que incrementa la seguridad y el mejor desarrollo es ASP.NET

3.2.2 ASP.NET.

Active Server Pages (ASP) es un lenguaje de scripts que permite ser integrado con clases .NET.

Dentro de ASP.NET cada objeto corresponde a un grupo de funcionalidades frecuentemente usadas y útiles para crear páginas Web dinámicas además estas páginas pueden ser generadas mezclando código de scripts del lado del servidor (incluyendo acceso a base de datos) con HTML.
Los servicios Web XML que ofrece permiten el intercambio de datos en escenarios cliente-servidor o servidor-servidor, utilizando estándares como los servicios de mensajería HTTP y XML para que los datos pasen los servidores de seguridad.

3.2.3 Java.

Por el otro extremo se encuentra Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems.

A diferencia de los lenguajes de programación convencionales, que generalmente están diseñados para ser compilados a código nativo, Java es compilado en un bytecode (código intermedio) que es ejecutado, por una máquina virtual Java usando un compilador JIT (just-in-time) es decir compilación en tiempo de ejecución, sus características se pueden resumir en la siguiente lista:

- Compatibilidad de navegadores Web para la ejecución de applets en el cliente.
- Posee la herramienta JavaServer para desarrollo de sitios Web.
- Utiliza de un recolector de basura para eliminar de forma automática aquellos objetos no requeridos.
- Utiliza una máquina virtual lo que hace que sea multiplataforma.

Java utiliza del lado del servidor el lenguaje JSP que permite crear aplicaciones Web de una forma dinámica.

3.2.4 JSP.

Java Pages Server (JSP), es una tecnología para crear aplicaciones Web dinámicas, es un desarrollo también de la compañía Sun Microsystems, y su funcionamiento se basa en scripts, que utilizan una variante del lenguaje java.
En las JSP, se escribe el texto que va a ser devuelto en la salida (normalmente código HTML) incluyendo código java dentro de él para poder modificar o generar contenido dinámicamente.

La principal ventaja de JSP frente a otros lenguajes es que permite integrarse con clases Java (.class) lo que permite separar en niveles las aplicaciones web, almacenando en clases java las partes que consumen más recursos, así como las que requieren más seguridad, y dejando la parte encargada de formatear el documento html en el archivo jsp.

3.2.5 Análisis de los Lenguajes de Programación.

Después de haber analizado las dos opciones por un lado Visual Basic.Net en conjunto con ASP.NET y del mismo modo Java integrado con JSP, se tomo la decisión de optar por la primera opción, VB.NET con ASP.NET, por las siguientes desventajas que encontramos en Java y JSP.

- Las aplicaciones Java pueden necesitar gran cantidad de memoria física.
- La apariencia de las fuentes no tiene las opciones de optimización activadas por defecto, lo que hace aparecer al texto como si fuera de baja calidad.
- Hay varias versiones del Entorno en Tiempo de Ejecución de Java, el JRE. Es necesario tener instalada la versión adecuada. El paquete JRE puede

ser de tamaño considerable, 7Mbytes, lo que puede ser un inconveniente a la hora de descargarlo e instalarlo.

- Las herramientas con que cuenta el JDK no son suficientemente potentes para construir de forma simple aplicaciones potentes.
- JSP no es un script 100% ya que antes de ejecutarse el servidor Web compila el script y genera un servlet, por lo tanto, se puede decir que aunque este proceso sea transparente para nosotros no deja de ser un aplicación compilada lo que a nuestro sistema no le conviene.

Por el contrario las características que nos convencieron de la opción elegida son las siguientes:

- En Visual Basic.NET su interfaz gráfica hace más sencilla la programación.
- Genera reportes basados en Crystal Reports justo lo que necesitamos para las cotizaciones y emisiones de pólizas.
- Tiene un entorno gráfico muy amigable.
- Por medio de aplicaciones ASP.NET se puede acceder a bases de datos para mostrar la información a los encargados de las agencias que coticen y emitan las pólizas.
- Por medio de ASP´s permite administrar la base de datos desde el código.
- También podemos almacenar mediante programación cualquier objeto o conjunto de datos en la memoria del servidor, esto nos beneficia de tal forma que nuestras aplicaciones puedan ahorrar el tiempo y los recursos necesarios para volver a crearlos.

A continuación analizaremos los manejadores de bases de datos más acordes a nuestro sistema y además que sean compatibles con los lenguajes de programación elegidos.

3.3. MANEJADORES DE BASES DE DATOS.

La selección del Sistema Manejador de Base de Datos (DBMS), tiene mucha importancia, al igual que la elección del lenguaje utilizado para desarrollar el sistema como se vio en el anterior tema.

El DBMS es el encargado de realizar las siguientes tareas:

- Almacenar, obtener y modificar los datos.- Será la interfaz entre los datos almacenados en bajo nivel y los programas de aplicación.
- Implantación de la integridad.- Asegurar que los datos permanezcan íntegros, esto quiere decir que no se pierdan o se cambien.
- Interactuar con el manejador de archivos.- Crear y modificar archivos donde se almacenarán los datos.

3.3.1 Análisis de Manejadores de Bases de Datos

En las tablas 5, 6, 7, 8 y 9 se comparan diversos manejadores de bases de datos en cuanto a la seguridad, ambiente gráfico, costo, facilidad de instalación y compatibilidad con Windows.

Seguridad de la Base de Datos	Muy Bueno	Bueno	Regular
MySQL 4.0		X	
Oracle 7.0	X		
Microsoft SQL Server 2000	X		

Tabla 5. Tabla comparativa de la seguridad de los DBMS.

Ambiente Gráfico	Muy Bueno	Bueno	Regular
MySQL 4.0		X	
Oracle 7.0		X	
Microsoft SQL Server 2000	X		

Tabla 6. Tabla comparativa del ambiente gráfico de los DBMS.

Disponibilidad y Costo de la Herramienta	Muy Bueno	Bueno	Regular
MySQL 4.0	X		
Oracle 7.0			X
Microsoft SQL Server 2000			X

Tabla 7. Tabla comparativa de la disponibilidad y costo del DBMS.

Facilidad de Instalación	Muy Bueno	Bueno	Regular
MySQL 4.0	X		
Oracle 7.0			X
Microsoft SQL Server 2000	X		

Tabla 8. Tabla comparativa sobre la facilidad para instalar el DBMS.

Compatibilidad con Windows	Muy Bueno	Bueno	Regular
MySQL 4.0	X		
Oracle 7.0	X		
Microsoft SQL Server 2000	X		

Tabla 9. Tabla comparativa de la compatibilidad con Windows de los DBMS.

Tomando de base las características evaluadas sobre los manejadores de datos con los que se cuenta, se ha decidido que el sistema constará de una base de datos en Microsoft SQL Server 2000, porque consideramos que para el desarrollo e implementación del sistema necesitaremos de:

- Escalabilidad: Esto para tener adaptación a las necesidades de la empresa, soportando desde unos pocos usuarios a varios cientos.
- Gestión: Esto por contar con una completa interfaz gráfica que reduzca la complejidad innecesaria de las tareas de administración y gestión de la base de datos.
- Compatibilidad: Esto por ser compatible con la totalidad de los productos y herramientas de Microsoft.

Microsoft SQL Server 2000 tiene la ventaja de acoplarse a los recursos de hardware con los que contamos ya que requiere las siguientes características mostradas en la tabla 10.

Procesador	DEC Alpha y sistemas compatibles Intel® o compatible (Pentium 166 MHz o superior, Pentium PRO o Pentium II)
Espacio en Disco Duro	250 MB (completa) 50 MB (sólo herramientas de administración)
Memoria RAM	Recomendada de 128 MB
Unidad de disco	Unidad de CD-ROM.
Sistema Operativo	Windows 95 ó 98, Windows Millennium Edition (Windows Me), Windows XP Professional, Windows XP Home Edition, Windows 2000 Professional.

Tabla 10. Requerimientos de Microsoft SQL Server 2000

3.4. ARQUITECTURA DE SISTEMAS.

Cuando la tecnología hace uso del Internet es importante distribuir las distintas tareas en distintos equipos para garantizar el uso eficiente de los recursos informáticos. Esta distribución es lo que denominamos arquitectura.

Se mencionaran las características de las que consideramos que nos pueden ayudar a cubrir nuestras necesidades y escogeremos la más óptima para el desarrollo del sistema para alcanzar el nivel de servicio deseado con el menor costo posible.

3.4.1 Arquitectura de Dos Capas.

La arquitectura de dos capas es conocida como Cliente – Servidor debido a que un "cliente" solicita servicios a un "servidor". Este servidor es a su vez el encargado de moderar las solicitudes de varios clientes a la vez.

El servidor contiene sólo el servicio de acceso a la base de datos y los clientes tienen la lógica de aplicación y la interfaz de usuario.

Las capas que esta arquitectura presenta son las mostradas en la figura 16 y se describen a continuación:

Capa de Aplicación: aquí se encuentra toda la interfaz del sistema y es la que el usuario puede disponer para realizar su actividad con el sistema.

Capa de Base de Datos: también llamado el Repositorio de Datos, es la capa en donde se almacena toda la información ingresada en el sistema

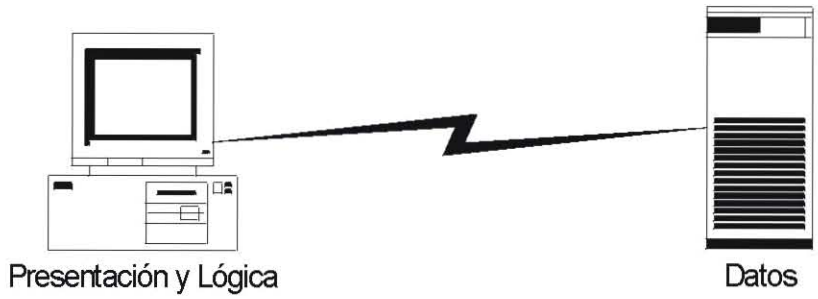


Figura 16. Arquitectura de dos capas.

Características:

- Mucha carga en el cliente.
- Poca carga en el servidor.
- Mucho tráfico en la red.
- Mantenimiento costoso, en cada cliente
- Las reglas del negocio quedan dispersas entre el nivel de aplicación y los procedimientos de la base de datos.

3.4.2 Arquitectura de Tres Capas.

La idea es separar la lógica de la aplicación del cliente. Para esto se construye una "capa" entre el servidor de base de datos y el cliente.
La nueva capa es el modelo de la aplicación y esta pide servicios a la base de datos y da servicios al cliente.

Las capas que esta arquitectura presenta son las mostradas en la figura 17.

Capa de Presentación: la diferencia de este nivel aplicado ahora en una arquitectura de tres capas es que sólo tiene que trabajar con la semántica propia de aplicación, sin tener que preocuparse de cómo esta implementado éste, ni de su estructura física.

Capa de Aplicación: se encarga de toda la estructura física y el dominio de aplicación.

Algo muy importante y que es la mayor ventaja de esta arquitectura es que ahora únicamente se cambia la regla en el servidor de aplicación y ésta actuará en todos los clientes, cosa que ni sucedía con la arquitectura en dos capas que si alguna regla se la cambia, se tenía que ir a cada cliente a realizar el cambio.

Capa de Base de Datos: sigue siendo la capa en donde se almacenan los datos y toda la información.

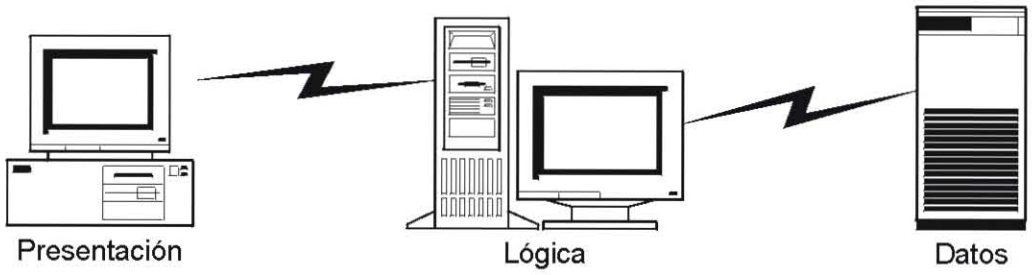


Figura 17. Arquitectura de tres capas.

Características:

Posibilidad de crear diferentes interfaces para la misma lógica de negocio.

No se puede hacer un repositorio de datos distribuido, un ejemplo de este tipo de repositorio es el de los bancos que necesitas de un repositorio de datos distribuido para poder tener sus sucursales en otros países y ciudades.

3.4.3 Arquitectura de N Capas.

Se utiliza para sistemas distribuidos. Consiste en dividir la funcionalidad del sistema total en capas lógicas que pueden ser encapsuladas como componentes que interactúan entre ellos a alto nivel.

Las capas son las siguientes y se muestran en la figura 18:

Capa de Presentación: no hace ningún cálculo o actualizaciones sobre el dominio, ni siquiera tiene una visualización sobre la capa de dominio. Es el responsable dar formato y generar las páginas para mostrar la información al usuario.

Capa de Aplicación: es la encargada de los cálculos, las actualizaciones y el acceso a la capa de dominio, esta interactúa con la capa de presentación y con capa de dominio para que se tenga un contacto indirecto entre estas dos capas.

En sí, esto es la diferencia básica de la arquitectura de tres capas y la de n capas, la capa del dominio y la capa de base de datos cumplen el mismo papel solo que esta ves la capa de datos puede estar distribuido en varios partes, esto facilita mucho al sistema para su distribución.

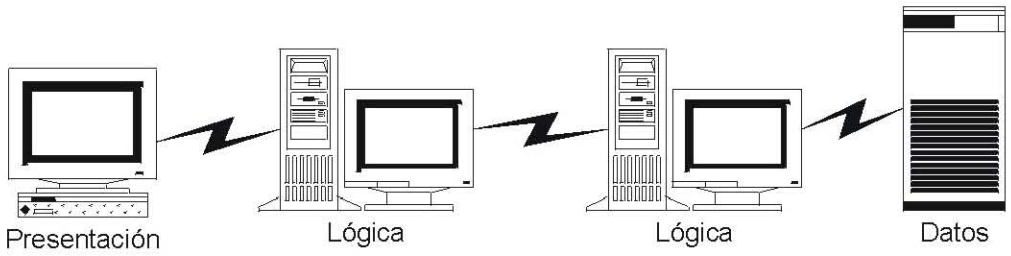


Figura 18. Arquitectura n Capas.

El costo de mantenimiento del sistema disminuye ya que una modificación en una capa no debe afectar a las demás. Con esto tenemos menos tiempos de desarrollo y por lo tanto menos costo. La contraparte es que se tiene que invertir en servidores para separar físicamente las capas, para tener un rendimiento satisfactorio.

3.4.4 Análisis de Arquitecturas.

Tomando en consideración las características de las arquitecturas mencionadas, la de tres capas es la que mejor se adapta a nuestros recursos físicos y lógicos así como a las prácticas orientadas a objetos, facilitando la reusabilidad del software.

Al conectarnos a Internet estamos navegando en 3 capas, al abrir un formulario Web de inscripción (capa de presentación), después de enviar la información esta es verificada (capa de negocios), finalmente la información es grabada en una base de datos (capa de datos).

Acoplando la arquitectura a lo que es la tecnología .NET el esquema queda representado como se muestra en la figura 19.

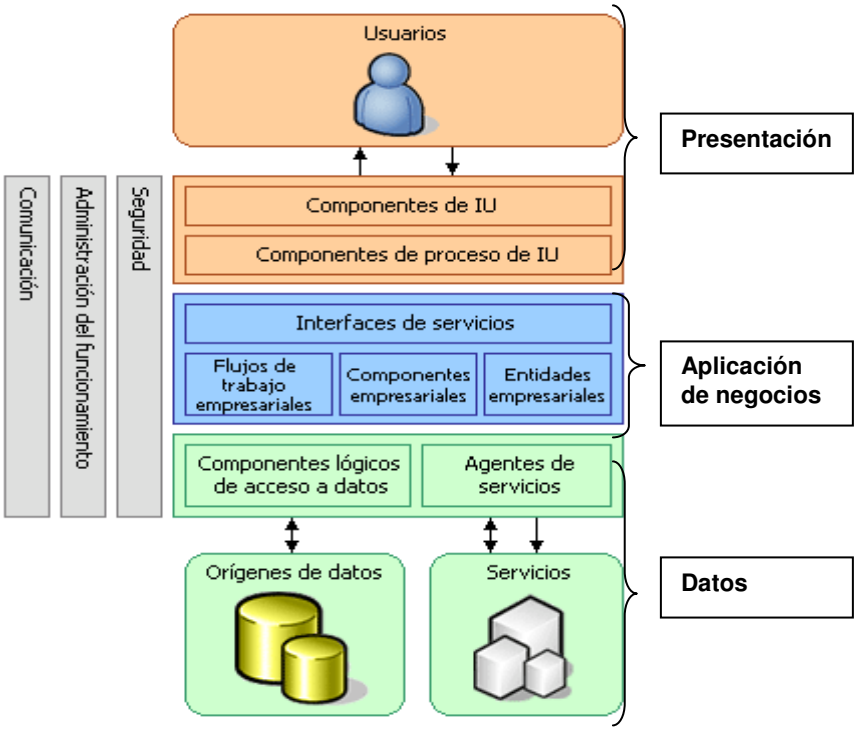


Figura 19. Arquitectura de 3 capas en .NET

1. La capa de presentación: Los componentes de IU pueden ser vistos. Las ventanas o páginas Web. Componentes de Procesos. Es decir, estos encapsulan lógica de navegación y control de eventos de la interfase.

2. La capa de negocios: Los servicios de esta capa son encapsulados en tres tipos de componentes. Las entidades empresariales, que representan objetos que van a ser manejados por toda la aplicación, estos podrían ser un modelo de objetos, datasets con tipo, estructuras de datos, que permitan representar objetos

que han sido identificados durante el modelo. Los otros tipos de objetos son los componentes empresariales que contienen lógica de negocio.

3. La capa de acceso a datos: que contiene clases que interactúan con la base de datos. Estas clases surgen como una necesidad de mantener la cohesión o clases altamente especializadas que ayuden a reducir la dependencia entre las clases y capas. Aquí podemos encontrar también una clase con métodos estáticos que permiten uniformizar las operaciones de acceso a datos a través de un único conjunto de métodos

Las capas mencionadas anteriormente interactúan como se muestra en la siguiente figura 20.

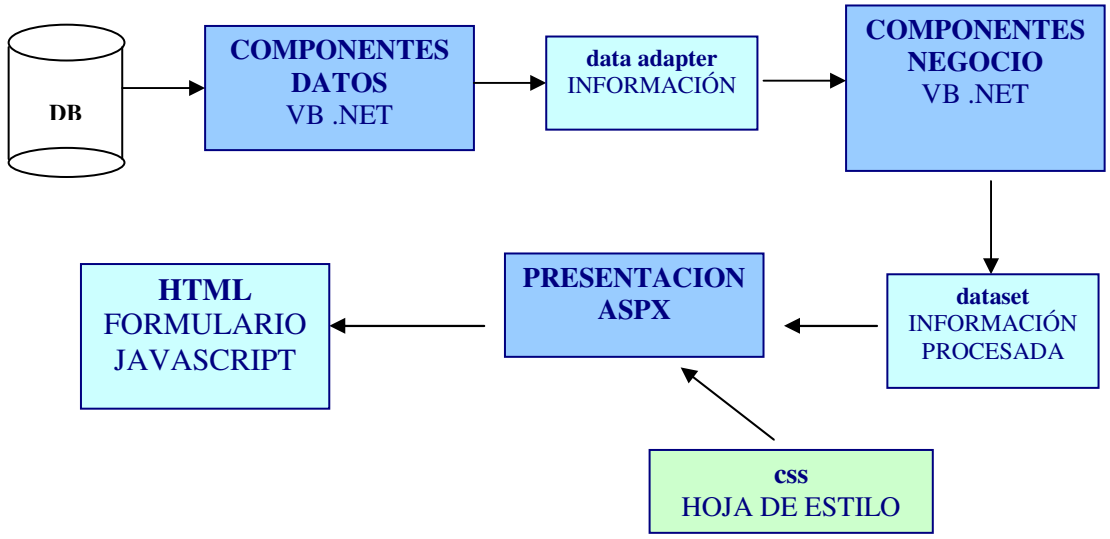


Figura 20. Flujo de la aplicación.

Para la capa de datos estaremos trabajando con lo que es la base de datos, tablas, procedimientos almacenados y componentes de datos. De acuerdo a las herramientas elegidas, aquí entra lo que es ADO.NET, SQL SERVER 2000, componentes de VB.net

Para la capa de Negocio, entra lo que son reglas de negocio, validaciones, cálculos, flujos y procesos. Las herramientas a utilizar son el lenguaje visual Basic.net, componentes locales y comunicación entre componentes.

En la capa de presentación se cuenta con formularios, reportes y respuestas al usuario. Las herramientas a utilizar son ASP.NET, HTML. JavaScript y Crystal Reports.

Un ejemplo se muestra en la figura 21.

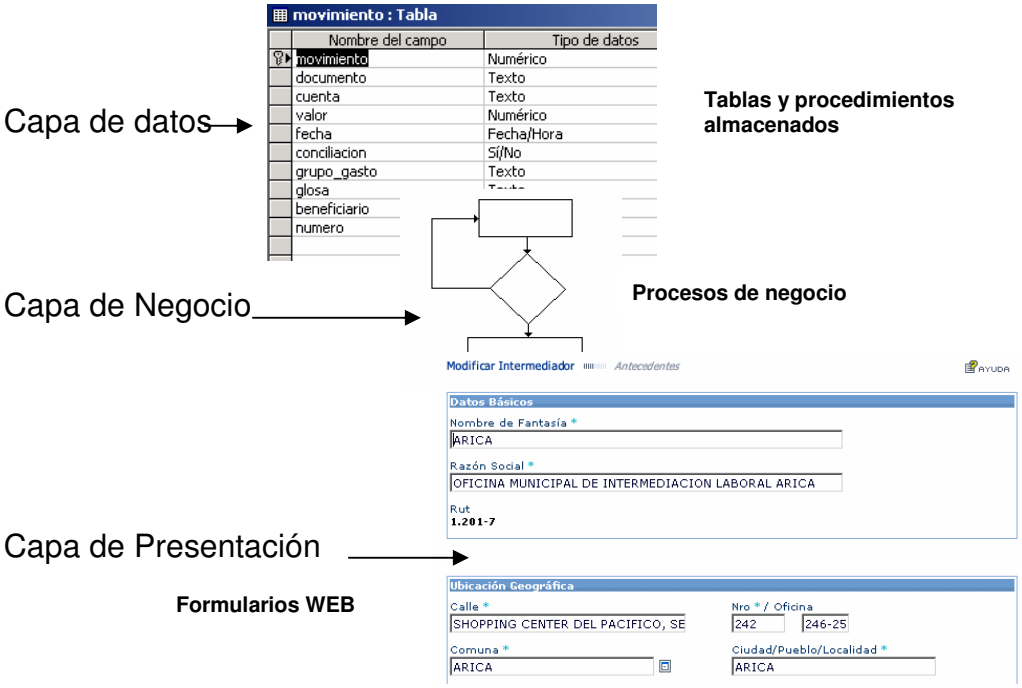


Figura 21. Presentación de la interacción de la arquitectura de 3 capas

CAPÍTULO IV ELABORACIÓN DEL SISTEMA

Después de un análisis completo de las diversas herramientas con las que podemos desarrollar el proyecto y una vez elegidas las más apropiadas, procedemos a la elaboración del sistema.

4.1 ANÁLISIS

Para cumplir con los requerimientos del sistema, así como la solución planteada, se llego a los siguientes Casos de Uso para el desarrollo del sistema.

4.1.1 Casos de Uso

La figura 22, representa la funcionalidad completa del sistema. Mostrando la interacción con los agentes externos, es decir el usuario.

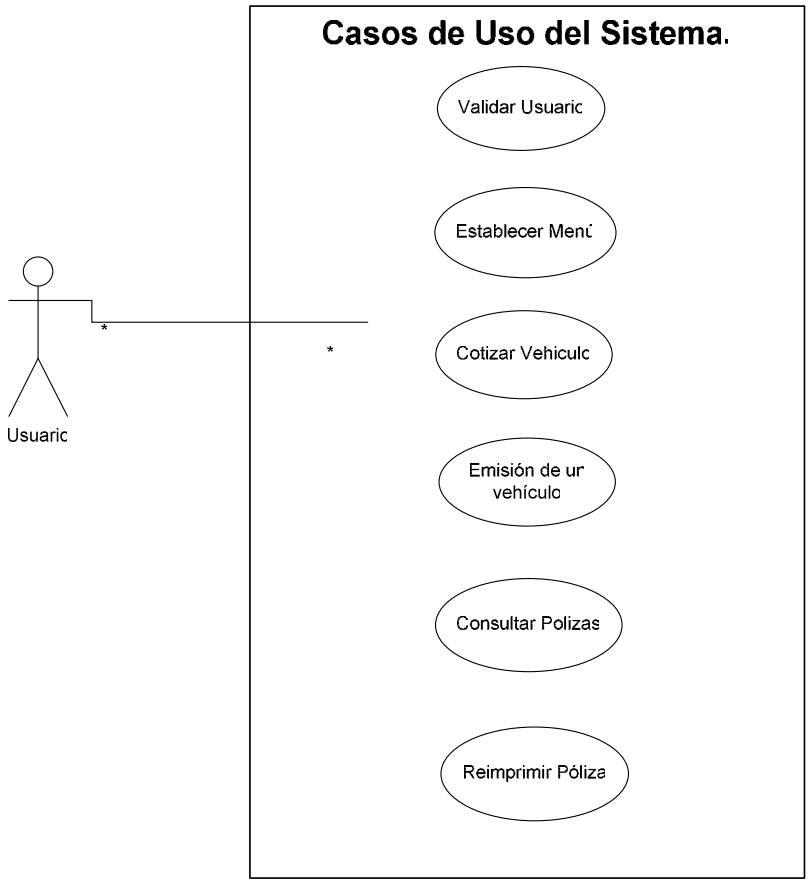


Figura 22. Casos de Uso del Sistema.

Estos casos de uso definen un conjunto de funcionalidades afines que el sistema debe cumplir para satisfacer todos los requerimientos mencionados anteriormente.

4.1.2 Descripción de Casos de Uso.

CU1: Validar Usuario

1. Descripción

Este caso de uso valida que el usuario que desee acceder al sistema sea correcto. El usuario debe proporcionar su clave y contraseña, estos datos se verifican que existan en la base de datos, y posteriormente se checa que tengan permiso para acceder a la aplicación. De estar todo correcto, el usuario accede a la aplicación con su sesión correspondiente, en caso contrario se manda un mensaje de datos incorrectos.

2. Flujo de Eventos

- Se muestra la pantalla de inicio donde se encuentran los campos de usuario y contraseña.
- Capturar información requerida (Usuario y contraseña).
- Se valida la información en la base de datos.
- En caso afirmativo accede al sistema para poder cotizar, emitir, consultar y reimprimir pólizas.
- En caso negativo se manda un mensaje de datos incorrectos y se da la opción de corregir los datos capturados.

3. Precondiciones

- Acceder a Internet a la dirección respectiva de la aplicación.
- Tener a la mano los datos de acceso.
- Por defecto se envía la clave de la aplicación y la clave del programa cuando se valida el usuario.
- El número de intentos para ingresar es ilimitado.

4. Poscondiciones

- Si el usuario es correcto podrá hacer cotizaciones, emisiones, reimprimir y consultar pólizas.
- Se establece una sesión activa de 20 minutos para el usuario que accedió al sistema.

CU2: Establecer Menú.

1. Descripción

Este caso de uso establece el menú que se mostrara de acuerdo al perfil del usuario. Es decir se muestran únicamente los módulos y Submódulos que podrá hacer uso de ellos.

2. Flujo de Eventos

- Cuando se valida el usuario se asigna un perfil de acuerdo a los valores de sesión del usuario, el cual determina que se le va a mostrar.
- Se muestra el menú con sus respectivos módulos y Submódulos.
- En el encabezado muestra el nombre, el área del usuario, así como al programa al cual pertenece.
- Si no tiene asignado ningún perfil, no se le muestra nada, por lo tanto no puede cotizar, emitir, consultar y reimprimir pólizas.

Flujo Alternativo

- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- Haberse validado en el sistema.
- La sesión no haya caducado.

4. Poscondiciones

- Puede cotizar y emitir pólizas.
- Puede reimprimir pólizas.
- Puede consultar pólizas.

CU3: Cotizar Vehículo.

1. Descripción

Este caso de uso muestra las diferentes cotizaciones que un vehiculo puede tener de acuerdo a la información capturada.

2. Flujo de Eventos

- Selecciona opción de cotizar.
- Capturar todos los datos completos.
- Clic para hacer cálculos.
- Validar que la información éste completa.
- Cálculos internos del sistema para determinar primas totales de vehículos.
- Muestra resultados en una tabla con diferentes aseguradoras, paquetes y formas de pago.

Flujos Alternativos

- Clic para limpiar información capturada y volver a cargar datos por defecto.
- Al cambiar las opciones seleccionadas ya sea de una lista o de un campo de texto, limpia la tabla de cotizaciones.
- Al seleccionar una agencia se actualiza la lista de estados y la lista del IVA.
- Al seleccionar el tipo de vehiculo se actualiza la marca, submarca, año, descripción, plazos y el servicio.
- Al seleccionar la marca se actualiza la submarca, año, descripción y plazos.
- Al seleccionar submarca se actualiza el año, descripción y plazos.

- Al seleccionar año se actualiza la descripción y plazos.
- Al seleccionar la descripción se actualizan los plazos.
- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- La sesión no haya caducado.
- Es obligatorio ingresar el valor factura de la unidad.

4. Poscondiciones

- Muestra las diferentes cotizaciones de aseguradoras, paquetes y formas de pago.
- Elegir una cotización para emisión.
- Imprimir la cotización personalizada.
- Mostrar las coberturas de la cotización respectiva en una ventana.

CU4: Emisión de un Vehículo

1. Descripción

Este caso de uso se utiliza para la emisión de una póliza una vez que se capturar los datos del cliente, del vehiculo, y de la póliza para su emisión correspondiente.

2. Flujo de Eventos.

- Elegir una opción de cotización para emisión.
- Capturar todos los datos completos.
- Clic para hacer cálculo de la póliza a emitir.
- Validar que la información éste completa.
- Valida que el número de serie tenga la longitud correcta.
- Valida que las fechas de emisión estén dentro del rango de de fechas establecidas para poder emitir.

- Válida que un vehículo no se emita más de una vez a excepción de que este cancelada o este vencida. Se toma como referencia el número de serie del vehículo.
- Cálculos internos del sistema para determinar prima total del vehículo.
- Graba información en la base de datos.
- Sí todo esta correcto se muestra el número de póliza generado por el sistema y se da la opción de imprimir.
- Si ocurre un error se muestra un mensaje y se queda en la página de emisión.

Flujos Alternativos

- Al seleccionar el tipo de persona física o física con actividad empresarial se muestran los datos del nombre, apellido paterno y materno, nacionalidad, ocupación, estado civil, RFC, Fecha de nacimiento y e-mail.
- Cuando se selecciona persona moral se pide razón social, RFC, fecha de constitución, giro y e-mail.
- Sí se elige otro conductor se pide el nombre completo del conductor.
- Validar que el RFC este correcto, ya sea para persona moral como para persona física.
- Al ingresar el código postal y cambiar el foco a otro campo se busca en la base de datos la colonia, delegación y estado.
- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- Haber elegido una cotización.
- La sesión no haya caducado.
- Capturar todos los datos correctos y necesarios para la emisión.

4. Poscondiciones

- Emisión de la póliza.

- Impresión de la póliza.

CU5: Consultar Póliza.

1. Descripción

Posibilidad de hacer consultas de pólizas emitidas. Se puede consultar de acuerdo a filtros establecidos. Estos filtros son consultar todas las pólizas de una agencia específica, por número de póliza, nombre del asegurado, Número de serie, fecha de inicio de vigencia, fecha de término de vigencia y fecha de emisión. Esta consulta muestra la información más relevante de la póliza

2. Flujo de Eventos

- Selecciona opción de consultas.
- Elegir las opciones disponibles de consulta. Depende del usuario que se haya validado en el sistema.
- Validar que la información éste completa.
- Consulta la base de datos con los filtros establecidos.
- Si no hay pólizas en la búsqueda se manda un mensaje.
- Si se encuentran pólizas muestra resultados de la póliza o pólizas emitidas en un listado con su respectiva paginación.

Flujos Alternativos

- Al seleccionar una opción de consulta muestra los datos de captura para su respectiva consulta.
- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- La sesión no haya caducado.

4. Poscondiciones

- Muestra las pólizas generadas en la búsqueda.
- Posibilidad de imprimir la póliza.
- Paginación en los resultados.

CU6: Reimprimir Póliza.

1. Descripción

Posibilidad de reimprimir una póliza emitida. El sistema busca la póliza tomando en cuanto el usuario que se valido en el sistema. Un usuario sólo puede reimprimir pólizas de las agencias a las cuales esta asignado.

2. Flujo de Eventos.

- Selecciona opción de reimpresión.
- Capturar número de póliza.
- Validar que la información éste completa.
- Consulta la base de datos del sistema.
- Si esta en la base validar que el usuario pueda reimprimirla.
- Mostrar resultado con descripciones breves de la póliza, como es aseguradora y tipo de vehículo.
- Imprimir póliza.
- Si no esta la póliza en la búsqueda se manda un mensaje.

Flujos Alternativos.

- Reimprimir pólizas ilimitadas.
- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- La sesión no haya caducado.

4. Poscondiciones

- Muestra la póliza generada en la búsqueda.
- Imprimir la póliza.

4.1.3 Subcasos de Uso

Estos hacen referencia a la descomposición de los casos de uso del punto anterior. Se dan cuando existe una relación entre dos casos de uso.
Aquí se manejan relaciones de uso, donde el caso requiere que el subcaso se haga completamente para que él mismo se realice bien y completamente.

La figura 23 muestra los subcasos de uso que se manejan en el sistema.

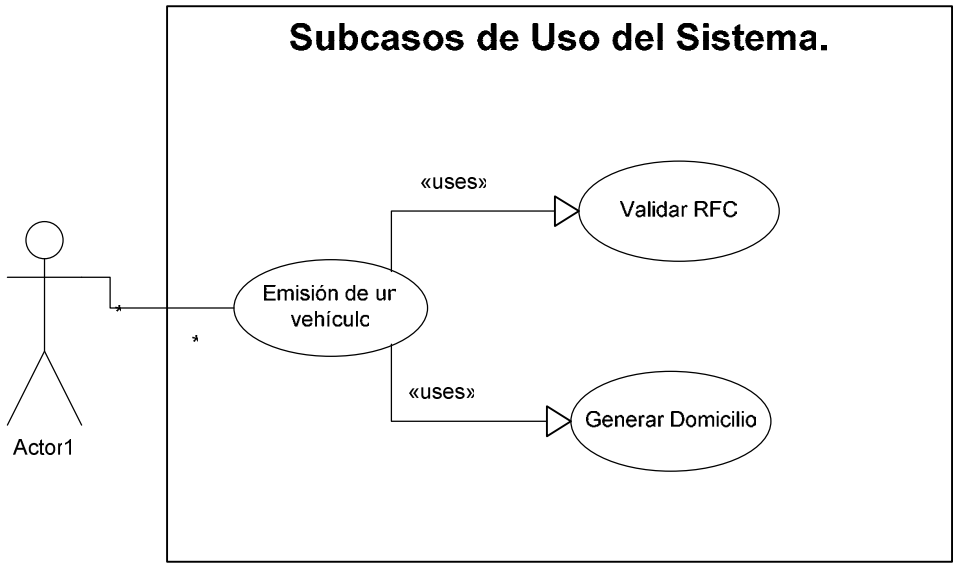


Figura 23. Subcasos de Uso del sistema.

SCU1: Validar RFC.

1. Descripción.

La captura de datos de la emisión requiere del RFC ya sea para una persona física o moral. Este dato es validado y a su vez genera la fecha de nacimiento o de constitución según sea el caso.

2. Flujo de Eventos.

- Selecciona una cotización para emisión.
- Capturar RFC.
- Validar si el RFC ingresado es correcto, tomando en cuenta el tipo de persona que se selecciono.
- Si es correcto genera ya sea la fecha de nacimiento o la fecha de constitución.
- Si no es correcto manda un mensaje con el error detectado.

Flujos Alternativos.

- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- La sesión no haya caducado.
- Ingresar RFC

4. Poscondiciones.

- Genera fecha de nacimiento o fecha de constitución.

5. Referencias

Del CU4. Emisión de póliza.

SCU2: Generar domicilio.

1. Descripción.

La captura de datos en la emisión requiere la dirección del cliente. Para agilizar la captura de la información, una vez que se ingresa el código postal se busca la colonia y el municipio o delegación correspondiente.

2. Flujo de Eventos.

- Selecciona una cotización para emisión.

- Capturar código postal.
- Busca en la base de datos la colonia y municipio correspondiente.
- En caso de que no se encuentre nada, se da la opción de captura libre para estos datos.

Flujos Alternativos

- Si da clic en salir se elimina la sesión del usuario y por lo tanto tiene que volver a firmarse.

3. Precondiciones

- La sesión no haya caducado.
- Ingresar Código Postal

4. Poscondiciones.

- Muestra la colonia y municipio correspondiente.

5. Referencias
Del CU4. Emisión de póliza.

4.1.4 Estructura del Proyecto

Basándonos en el modelo de servicios así como la arquitectura de 3 capas que se va a utilizar, la arquitectura de la figura 24 se adapta a las necesidades de los requerimientos. Esta tendría las siguientes capas:

- Web
- BusinnesFacade
- BusinnesRules
- DataAccess
- SystemFramework
- Common

El conjunto de estas capas forma parte de la arquitectura Duwamish[1].

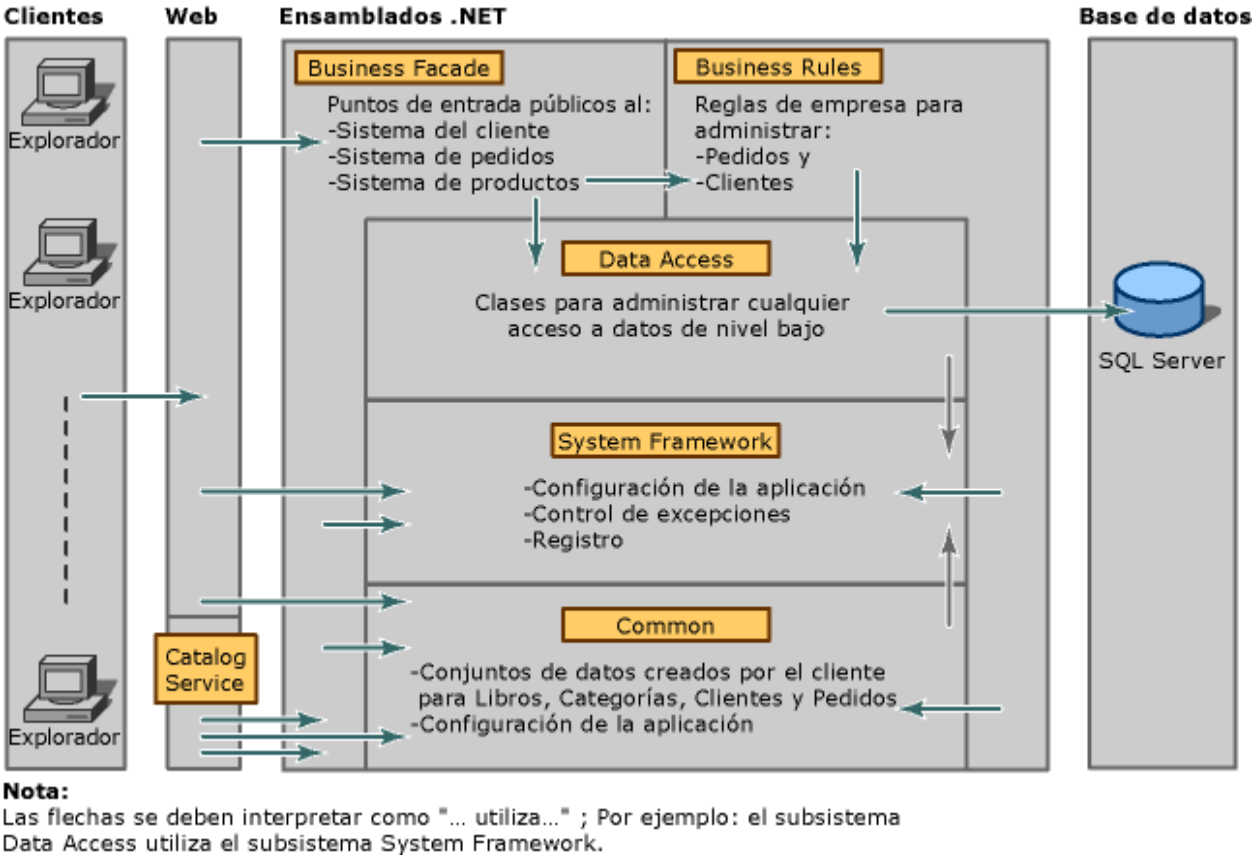


Figura 24. Arquitectura Duwamish.

- La capa Web permite que los clientes tengan acceso a la aplicación.
- La capa Business Facade es la interface con la capa Web, es una capa de aislamiento, ya que separa la interfaz del usuario de la implementación de las distintas funciones empresariales.
- La capa Business Rules contiene la implementación de las reglas y la lógica empresarial.
- La capa Data Access proporciona servicios de datos.
- La capa Common contiene conjunto de datos que se utilizan para pasar información entre las distintas capas.

[1] En esta ubicación obtendrá mayor información de la arquitectura Duwamish. http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/dwamish7/html/vtgrfArchitecturalOverviewOfDuwamish70.asp

- La capa SystemFramework contiene clases para la configuración de la aplicación.

Con esta arquitectura se administran mejor las capas, pero debido a que todas las aplicaciones Web van a depender de las capas, cada que haya un cambio en alguna de ellas se tiene el riesgo de afectar a otra aplicación. Por tal motivo se pensó en integrar el proyecto en aplicaciones Web independientes. Este punto es para atender el requerimiento de que otra aplicación Web de cotización entre en este esquema, sin afectar o perjudicar a las restantes ya que en .NET se compila todo el proyecto.

Por lo tanto, pensando en que las aplicaciones deben ser independientes, se llegó a la siguiente arquitectura de la figura 25.

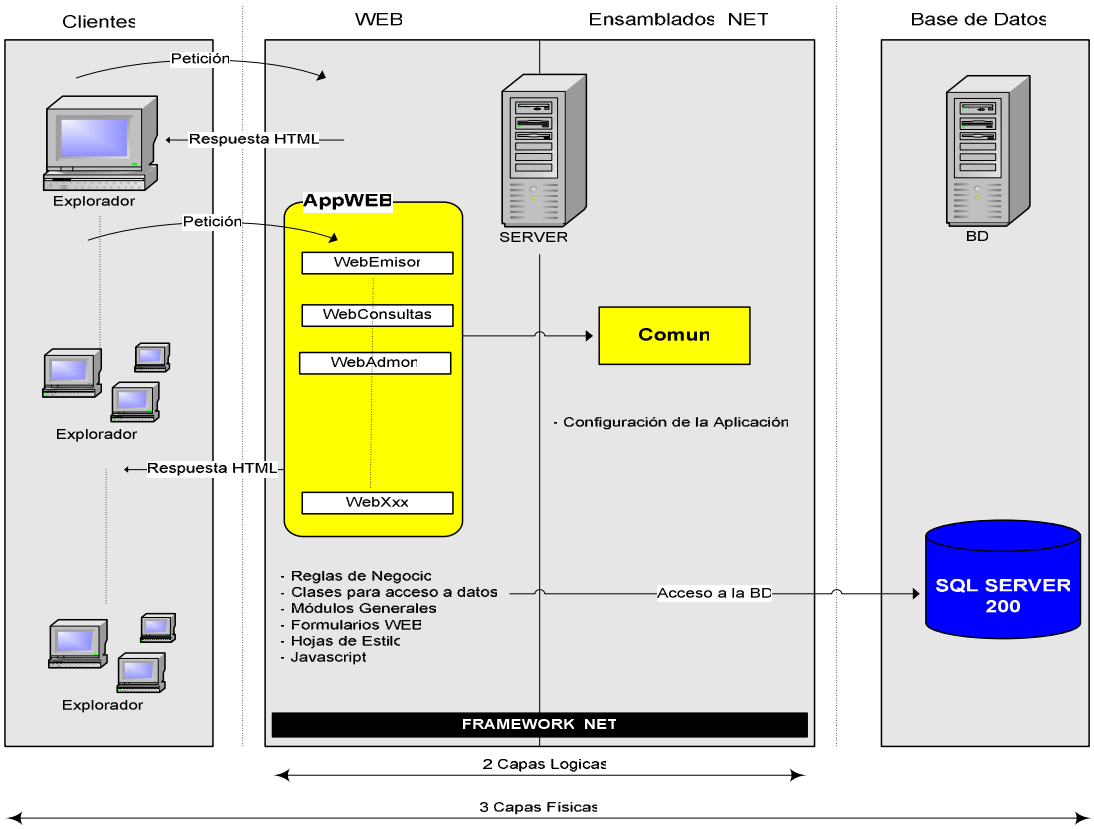


Figura 25. Arquitectura del Emisor.

La arquitectura esta compuesta de 3 capas físicas y dos capas lógicas.
Las capas físicas consisten en lo siguiente:

- La PC de cada cliente es el medio por el cual el cliente envía la petición a la aplicación Web y obtiene una respuesta en HTML que se ve reflejada en el explorador del cliente.
- Servidor de Aplicaciones Web y servicios.
- Servidor de Base de Datos.

Las capas lógicas contienen por un lado la aplicación Web, lo que es la interfaz gráfica, así como las reglas de negocio. Por otro lado la configuración de la aplicación donde se establece el nombre del alias para la dirección de la aplicación y la cadena de conexión.

4.2 DISEÑO.

En esta parte se explica la forma en que va estar integrado el sistema en lo que se refiere a la base de datos, así como la parte de cómo se elaboraran los formularios para la interfaz gráfica.

4.2.1 Diagrama de clases.

El diagrama de la figura 26, nos muestra una vista de la aplicación de un modo estático. Las clases son la plantilla de los objetos, y aquí podemos ver representados a estos con sus características y su comportamiento, así como la relación entre ellas.

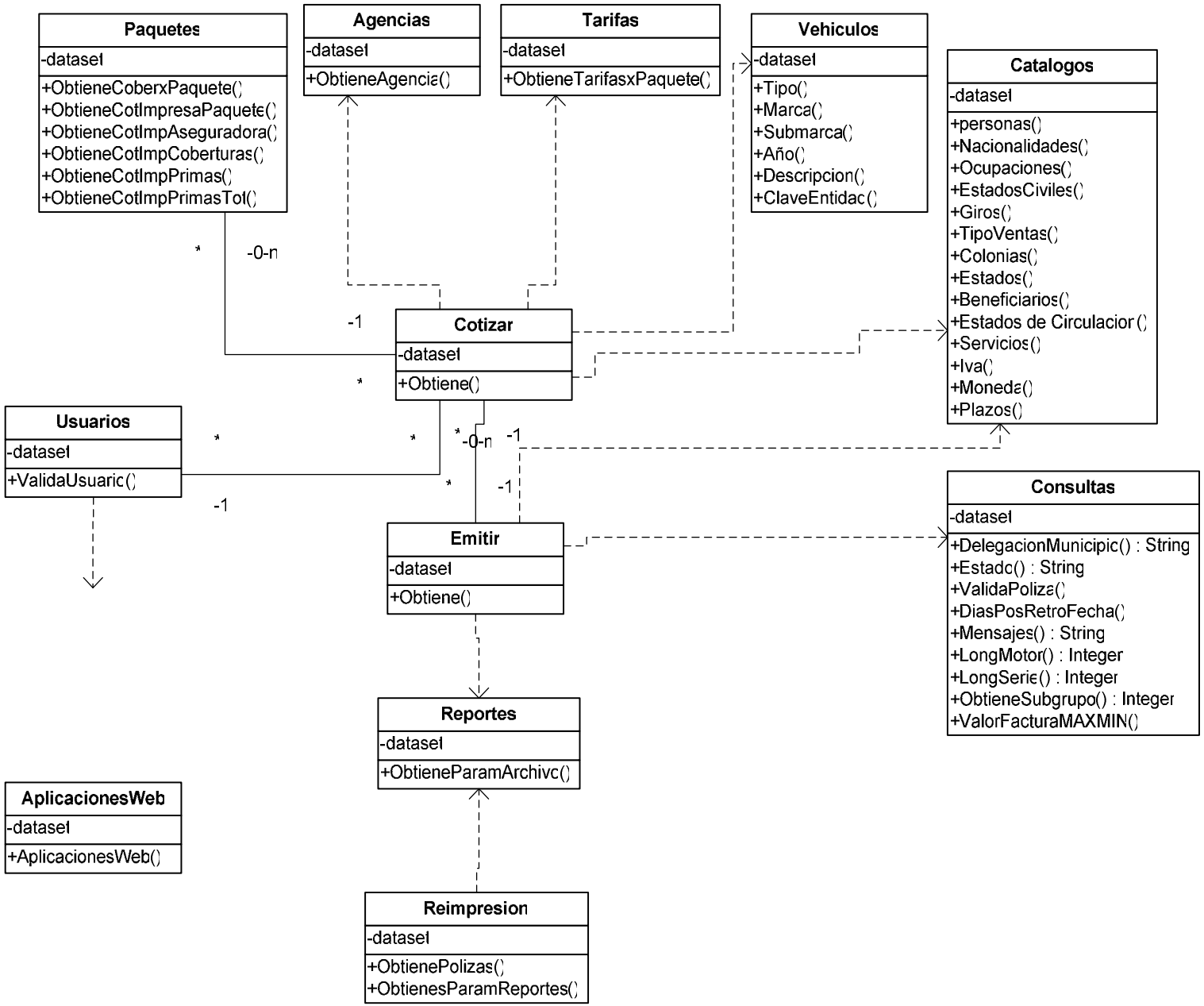


Figura 26 Diagrama de clases

4.2.2 Diagrama Entidad-Relación del sistema.

Ver anexo I

Para entender que es lo que contiene el modelo de la base de datos a continuación se detalla el diccionario de datos con las partes más importantes.

4.2.3 Diccionario de Datos.

Para un mejor entendimiento el tipo de datos maneja las siguientes abreviaturas.

I = Integer
VA = VARCHAR, el número representa la longitud.
BT = Bigint
D = DateTime
F = Flota
TXT = Text

Entidad Catálogo de Administradores del Sistema

Descripción:	Catálogo de Administradores del Sistema
Entidad:	CM_A_ADMONSISTEM

Atributos

Descripción	Campo	Tipo	NULO
Identificador del Administrador	ID_ADMON	I	Si

Entidad Catálogo de agentes

Descripción:	Catálogo de agentes
Entidad:	CM_C_AGENTES

Atributos

Descripción	Campo	Tipo	NULO
Clave del agente asignada por la aseguradora	CVE_AGENTE	I	Si
Descripción del agente	DESC_AGENTE	VA100	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de áreas de BOA

Descripción:	Catálogo de áreas de BOA
Entidad:	CM_C_AREAS_BOA

Atributos

Descripción	Campo	Tipo	NULO
Clave del área de BOA	CVE_AREA	I	Si
Descripción del área de BOA	DESC_AREA	VA100	Si
Numero de Fax 1	NUM_FAX1	VA15	No
Numero de Fax2	NUM_FAX2	VA15	No
Numero de Fax3	NUM_FAX3	VA15	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Aseguradoras

Descripción: Catálogo de Aseguradoras
Entidad: CM_C_ASEGURADORAS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la aseguradora	CVE_ASE	I	Si
Nombre de la aseguradora	RAZON_SOCIAL	VA100	Si
Nombre corto de la aseguradora	DESC_CORTO_ASE	VA10	Si
Periodo de Cobro de la Póliza	PERIODO_COBRO	I	Si
Registro Federal de Contribuyentes	RFC	VA13	Si
Dirección fiscal de la aseguradora	DIR_FISCAL_ASE	VA255	No
Teléfono de oficina central	TEL_OFNA_ASE	VA20	Si
Teléfono de Fax	TEL_FAX_ASE	VA20	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de clientes

Descripción: Catálogo de clientes
Entidad: CM_C_CLIENTES

Atributos

Descripción	Campo	Tipo	NULO
Clave del cliente	CVE_CLIENTE	I	Si
Clave del Usuario que da la Alta	CVE_USUARIO_ALTA	I	Si
Nombre del cliente	NOMBRE_CLIENTE	VA100	No
Apellido paterno del cliente	APE_PAT_CLIENTE	VA50	No
Apellido materno del cliente	APE_MAT_CLIENTE	VA50	No
Razón social persona moral	RAZON_SOCIAL_CLI	VA150	No
Fecha de Constitución	FEC_CONSTITUCION	D	No
Registro Federal de Contribuyentes	RFC_CLIENTE	VA20	Si
Fecha de nacimiento del cliente	FEC_NAC_CLI	D	No
Sexo del cliente	SEXO_CLI	VA1	No
Correo del cliente	EMAIL_CLI	VA50	No
Numero de Celular	NUM_CELULAR	VA20	No
Fecha de alta del cliente	FEC_ALTA_CLI	D	Si
Cliente VIP	CLIENTE_VIP	BT1	No
Edad del beneficiario	EDAD	I	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Coberturas

Descripción: Catálogo de Coberturas
Entidad: CM_C_COBERTURAS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la cobertura	CVE_COBERTURA	I	Si
Descripción de la cobertura	DESC_COBERTURA	VA100	Si
Porcentaje de comisión	POR_COM_COBERTURA	F	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Colonias

Descripción: Catálogo de Colonias
Entidad: CM_C_COLONIAS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la Colonia	CVE_COLONIA	VA5	Si
Descripción de la Colonia	DESC_COLONIA	VA150	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Delegaciones y Municipios

Descripción: Catálogo de Delegaciones y Municipios
Entidad: CM_C_DELEGMUNICIPIOS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la Delegación o Municipio	CVE_DELGMUNICIPIO	VA3	Si
Descripción del Municipio o Delegación	DESC_DELGMUNICIPIO	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de direcciones del cliente

Descripción: Catálogo de direcciones del cliente
Entidad: CM_C_DIRECCIONES_CLI

Atributos

Descripción	Campo	Tipo	NULO
Clave dirección	CVE_DIRECCION	I	Si
Clave del Código Postal	CVE_CP	VA5	Si
Departamento	DIR_DEPTO	VA100	No
Plaza	PLAZA	VA50	No
Edificio	EDIFICIO	VA100	No
Piso	PISO	VA5	No
Dirección	DIRECCION	VA200	Si
Colonia	COLONIA	VA150	No
Delegación	DELEGACION	VA50	No
Estado	ESTADO	VA30	No
Clave Lada	LADA	VA4	No
Teléfono de la dirección	TELEFONO	VA15	Si
Extensión	EXTENSION	VA6	No
Teléfono de Fax	FAX	VA15	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Catálogo de Estados

Descripción: Catálogo de Estados
Entidad: CM_C_ESTADOS

Atributos

Descripción	Campo	Tipo	NULO
Clave del Estado	CVE_EDOSEPOMEX	VA2	Si
Descripción del Estado	DESC_ESTADO	VA30	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de formas de pago

Descripción: Catálogo de formas de pago
Entidad: CM_C_FORMAS_PAGO

Atributos

Descripción	Campo	Tipo	NULO
Clave de la forma de pago	CVE_FORMA_PAGO	I	Si
Descripción de la forma de pago	DESC_FORMA_PAGO	VA30	Si
Periodo de la Forma de Pago	MESES_FORMA_PAGO	F	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Formulas por Tarifa

Descripción:	Catálogo de Formulas por Tarifa
Entidad:	CM_A_FORMULAS

Atributos

Descripción	Campo	Tipo	NULO
Descripción de la Fórmula	DESC_FORMULA	VA255	Si
Clave de la Cobertura Dependiente	CVE_COBERTURA_DEP	I	Si
Entidad Asignada para Calculo	ENTIDAD_CALCULO	VA50	No
Observaciones de la Fórmula	OBSERV_FORMULA	VA255	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Catálogo de grupos de clientes

Descripción:	Catálogo de grupos de clientes
Entidad:	CM_C_GRUPOS

Atributos

Descripción	Campo	Tipo	NULO
Clave del grupo	CVE_GRUPO	I	Si
Descripción del grupo	DESC_GRUPO	VA255	Si
Mostrar para Facturación de UDI	MOSTRAR_EN_FACTURACION	BT	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de IVA por Tarifa

Descripción:	Catálogo de IVA por Tarifa
Entidad:	CM_A_IVATARIFA

Atributos

Descripción	Campo	Tipo	NULO
Bandera de activación	ACTIVO	BT1	Si

Entidad Catalogo de Marcas

Descripción:	Catálogo de Marcas
Entidad:	CM_C_MARCAS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la Marca	CVE_MARCA	I	Si
Descripción de la Marca	DESC_MARCA	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Módulos del Sistema

Descripción:	Catálogo de Módulos del Sistema
Entidad:	CM_C_MODULOS_SISTEMA

Atributos

Descripción	Campo	Tipo	NULO
Clave del Módulo del sistema	CVE_MODULO	I	Si
Descripción del Módulo	DESC_MODULO	VA50	Si
Ancho del Menú	ANCHO_MENU	I	Si
Alto del Menú	ALTO_MENU	I	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Perfiles por Usuario

Descripción:	Catálogo de Perfiles por Usuario
Entidad:	CM_C_PERFILES_USUARIO

Atributos

Descripción	Campo	Tipo	NULO
Clave del Perfil del Usuario	CVE_PERFIL	I	Si
Descripción del Perfil	DESC_PERFIL	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Periodos de pólizas

Descripción:	Catálogo de Periodos de pólizas
Entidad:	CM_C_PERIODOS_POLIZAS

Atributos

Descripción	Campo	Tipo	NULO
Número de meses comprendidos en el periodo	CVE_PERIODO_POL	I	Si
Descripción del tipo de periodo	DESC_PERIODO_POL	VA30	Si
Días por periodo	NUM_DIAS_PERIODO	I	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Plazos por Factores de Forma de Pago

Descripción:	Catálogo de Plazos por Factores de Forma de Pago
Entidad:	CM_C_PLAZOFACTFRMPAGO

Atributos

Descripción	Campo	Tipo	NULO
Clave de Plazo por Factor de Forma de Pago	CVE_PLAZOFACTFRMPAGO	I	Si
Factor de Plazo	FACT_PLAZO	F	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Puntos de Venta

Descripción:	Catálogo de Puntos de Venta
Entidad:	CM_C_PUNTOSVTA

Atributos

Descripción	Campo	Tipo	NULO
Clave Punto de Venta	CVE_PTO_VTA	I	Si
Clave Agencia Anterior	CVE_AGENCIA_ANT	VA10	No
Clave en la Aseguradora	CVE_AGENCIA_ASE	VA5	No
Agencia Fronteriza	FRONTERIZA	BT	No
Nombre Comercial del Punto de Venta	NOMBRE_COMERCIAL	VA200	No
Nemonico	NEMONICO	VA50	Si
IVA para Facturar Comisiones	IVA_FACTURA_UDI	F	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Ramos

Descripción:	Catálogo de Ramos
Entidad:	CM_C_RAMOS

Atributos

Descripción	Campo	Tipo	NULO
Clave del ramo	CVE_RAMO	I	Si
Descripción del ramo	DESC_RAMO	VA50	Si
Descripción para estado de cuenta	DESC_EDO_CTA	VA50	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Rangos de Pólizas

Descripción:	Catálogo de Rangos de Pólizas
Entidad:	CM_A_RANGOS_POLIZAS

Atributos

Descripción	Campo	Tipo	NULO
Serie del Rango	SERIE_RANGO	VA25	Si
Observaciones	OBSERV_RANGO	VA255	No
Fecha de Alta del Rango	FEC_ALTA	D	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Catálogo de Recargos por Producto

Descripción:	Catálogo de Recargos por Producto
Entidad:	CM_C_RECARGOSPROD

Atributos

Descripción	Campo	Tipo	NULO
Clave del Factor Recargo	CVE_RECARGO	I	Si
Factor del Recargo	FACT_FRMPAGO	F	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Servicios para Autos

Descripción:	Catálogo de Servicios para Autos
Entidad:	CM_C_SERVICIOS

Atributos

Descripción	Campo	Tipo	NULO
Clave del tipo de uso	CVE_SERVICIOS	I	Si
Descripción del tipo de servicio	DESC_SERVICIO	VA25	Si

Entidad Catálogo de Submarcas

Descripción:	Catálogo de Submarcas
Entidad:	CM_C_CATSUBMARCAS

Atributos

Descripción	Campo	Tipo	NULO
Clave de la Submarca	CVE_SUBMARCA	I	Si
Descripción de la Submarca	DESC_SUBMARCA	VA50	Si

Entidad Catálogo de Submódulos para el Sistema

Descripción:	Catálogo de Submódulos para el Sistema
Entidad:	CM_C_SUBMODULOS_SISTEMA

Atributos

Descripción	Campo	Tipo	NULO
Clave del Submódulo	CVE_SUBMODULO	I	Si
Descripción del Modulo	DESC_SUBMODULO	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Subramos

Descripción:	Catálogo de Subramos
Entidad:	CM_C_SUBRAMOS

Atributos

Descripción	Campo	Tipo	NULO
Clave de subramo	CVE_SUBRAMO	I	Si
Descripción de subramo	DESC_SUBRAMO	VA50	Si
Sufijo del Subramo	SUFIJO	VA2	Si
Porcentaje de Comisión	PORCENT_COMISION	F	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de tipos de moneda

Descripción:	Catálogo de tipos de moneda
Entidad:	CM_C_TIPO_MONEDA

Atributos

Descripción	Campo	Tipo	NULO
Clave del tipo de moneda	CVE_TIPO_MONEDA	I	Si
Descripción del tipo de moneda	DESC_TIPO_MONEDA	VA20	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de tipos de venta

Descripción:	Catálogo de tipos de venta
Entidad:	CM_C_TIPO_VENTA

Atributos

Descripción	Campo	Tipo	NULO
Clave de tipo de venta	CVE_TIPO_VENTA	I	Si
Descripción de tipo de venta	DESC_TIPO_VENTA	VA20	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de usuarios

Descripción:	Catálogo de usuarios
Entidad:	CM_C_USUARIOS

Atributos

Descripción	Campo	Tipo	NULO
Clave del Usuario	CVE_USUARIO	I	Si
Usuario del Sistema Login	LOGIN	VA50	Si
Password del usuario	PASS_USUA	VA20	Si
Nombre del usuario	NOMBRE_USR	VA50	Si
Apellido Paterno del Usuario	APATERNO	VA50	Si
Apellido Materno del Usuario	AMATERNO	VA50	No
Bandera de activación	ACTIVO	BT1	Si

Entidad Catálogo de Vehículos

Descripción:	Catálogo de Vehículos
Entidad:	CM_C_VEHICULOS

Atributos

Descripción	Campo	Tipo	NULO
Clave Asignada por BOA	CVE_BOA	VA30	Si
Modelo del Vehículo	MODELO	I	Si
Descripción	DESCRIPCIÓN	VA150	Si
Fecha Alta Vehículo	FEC_CARGA_VEHI	D	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Catálogo de Códigos Postales

Descripción:	Catálogo de Códigos Postales
Entidad:	CM_CODIGOS_POSTALES

Atributos

Descripción	Campo	Tipo	NULO
Clave del Código Postal	CVE_CP	VA5	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Cuotas y Constantes

Descripción:	Cuotas y Constantes
Entidad:	CM_A_CUOTAS_CONSTANTES

Atributos

Descripción	Campo	Tipo	NULO
Precio de la Unidad	PRECIO_VEHICULO	F	No
Número de Ocupantes	NUM_OCUPANTES	I	No
Cuota Neta	CUOTA_NETA	F	Si
Constante Neta	CONSTANTE_NETA	F	Si
Tonelaje	TONELAJE	F	No
Límite mínimo valor factura	LIMITE_MINVALORFACT	F	No
Límite máximo valor factura	LIMITE_MAXVALORFACT	F	No
Límite mínimo del equipo especial	LIMITE_MINEQESPECIAL	F	No
Límite máximo del equipo especial	LIMITE_MAXEQESPECIAL	F	No
Fecha de Alta de la Cuota	FEC_ALTACUOTA	D	Si
Fecha de Baja de la Cuota	FEC_BAJACUOTA	D	No
Porcentaje Máximo Valor Factura	PORCEN_MAXVALORFACT	F	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Definición de Retro Actividad y Posfechas

Descripción:	Definición de Retro Actividad y Posfechas
Entidad:	CM_A_DEFPOSFEC_RETRO

Atributos

Descripción	Campo	Tipo	NULO
Bandera de activación	ACTIVO	BT1	Si

Entidad Definición de SubMódulos por Grupo y Módulo

Descripción:	Definición de SubMódulos por Grupo y Módulo
Entidad:	CM_A_DETSUBMODULOS

Atributos

Descripción	Campo	Tipo	NULO
Dirección Web	URL	VA150	Si
Orden de los Módulos	ORDEN_MODULO	I	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Descripción del Puesto del Usuario

Descripción:	Descripción del Puesto del Usuario
Entidad:	CM_C_DESC_PUESTO

Atributos

Descripción	Campo	Tipo	NULO
Clave del origen del usuario	CVE_PUESTO	I	Si
Descripción del origen del usuario	DESC_PUESTO	VA100	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Detalle de paquetes de coberturas

Descripción:	Detalle de paquetes de coberturas
Entidad:	CM_A_DET_PAQ_COB

Atributos

Descripción	Campo	Tipo	NULO
Identificador Detalle Paquete Coberturas	ID_DETPAQCOB	I	Si
Adición de signo de pesos	SIGNO_PESOS	VA1	No
Suma asegurada de la Cobertura	SUMA_ASEGURADA	VA25	Si
Prima aplicada a la cobertura	PRIMA_APLICADA	F	Si
Deducible aplicable a la Cobertura	DEDUCIBLE	VA25	No
Coaseguro aplicable a la Póliza	COASEGURO	VA25	No
Cobertura Adicional	ADICIONAL	BT1	No
Comisión de cobertura	COMISION	F	No
Clave Cobertura según Libro de Condiciones	CVE_COBASE_CONDICIONES	VA5	No
Orden de Impresión de las Coberturas	ORDEN_IMPRESION	I	Si
Imprimir Cobertura	IMP_COBERTURA	BT1	Si

Entidad Detalles de Puntos de Venta

Descripción:	Detalles de Puntos de Venta
Entidad:	CM_A_DET_PTOVTA

Atributos

Descripción	Campo	Tipo	NULO
Clave Agencia	ID_PTOVTA	I	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Paquetes de Pólizas

Descripción:	Paquetes de Pólizas
Entidad:	CM_C_PAQUETES

Atributos

Descripción	Campo	Tipo	NULO
Clave del paquete	CVE_PAQUETE	I	Si
Descripción del paquete	DESC_PAQUETE	VA100	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Paquetes Predefinidos

Descripción:	Paquetes Predefinidos
Entidad:	CM_C_PAQ_PREDEFINIDOS

Atributos

Descripción	Campo	Tipo	NULO
Identificador del Paquete Definido	ID_PAQDEFINIDO	I	Si
Descripción de la Cobertura Homologada	DESC_COBHOMOLOGADA	VA100	No
Adición de signo de pesos	SIGNO_PESOS	VA1	Si
Suma Asegurada de la Cobertura	SUM_ASEGURADA	VA50	Si
Clave de suma asegurada según sistema en la aseguradora	CVE_SUMA_ASEGURADA_ASE	VA5	No
Complemento de Suma Asegurada	COM_SUM_ASEGURADA	VA50	No
Clave Cobertura de la Aseguradora	CVE_COBASE	VA5	No
Clave Cobertura según Libro de Condiciones	CVE_COBASE_CONDICIONES	VA5	Si
Cobertura Amparada	AMPARADA	BT1	Si
Cobertura Adicional	ADICIONAL	BT1	Si
Factor de Cálculo de la Cobertura Adicional	FACTOR_COBERTURA	F	Si
Deducible aplicable a la Cobertura	DEDUCIBLE	VA25	Si
Orden de Coberturas para Impresión	ORDEN_IMP	I	No
Imprimir Cobertura	IMP_COBERTURA	BT1	Si
Mostrar Cobertura para Cotización	MOSTRAR_COBERTURA	BT	Si
Complemento de Suma Asegurada en Cotizaciones	COMP_SUMASE_COTIZA	VA50	No
Descripción de Condiciones por Cobertura	DESC_COND_COBERTURA	VA300	No
Etiqueta de Descripción de las condiciones por cobertura	ETIQUETA_COND_COBERTURA	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Pólizas Comunes

Descripción:	Pólizas Comunes
Entidad:	CM_A_POLIZAS

Atributos

Descripción	Campo	Tipo	NULO
Número de póliza de la aseguradora	NUM_POLIZA_ASE	VA25	Si
Fecha emisión de póliza	FEC_EMISION	D	Si
Fecha de inicio de vigencia	FEC_INICIO_VIGENCIA	D	Si
Fecha de termino de vigencia	FEC_FIN_VIGENCIA	D	Si
Fecha captura	FEC_CAPTURA	D	Si
Número de carátula de la póliza	NUM_CARATULA	VA20	No
Contrato de la póliza	NUM_CONTRATO	VA20	No
Prima neta de la póliza	PRIMA_NETA_POL	F	Si
Gastos de expedición de la póliza	GASTOS_EXPEDICION	F	No
Subtotal de la Póliza	SUBTOTAL_POLIZA	F	No
Bonificación de la póliza	BONIF_POL	F	No
Recargos de la póliza	RECARGOS_POL	F	No
IVA de la póliza	IVA_POL	F	Si
Porcentaje del IVA	PORCENT_IVA	F	Si
Prima total de la póliza	PRIMA_TOTAL_POL	F	Si
Porcentaje de Comisión	PORCENT_COMISION	F	No
Importe de la comisión de la póliza	IMPORTE_COMISION	F	No
Porcentaje de sobre comisión	PORCENT_SOBRECOM	F	No
Porcentaje de Recargo	PORCENT_RECARGO	F	No
Importe de la sobre comisión	IMP_SOBRECOM	F	No
Numero de Factura en International	NUM_FACT_INTER	VA25	No
Tipo de cambio	TIPO_CAMBIO	F	No
Número Cliente en la Aseguradora	NUM_CLIENTE_ASE	VA25	No
Leyenda de la Pólizas	LEYENDA_PÓLIZA	VA255	No
Leyenda en Gota de Agua	LEYENDA_GOTAAGUA	VA255	No
Fecha del Estatus de la Póliza	FEC_STATUSPOL	D	No
Fecha de cancelación del recibo	FEC_CANCELACION	D	No
Número de endoso de la aseguradora	NUM_ENDOSO_ASE	VA25	No
Observaciones de la póliza	OBSERVACIONES_POL	TXT3000	No

Entidad Recibos de las Pólizas

Descripción:	Recibos de las Pólizas
Entidad:	CM_A_RECIBOS

Atributos

Descripción	Campo	Tipo	NULO
Número de recibo de la aseguradora	NUM_RECIBO_ASE	VA25	Si
Serie de Recibos	SERIE_RECIBO	I	Si
Total de Recibos	TOTAL_RECIBOS	I	Si
Fecha de inicio vigencia	FEC_INICIO_RECIBO	D	Si
Fecha de termino de vigencia	FEC_FIN_RECIBO	D	Si
Prima Total del recibo	PRIMA_TOTAL_REC	F	Si
Prima neta del recibo	PRIMA_NETA_REC	F	Si
Recargos del recibo	RECARGOS_RECIBO	F	No
Gastos de Expedición	GASTOS_RECIBO	F	No
IVA del recibo	IVA_RECIBO	F	Si
Bonificación del recibo	BONIFICACION_RECIBO	F	No
Fecha de cancelación del recibo	FEC_CANCELACION	D	No
Folio de Prepago	FOLIO_PREPAGO	I	No
Fecha última Modificación	FEC_ULTIMOD	D	Si
Días de Reactivación	DIAS_REACTIVACIÓN	I	Si
Rehabilitación	REHABILITADO	BT1	No
Fecha de Rehabilitación	FEC_REHABILITACION	D	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Registro de incisos para automóviles

Descripción:	Registro de incisos para automóviles
Entidad:	CM_A_INCISOS

Atributos

Descripción	Campo	Tipo	NULO
ID del Inciso de Auto	ID_INCAUTO	I	Si
Número de Inciso Asignado por la Aseguradora	NUM_INCISO_AUTO	VA25	No
Nombre del Contratante	NOMBRE_CONTRATANTE	VA500	No
Nombre del conductor	NOMBRE_CONDUCTOR	VA200	No
Apellido Paterno del Conductor	APELLIDO_PATCONDUCTOR	VA50	No
Apellido Materno del Conductor	APELLIDO_MATCONDUCTOR	VA50	No
Nombre del Titular ó Asegurado	NOMBRE_TITULAR	VA50	No
Apellido Paterno del Titular	APELLIDO_PATITULAR	VA50	No
Apellido Materno del Titular	APELLIDO_MATITULAR	VA50	No
Teléfono del Titular	TEL_TITULAR	VA50	No
Número de Cliente en la Aseguradora	NUM_CLIASE	VA25	No
Número de Oficina Referencia de la Aseguradora	OFICINA_ASE	VA10	No
Número de Zona Asignada por la Aseguradora	ZONA_ASE	VA5	No
Número de Folio asignado por la Aseguradora	NUM_FOLIOASE	VA30	No
Leyenda del Endoso	LEYENDA_ENDOSOS	VA255	No
Clave del auto asignada por la aseguradora AMISNASA	CVE_AUTO_ASE	VA10	Si
Descripción de la marca	DESC_MARCA_ASE	VA80	Si
Descripción de la Submarca	DESC_SUBMARCA_ASE	VA50	No
Descripción del auto	DESC_AUTO_ASE	VA150	Si
Modelo del auto	MODELO_AUTO	I	Si
Número de serie	NUM_SERIE	VA25	Si
Número de motor	NUM_MOTOR	VA26	Si
Número Económico	NUM_ECONOMICO	VA20	No
Placas del auto	PLACAS_AUTO	VA10	No
Adaptaciones del auto	ADAPTACIONES	VA255	No
Importe de las adaptaciones del auto	IMPORTE_ADAP	F	No
Tipo de cambio para el importe de la adaptación	TIPO_CAMBIO_ADAP	F	No
Fecha del Tipo de Cambio de la Adaptación	FEC_TIPOCAMBIO_ADAP	D	No
Tipo de Moneda de la Adaptación	CVE_MONEDA_ADAP	N16,4	No
Equipo especial	EQUIPO_ESPECIAL	VA255	No
Importe del equipo especial	IMPORTE_EQ_ESP	F	No
Clave del estado donde circula el auto	REF_EDO_CIRCULA	VA5	No
Fecha de inicio del contrato de financiamiento	FEC_INI_FINAN	D	No
Fecha fin del contrato de financiamiento	FEC_FIN_FINAN	D	No
Folio Contrato Peugeot	FOLIO_PEUGEOT	VA10	No
Producto Peugeot	PRODUCTO_PEUGEOT	VA10	No
Grupo SuAuto	GRUPO_SUAUTO	VA10	No
Integrante su Auto	INTEGRANTE_SUAUTO	VA10	No
Digito Verificador	DV_SUAUTO	VA10	No
Contrato de Financiamiento al cliente	CONTRATO_FINAN	VA20	No

Descripción	Campo	Tipo	NULO
Tipo de cambio del financiamiento	TIPOCAMBIO_FINAN	F	No
Tipo de Moneda del Financiamiento	CVE_MONEDA_FINAN	VA20	No
Fecha del Tipo de Cambio del Financiamiento	FEC_TIPOCAMBIO_FINAN	D	No
Descripción del beneficiario del preferente	DESC_BENEF_PREF	VA255	No
Color del Vehículo	COLOR_VEHICULO	VA25	No
Observaciones	OBSER_AUTO	VA255	No
Valor del Vehículo	VAL_VEHICULO	F	Si
Tipo Carga del Vehículo	TIPO_CARGA	VA25	Si
Complemento de la Descripción del Vehículo	COMPLEMENTO	VA100	Si
Número de Ocupantes	NUM_OCUPANTES	I	No
Descripción Remolque	DESCR_REMOLQUE	VA50	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Tarifas Asignadas al Agente

Descripción: Tarifas Asignadas al Agente
Entidad: CM_C_TARIFAS

Atributos

Descripción	Campo	Tipo	NULO
Clave Tarifa Agente	CVE_TARIFA_AGE	I	Si
Clave Tarifa Aseguradora	CVE_TARIFA_ASE	VA25	No
Fecha de Alta	FEC_ALTATARIFA	D	Si
Fecha de Termino	FEC_TERMTARIFA	D	No
Paquete Integrado	PAQ_INTEGRADO	BT1	Si
Número de Oficina Referencia de la Aseguradora	OFICINA_ASE	VA10	Si
Número de Folio asignado por la Aseguradora	NUM_FOLIOASE	VA30	No
Aplicar Equipo Especial	APLICA_EQESPECIAL	BT1	Si
Importe de los Gastos de Expedición	IMP_GTOEXP	F	Si
Aplicar Gastos de Expedición en el Primer Recibo	APLICAR_1ERECIBO	BT1	Si
Número de Zona Asignada por la Aseguradora	ZONA_ASE	VA5	Si
Aplicar Valor Factura en esta Tarifa	APLICA_VALORFACTURA	BT1	Si
Aplicar Seguro de Vida a la Tarifa	APLICA_SEGVIDA	BT1	Si
Aplicar Recibos Irregulares	APLICA_REC_IRREGULAR	BT1	Si
Días Límite para cobro de la póliza	DIAS_COBRO	I	Si
Leyenda del Contratante	LEYENDA_CONTRATANTE	VA255	No
Incluir el Importe del Valor Factura en la Póliza	APLICA_IMPVALFACT	BT	Si
Incluir Importe del Valor del Deducible	APLICA_IMPVALDEDUCIBLE	BT	Si
Entidad de Referencia para Límites del Valor del Vehiculo	ENTIDAD_REFLIMITES_VEHI	VA25	No
Entidad de Referencia Límite de suma asegurada equipo especial	ENTIDAD_REFLIMITES_EQESPECIAL	VA25	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Tipo de Persona

Descripción:	Tipo de Persona
Entidad:	CM_C_TIPOPERSONA

Atributos

Descripción	Campo	Tipo	NULO
Clave tipo Persona	CVE_TIPOPERSONA	I	Si
Descripción de la Persona	DESC_PERSONA	VA25	Si
Bandera de activación	ACTIVO	BT1	Si

Entidad Tipo de Usuario

Descripción:	Tipo de Usuario
Entidad:	CM_C_TIPO_USUARIO

Atributos

Descripción	Campo	Tipo	NULO
Clave del tipo de usuario	CVE_TIPO_USUARIO	I	Si
Clasificación del Usuario	CLASIFICACION_USUARIO	VA1	Si
Descripción del tipo de usuario	DESC_TIPO_USUARIO	VA50	Si
Bandera de Activación	ACTIVO	BT1	Si

Entidad Vehículo Entidad Clave

Descripción:	Vehículo Entidad Clave
Entidad:	CM_A_VEHICULOENTCLAVE

Atributos

Descripción	Campo	Tipo	NULO
Identificador de la Clave del Vehículo	ID_VE_ENTIDAD	I	Si
Clave del Vehículo de la Entidad	CVE_VEHI_ENTIDAD	VA20	Si
Fecha Alta Vehículo	FEC_CARGACVE_VEHI	D	Si

4.2.4 Diagrama de Control de Acceso al Sistema

Para la validación de usuarios se hará uso del siguiente modelado que se muestra en la figura 27.

En la autenticación y control de acceso de los usuarios, el usuario tendrá definida un área, un puesto y un tipo de usuario que lo podrán clasificar al momento de ingresar al sistema.

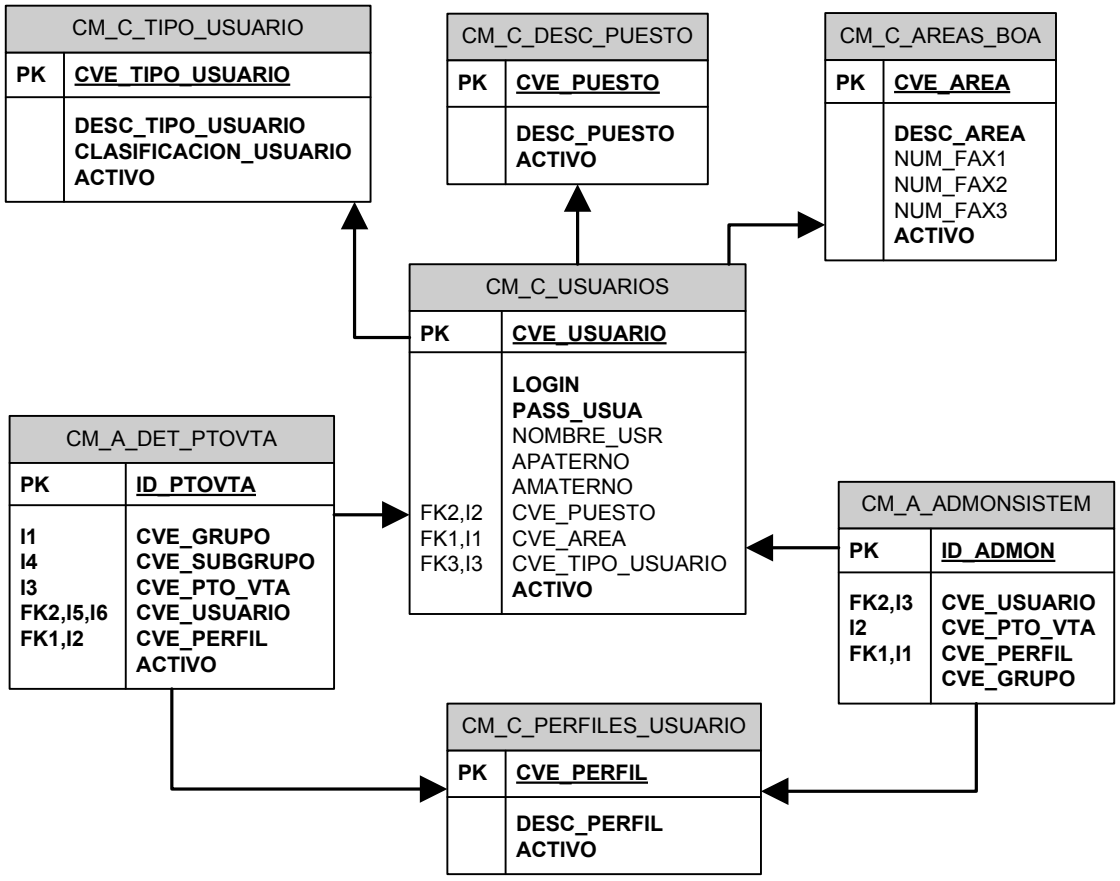


Figura 27. Modelo para la validación de usuarios.

Una vez que el usuario es autenticado este además de conocer su nivel de usuario y sus atributos que complementan la identificación del mismo, como parte el esquema de seguridad se le debe permitir el uso de ciertos módulos. Estos módulos estarán delimitados por el perfil que tenga cada uno de los usuarios. Que significa esto, por

ejemplo: Juan Pérez pertenece a una agencia x y el puesto de él es vendedor, su área es la de ventas de seguros y de acuerdo a su perfil sólo debe poder emitir pólizas. Supongamos que este control no existiera, Todos los usuarios podrían hacer uso de todos los módulos disponibles en la aplicación.

Es por ello que cada usuario se le debe asignar un aplicativo dependiendo el grupo al que este pertenezca, con ello no podrá hacer uso de otro que no se le haya asignado.

Definición de los administradores del sistema de emisión de pólizas.

Como medida de contingencia y atención a clientes, un usuario de nivel administrador puede emitir a nombre de cualquier punto de venta que tenga asignado. Esto es, cuando alguna agencia ó cliente no cuenta con Internet ó por causas ajenas al sistema no puede generar sus pólizas, personal operativo debe tener la faculta de realizar emisión de pólizas según el punto de venta que tenga esta incidencia. Una alternativa sería conocer su usuario y contraseña para poder realizar esta operación, sin embargo bajo requerimiento el operador debe estar facultado a través del sistema para realizar esta operación con el solo hecho de seleccionar el punto de venta a administrar.

4.2.5 Diseño de Interfaz Gráfica.

En la figura 28 se muestra el comportamiento que va a tener la aplicación Web. Primero se accede a la página de inicio, una vez validado el usuario, se mostrará una página por defecto con su respectiva carga del menú considerando el perfil del usuario. Posteriormente podrá cotizar, reimprimir y consultar y se puede estar moviendo entre estas opciones.

Cuando cotiza tiene la opción de Imprimir la cotización, ver las coberturas que incluye la cotización y también emitir una póliza.

Puede imprimir pólizas cuando emite una póliza, en la reimpresión o desde el módulo de consultas.

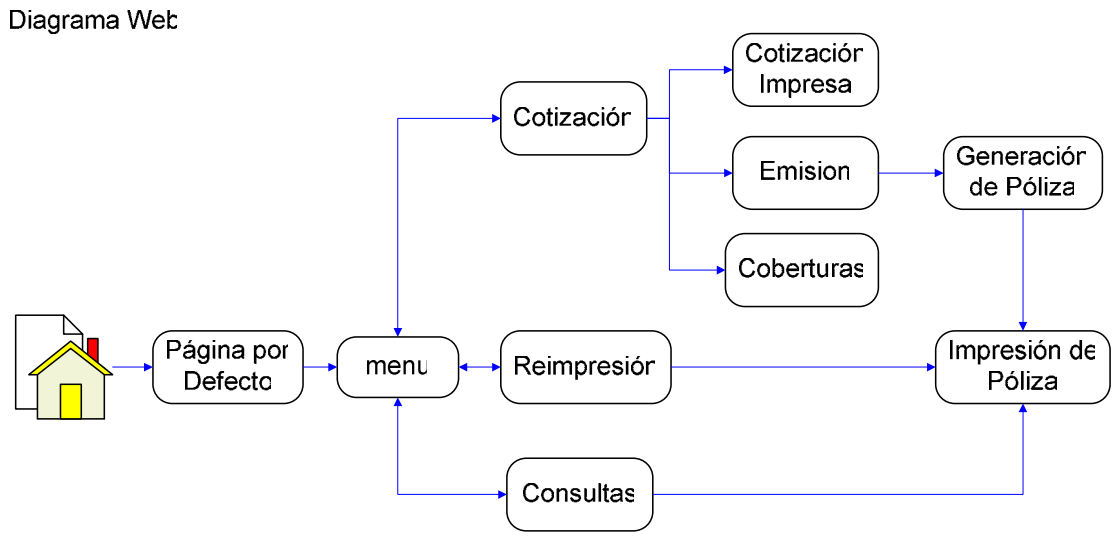


Figura 28. Diagrama Web.

La interfaz gráfica del usuario estará dada mediante las siguientes pantallas.

La figura 29, representa la pantalla de inicio, aquí se piden los datos de usuario y contraseña.

Validación de campos.

Campo	Obligatorio	Otras validaciones.
Usuario	si	Sólo acepta caracteres alfanuméricos.
Contraseña	si	Sólo acepta caracteres alfanuméricos.

Figura 29. Pantalla de inicio.

Para establecer el menú se coloca la información del usuario como es el área al cual pertenece, un link de salida del sistema, así como el programa al cual entro.
Así mismo muestra los módulos disponibles para el usuario, tal como se muestra en la figura 30.

Figura 30. Carga de Menú.

El formulario de cotización esta formado por los siguientes campos:

- Agencia: es el punto de venta que cotiza y emite pólizas.
- Tipo de Vehículo: Se refiere al vehículo a cotizar, es decir si se trata de un automóvil o un camión.

- Marca: Marca del vehículo (Ford, Chevrolet, Honda, etc.)
- Submarca: Submarca del vehículo de acuerdo a una marca, es decir si se trata de la marca Ford la submarca es Fiesta, Focus, etc.
- Año: Modelo del vehículo
- Descripción: Características del vehículo
- Valor Factura: El costo del vehículo.
- IVA: El IVA que se considera para hacer los cálculos.
- Estado: Representa el estado donde circula la unidad.
- Plazo: Es el periodo que tendrá la póliza.
- Servicio: El servicio que tiene el automóvil, por ejemplo, particular, público, etc.

Todo lo anterior se presenta en la figura 31.

Figura 31. Forma de Cotización.

Validación de campos.

Campo	Obligatorio	Otras validaciones.
Valor Factura	Si	Sólo acepta dígitos
Agencia	Si	
Tipo de Vehículo	Si	
Marca	Si	
Submarca	Si	
Año	Si	
Descripción	Si	
IVA	Si	
Estado	Si	
Plazo	Si	
Servicio	Si	

El formulario de emisión esta formado por los siguientes campos:

- Persona: Tipo de persona que emite pólizas.
- Nombre: Nombre completo del asegurado.
- RFC: RFC del asegurado.
- Fecha de nacimiento: El día en que nació el asegurado.
- E-Mail: Correo electrónico del asegurado.
- Conductor: Quien conduce el vehículo.
- Calle, CP, Colonia, Delegación, Estado: Dirección completa del asegurado
- Lada: Número de lada de su ubicación geográfica del asegurado.
- Teléfono: Número telefónico de su casa o trabajo del asegurado.
- Extensión: Número de extensión de su trabajo del asegurado.
- Fax: Número de fax de su trabajo del asegurado.
- Celular: Teléfono móvil del asegurado.
- Número de motor del vehículo.
- Número de serie del vehículo.
- Número de placas del vehículo.
- Color del vehículo.

- Inicia Póliza: Fecha en la cual la póliza empieza a tener vigencia.
- Vence Póliza: Fecha en que la póliza se vence.

Todo lo anterior se muestra en la figura 32.

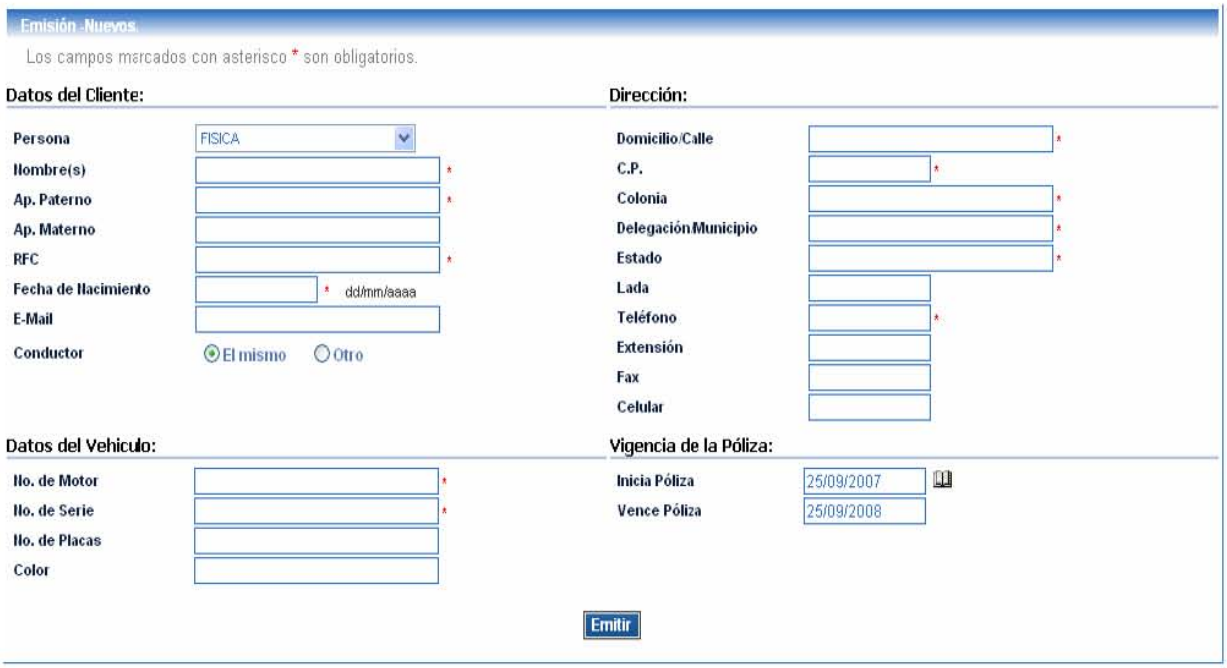


Figura 32. Forma de Emisión.

En la tabla 11 se aprecia la validación de campos para la emisión de la póliza.

Campo	Obligatorio	Otras validaciones.
Nombre	Si	Acepta caracteres alfanuméricos
Apellido Paterno	Si	Acepta caracteres alfanuméricos
Apellido Materno		Acepta caracteres alfanuméricos
RFC	Si	Acepta caracteres alfanuméricos
Fecha de Nacimiento	Si	Valida el formato dd/mm/aa
E-Mail		Valida Formato correcto de E mail
Nombre Conductor	Si	Acepta caracteres alfanuméricos
A Paterno Conductor	Si	Acepta caracteres alfanuméricos
A Materno Conductor		Acepta caracteres alfanuméricos
Domicilio/Calle	Si	Acepta caracteres alfanuméricos
C.P.	Si	Solo acepta dígitos
Colonia	Si	Acepta caracteres alfanuméricos
Delegación/Municipio	Si	Acepta caracteres alfanuméricos
Estado	Si	Acepta caracteres alfanuméricos
Lada		Acepta caracteres alfanuméricos
Teléfono	Si	Solo acepta dígitos
Extensión		Acepta caracteres alfanuméricos
Fax		Acepta caracteres alfanuméricos
Celular		Solo acepta dígitos
Número de Motor	Si	Acepta caracteres alfanuméricos
Número de Serie	Si	Acepta caracteres alfanuméricos
Número de Placas		Acepta caracteres alfanuméricos
Color		Acepta caracteres alfanuméricos

Tabla 11. Validación de campos para la emisión

Para el módulo de consultas se presenta la selección de un filtro de búsqueda y una agencia como se muestra en la figura 33.

Figura 33 Consultas.

En la figura 34, se presenta la parte de reimpresión con el campo de número de póliza a reimprimir.

Figura 34. Reimpresión.

4.3 CODIFICACIÓN.

En este tema se explicará como se manejo el proyecto en el código, así como técnicas de codificación y una especificación técnica de funciones.

4.3.1 Estructura para organizar un Proyecto WEB de .NET

Para tener un estándar en la realización del proyecto se hizo lo siguiente:

A) Se creo una solución buscando el nombre más apropiado para la agrupación del o de los proyectos.

B) Se agrupó y nombró las carpetas y archivos de la siguiente manera:

- Nombre del Proyecto: Palabra “Web” y el nombre del proyecto elegido, ejemplo, *WebAdmon*.
- Clases: Contendrá todas las clases necesarias para el proyecto, en esta parte se define el acceso de datos y las reglas de negocio. Se nombrará con la primera letra “**C**” y el nombre de la clase, ejemplo *Ccatalogos*.
- Images: Esta carpeta contendrá todas las imágenes que se utilizan en el proyecto.
- Librería: Esta carpeta contendrá todos los archivos que serán globales para el proyecto. Se integrará de subcarpetas como js, funciones de javascript, css, hojas de estilo, etc.
- Módulos: Contendrá métodos públicos generales para todo el proyecto. Se nombrará con la primera letra “M” y el nombre del módulo, ejemplo *Mmensajes*.
- Carpetas de módulo o submódulo: Estas carpetas contendrán las páginas que corresponden a la interfaz del usuario para un módulo en especial. Por ejemplo la carpeta derivación.

C) De acuerdo a las necesidades del proyecto los siguientes archivos se utilizan en la aplicación.

- Global.asax: Se establece el proceso a seguir cuando se carga la aplicación, cuando ocurre un error o cuando se termina de ejecutar la aplicación.
- Web.config: establece la configuración apropiada para el proyecto
- CaducoSesion.aspx: Su función es regresar a la pantalla de inicio cuando la sesión haya caducado.
- Default.aspx: Pantalla por defecto que se muestra al tener una validación exitosa.
- ErrorServer.aspx: Pantalla que se muestra cuando ocurre un error en la aplicación.
- Login.aspx: Pantalla de validación del usuario, si el usuario no se ha validado se redirecciona a esta pantalla.

Lo anterior se puede visualizar en la figura 35.

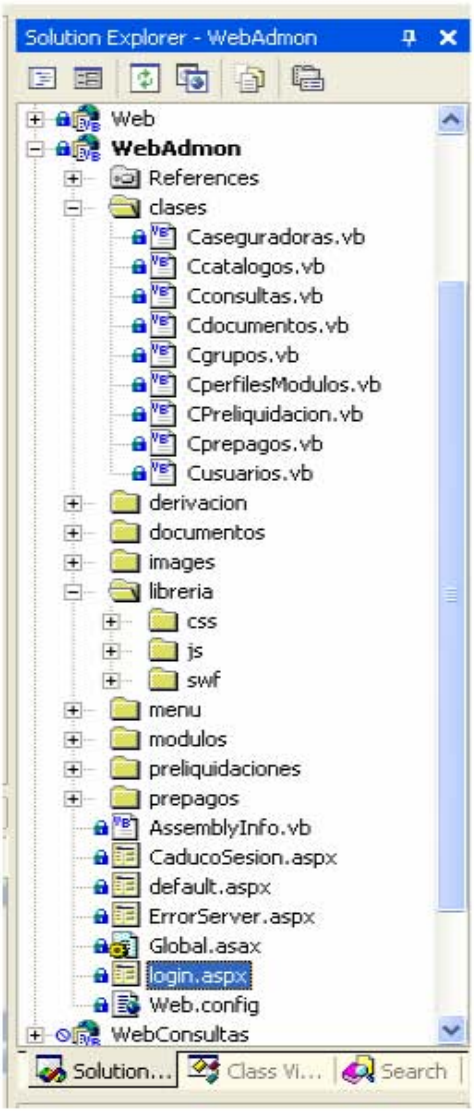


Figura 35. Estructura del proyecto en .NET.

La estandarización de código que se sigue, para el manejo de variables y el manejo de controles Web Form y Objetos de ADO.NET son los que se muestran en las siguiente tablas.

4.3.2 Prefijos para Tipo de Datos.

La tabla 12, lista las variables estándar y el tipo de datos para manejarse en visual Basic.net, con su respectivo prefijo y un ejemplo.

Tipo de Datos	Prefijo	Ejemplo
Array	a	aValores
Bolean	b	bIsValid
Byte	by	byValue
Char	c	cLetter
Date	dt	dtStart
Decimal	dec	decValue
Double	d	dValue
Integer	i	iLoop
Long	l	lValue
Object	o	oValue
Short	shr	shrValue
Single	sng	sngValue
String	s	sDescripción

Tabla 12. Prefijo para Tipo de Datos.

Nombrado de Controles Web Form.

Se sugiere usar prefijos para indicar el tipo de control que se va a usar en la interfaz, para identificar fácilmente en el código a que campo corresponde. La tabla 13 lista los controles Web Form con su respectivo prefijo que se recomienda usar.

Control	Prefijo
Label	lbl
TextBox	txt
Button	btn
LinkButton	lnk
ImageButton	imb
HyperLink	hyp
DropDownList	ddl
ListBox	lst
DataGrid	dg
DataList	dl
Repeater	rep
CheckBox	chk
CheckBoxList	cbl
RadioButtonList	rbl
RadioButton	rdo
Image	img
Panel	pnl
PlaceHolder	plc
Calendar	cal
AdRotator	ad
Table	tbl
RequiredFieldValidator	reqv
CompareValidator	cmpv
RangeValidator	rngv
RegularExpressionValidator	rexpv
CustomValidator	custv
ValidationSummary	vsum
Xml	xml
Literal	lit
CrystalReportViewer	crv

Tabla 13 Prefijos de Controles Web Form.

Nombrado de Objetos de ADO.NET

La tabla 14 lista los prefijos más comunes que se usan para establecer comunicación con la base de datos.

Clase	Prefijo para Objeto
DataSet	ds
DataTable	dt
DataView	dv
DataRow	drw
DataRowView	drv
Connection	cnn
Command	cmd
DataAdapter	da
CommandBuilder	bld
DataReader	dr

Tabla 14 Prefijo para clases ADO.NET

Estos objetos aplican para OleDb, SQLClient, u otro Namespace dependiendo del driver que estemos usando para acceder a la base de datos.

Ejemplos en el código cuando usamos una instancia particular de un objeto.

```
Dim dr As New SqlDataReader()
Dim ds As DataSet
```

Ejemplos si se necesitan más de una instancia para cargar objetos ADO.NET.

```
Dim drEmps As New SqlDataReader()
Dim  drCust As New SqlDataReader()
Dim  dsEmps As DataSet
```

4.3.3 Diagrama de Estados.

Este diagrama de la figura 36 representa la secuencia de los estados que interactúan en la aplicación.

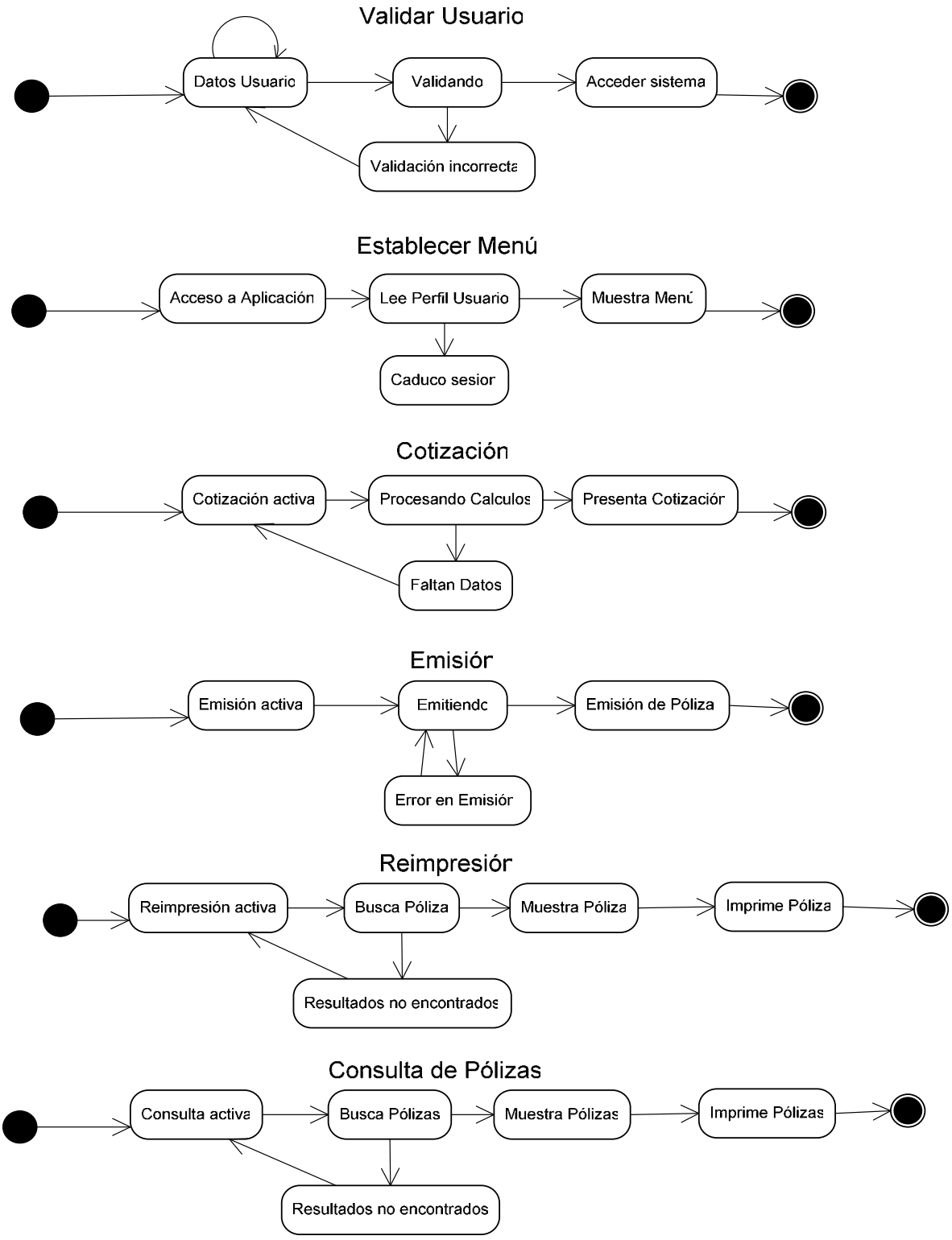


Figura 36. Diagrama de Estados.

4.3.4 Diagrama de actividades.

Este diagrama lo utilizamos para visualizar, especificar y documentar el flujo de control que siguen las operaciones internas del sistema. Estos son la generalización de los casos de uso para formar una actividad o un subsistema, se puede apreciar esto en las figuras 37 y 38.

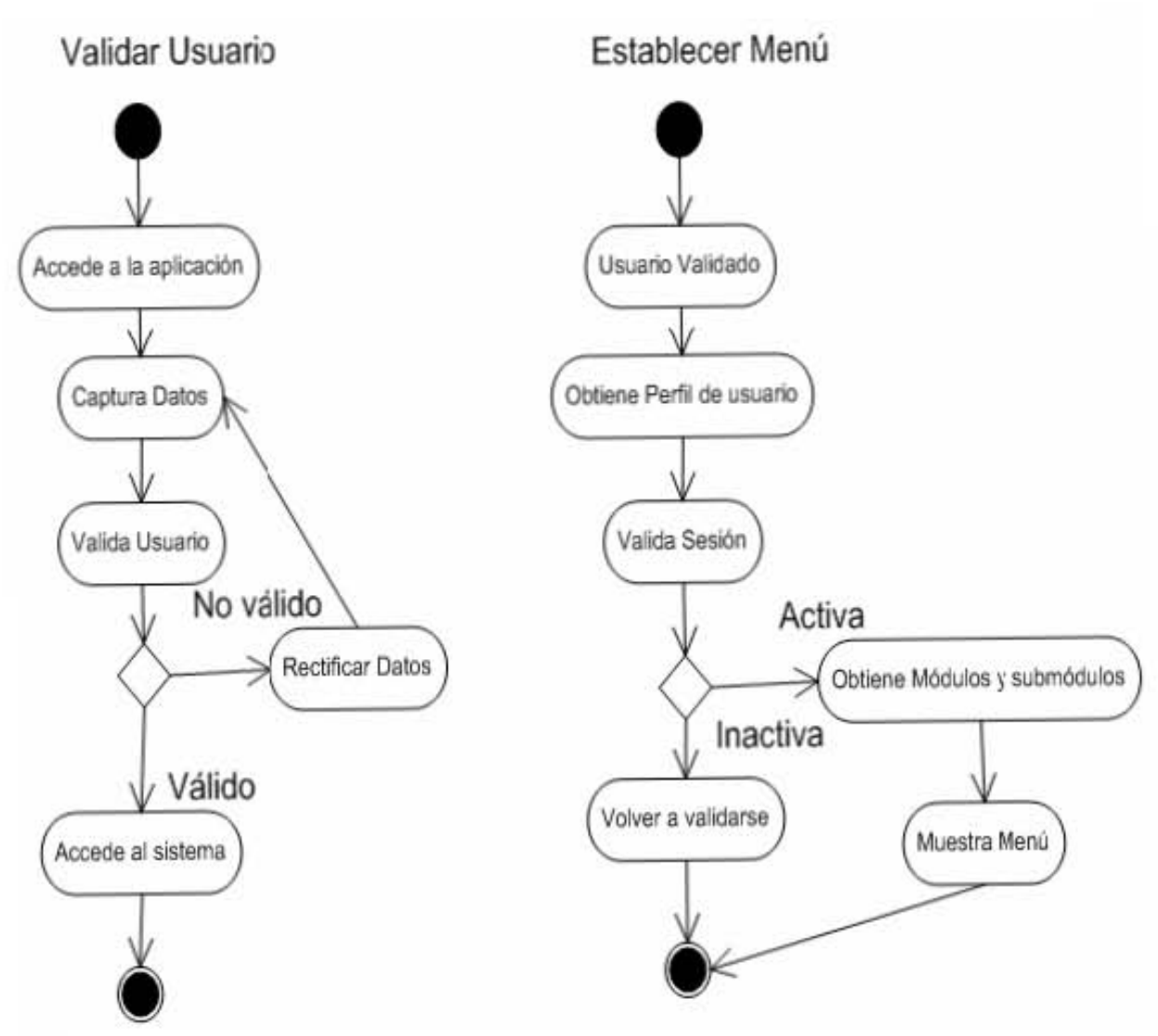


Figura 37. Diagrama de Actividades 1.

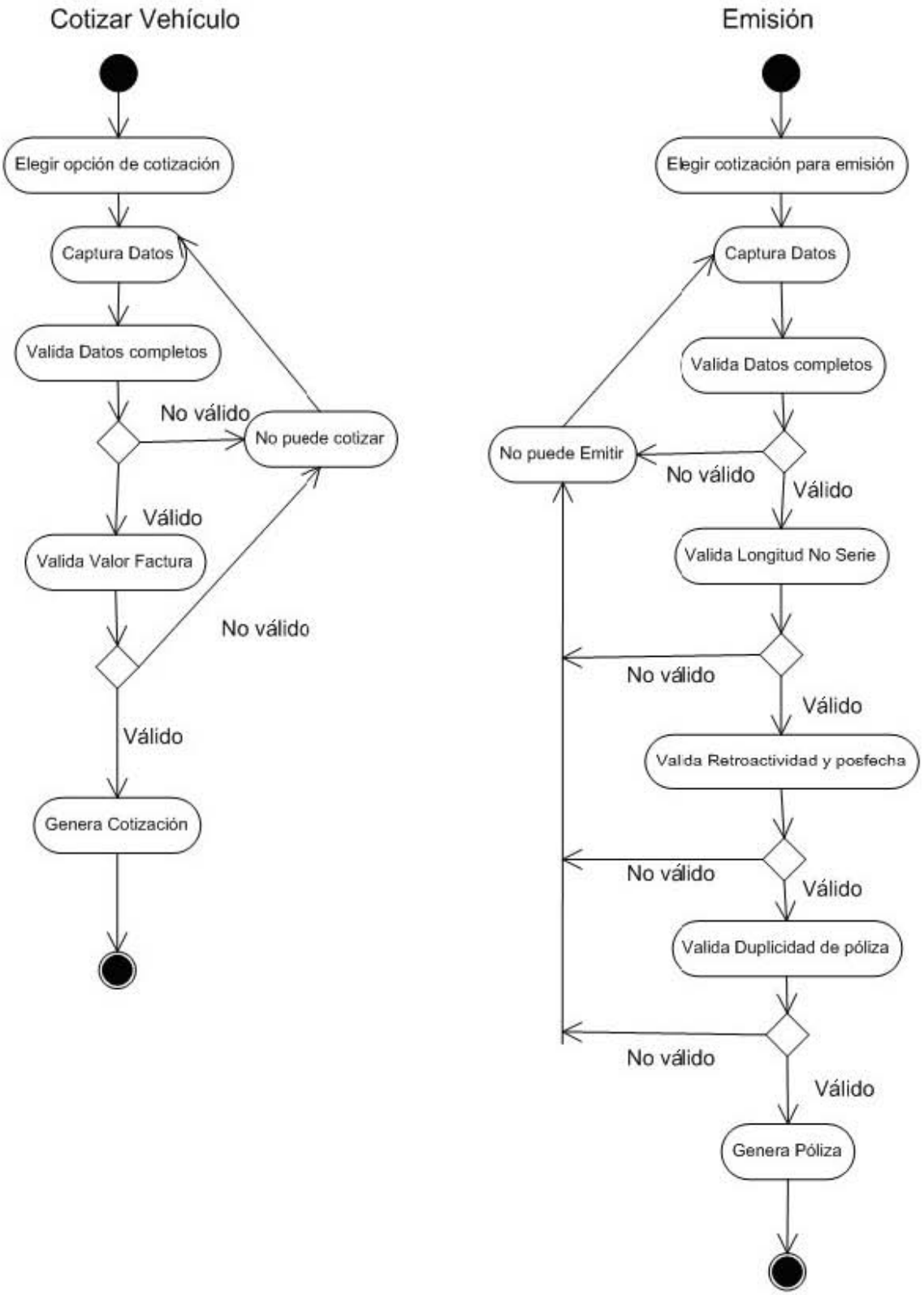


Figura 37. Diagrama de actividades 2.

4.3.5 Especificación Técnica de funciones.

Para tener una idea clara de ver como esta integrado nuestro código, a continuación trataremos de detallar lo que se uso.

Las páginas que contiene la aplicación se clasifican de acuerdo al proceso que tienen que cumplir.

Para llevar a cabo el proceso de cotización se usaron las páginas siguientes.

- Cot.aspx: Formulario de cotización.
- Cotimp.aspx: Impresión de la cotización.
- DetallePaquete.aspx: Detalle de las coberturas de cotización.

Para la emisión:

- Emision.aspx: Formulario de emisión.
- metodosServer.aspx: Aquí se definen métodos que se ejecutan por medio de remote scripting.
- Reportes.aspx: Presentación de la póliza emitida y para imprimirla.
- ValidaPoliza.aspx: Se presenta cuando la póliza esta duplicada.

Para la reimpresión:

- RI.aspx: Forma de reimpresión.

Para las consultas:

- ConsultaPoliza.aspx: Página donde se presenta el listado de las pólizas hechas en la consulta.

4.4 PRUEBAS

Los conceptos y definiciones de integración y pruebas del sistema.

La integración es una técnica sistemática que consiste en poner juntos a los módulos para detectar errores asociados con la interacción.

Las pruebas constituyen un parte integral y vital del desarrollo del sistema las cuales tienen como propósito describir defectos que se establecen para mejorar la calidad del sistema.

El objetivo de la fase de pruebas del sistema es el de detectar toda posible falla en el funcionamiento antes de que entre en operación.

4.4.1 Pruebas realizadas al sistema.

Para la validación de usuarios se ingreso un usuario que no estaba en la base de datos y nos mando un mensaje de error como se ve en la figura 39.

Figura 39. Usuario Inválido.

También se valido la entrada de campos obligatorios en la pantalla de inicio y nos generó el mensaje mostrado en la figura 40.

Figura 40. Campos Obligatorios en la Validación.

Para validar que la sesión tiene una duración de 20 minutos, se dejo inactivo 21 minutos y esto fue lo que nos mostró de acuerdo a la figura 41.

Figura 41. Sesión Inactiva.

Otra validación que se hizo, fue la de ingresar a páginas de cotización y emisión sin pasar por la validación de usuarios y el resultado fue que siempre nos mandaba a la página de inicio.

4.4.2 Requerimientos mínimos del sistema.

Todo sistema requiere de características específicas con las que se deben de cumplir para su buen funcionamiento. Estos requerimientos pueden ser de software y de hardware. Probablemente si no se cumple con estos requerimientos mínimos se pueden presentar errores o puede que no funcione correctamente, a pesar de estar bien diseñado y desarrollado.

Los requerimientos son:

- Para el servidor.
    - Procesador Pentium IV a 2.0 GHZ
    - 256 MB en memoria RAM
    - 1 GB de espacio libre en disco
    - Windows 2000 Advanced Server
    - Microsoft SQL Server 2000 Developer Edition
    - Framework 1.1

- Para el usuario.
    - Pentium III 500mhz
    - Disco Duro 700mb de espacio libre
    - 128 memoria RAM
    - Windows 2000 profesional Service Pack 4
    - Windows XP Professional Service Pack 1
    - Internet Explorer 6 Service Pack 1
    - Acrobat Reader 5.1
    - Flash Player 6.0 ó superior

Este sistema integra un emisor de Internet con nueva tecnología para garantizar un servicio de calidad y de fácil uso. Ahora tu sistema es 4 veces más rápido que otros.

CONCLUSIONES

En cada una de las etapas que conformaron el proyecto se aplicaron conocimientos que fueron adquiridos durante la carrera y la experiencia laboral, a si mismo otros tuvieron que ser aprendidos para poder continuar con el desarrollo y obtener el sistema final.

Se concluye que la metodología RUP es una de las mejores que existen para el desarrollo de sistemas basados en la programación orientada a objetos, todo esto por su buena respuesta a los distintos casos de uso y la facilidad para adaptarse a las herramientas de desarrollo de software utilizadas.

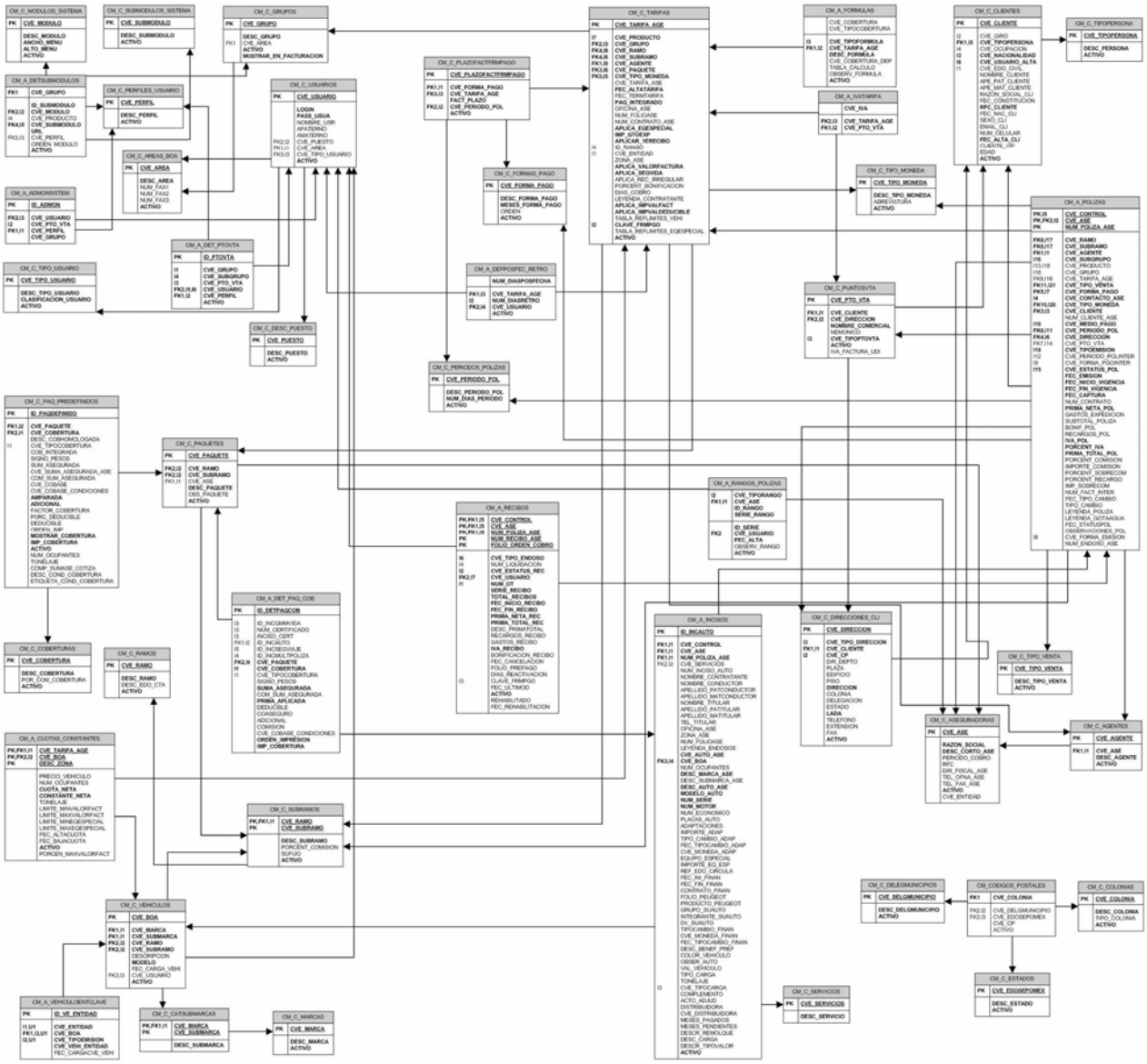
Se cubrieron todas las expectativas y necesidades planteadas por el cliente en el sistema solicitado, de esta forma el sistema es seguro, confiable, flexible, portable y eficaz para cotizar y emitir pólizas de las distintas aseguradoras con la que cada agencia de autos tenga algún convenio además de que cada usuario tendrá una clave y contraseña con los permisos autorizados de la agencia de autos o punto de venta al que pertenezca, para evitar así un mal uso del sistema.

Se logró integrar toda la información en un sistema que en un principio se encontraba dispersa, obteniendo como resultado una administración sencilla de cotización y emisión de pólizas.

Este sistema es altamente competitivo y eficiente por su doble impacto, por una parte reduce tiempos y trámites a los clientes interesados en adquirir una póliza y por la otra permite reducir costos y a su vez aumentar la productividad al momento que las agencias de autos otorguen el servicio.

ANEXOS

ANEXO I - DIAGRAMA ENTIDAD-RELACIÓN



BIBLIOGRAFÍA

Pressman Roger S. Ingeniería del software. Quinta edición, McGraw-Hill. USA 2002.

LIGAS

http://fisi.espe.edu.ec/~elascano/distribuidas/documentacionyejemplos/aplicaciones2_3_n_capas.doc

http://people.cs.uchicago.edu/~borja/pubs/revistaeside2002.pdf

https://pid.dsic.upv.es/C1/Material/Documentos%20Disponibles/Introducción%20a%20RUP.doc

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/cpguide/html/cpconaspstatemanagement.asp

http://es.wikipedia.org/wiki/Visual_Basic

http://recursos.dotnetclubs.com/.../DNC-Sevilla-11-Aplicaciones%20con%20N%20capas%202/Dnc-Sevilla-Arquitecturas.ppt

http://www.alegsaonline.com/art/13.php

http://www.desarrolloweb.com/manuales/48/

http://www.microsoft.com/net/

http://www.mslatam.com/latam/msdn/comunidad/dce2005/

http://www.pdsa.com/Tips/DotNetProgrammingStds.doc

http://www.willydev.net/descargas/articulos/general/cualxpfddrup.PDF

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/dwamish7/html/vtgrfArchitecturalOverviewOfDuwamish70.asp