

UNIVERSIDAD CENTRAL DE VENEZUELA  
FACULTAD DE CIENCIAS  
ESCUELA DE COMPUTACIÓN



CENTRO DE INVESTIGACION EN SISTEMAS DE INFORMACIÓN  
(CISI)

Aplicación web para la cotización, compra y manejo de pólizas de seguro de  
dispositivos móviles

Trabajo especial de grado presentado ante la ilustre  
Universidad Central de Venezuela por los bachilleres:

Br. Pascale Marco  
Br. Varguillas Lenys  
Tutora: Profa. Concettina Di Vasta

Octubre 2016

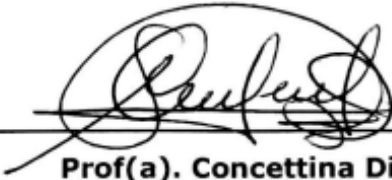
UNIVERSIDAD CENTRAL DE VENEZUELA  
FACULTAD DE CIENCIAS  
ESCUELA DE COMPUTACIÓN

ACTA

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado titulado "Aplicación web para la cotización, compra y manejo de pólizas de seguro de dispositivos móviles" y presentado por los bachilleres: Br. Marco Pascale C.I.: 15.379.286 y Br. Lenys Varguillas C.I.: 14.953.817, a los fines de optar al título de **Licenciado en Computación**, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día 17 de Octubre de 2016, a las 11 horas, para que los autores lo defendieran en forma pública, lo que estos hicieron en la Sala PBIII de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondieron a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de 20 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 17 de Octubre de 2016.

  
\_\_\_\_\_  
**Prof(a). Concettina Di Vasta**  
**(Tutora)**  
\_\_\_\_\_  
**Prof(a) Mercy Ospina**  
**(Jurado)**  
\_\_\_\_\_  
**Prof. Franklin Sandoval**  
**(Jurado)**



## **Dedicatorias**

Agradezco primeramente dios por ayudarme alcanzar esta meta y estar a mi lado siempre guiando mis pasos.

A mis padres Luis y Ramona quienes en vida lo dieron todo por mis hermanos y por mí; que con sus esfuerzos y su amor invaluable siempre quisieron lo mejor para mi impulsándome siempre a ser mejor persona y lograr esta meta tan importante. Así que por esto y muchas cosas más te doy las gracias papá porque tus consejos me ayudan a tomar decisiones correctas; a mi madre que nos enseñó a querer con todo su corazón sin esperar nada a cambio, siempre protectora, siempre presente mamá, no hay palabras para expresarte lo mucho que te quiero. Estoy seguro que desde el cielo me siguen cuidando mis padres.

A mis hermanos Luis y Marbelys por ser parte importante en mi vida y representar la unidad familiar y el constante apoyo incondicional.

A mis hijos que si nos los tuviera mi vida sería un desastre, porque con ellos siento más ganas de trabajar fuertemente y seguir con el objetivo de alcanzar mis metas, son mi principal motivación.

A Nicolina que siempre me acompaña y apoya inclusive en los momentos y situaciones más tormentosas. ¡Muchas Gracias!

A mis abuelos, tíos y otros familiares en general por su apoyo.

A Mariangelica Monasterios por su gran amistad y apoyo.

A Mirian Prieto por su amistad y apoyo durante todo este tiempo.

A todas las familias que siempre me acompañan y apoyan, a los compañeros y amigos con los que compartí durante todo este tiempo; Gracias por esos momentos que compartimos y los que vendrán. A todos mis hermanos de vida. Gracias a Oskar, Francisco, Cesar, Fernando, Juan, y demás compañeros del Amper por todos los momentos que compartimos.

A mi tutora Concettina Di Vasta por el apoyo y la confianza que me brindo para realizar este Trabajo Especial de Grado (TEG), quien con su experiencia, paciencia y motivación ha logrado que pueda terminar mis estudios con éxito.

Son muchas las personas que han formado parte de mi vida a las que me encantaría agradecerles su amistad, consejos, apoyo, ánimo y compañía en los momentos difíciles de mi vida. Sin importar donde estén quiero darle las gracias, para ellos muchas bendiciones.

A mi compañero de tesis con el cual fue un placer haber desarrollado este trabajo especial de grado (TEG).

**Lenys Varguillas**

A Dios por darme la salud, voluntad, ánimo, ganas y en los momentos más difíciles la fuerza necesaria para poder completar una más de mis grandes metas. Junto a él sé que están mis padres, que me iluminaron, guiaron y cuidaron.

A mis hermanos los cuales sin ellos no sería nada de lo que hoy soy. Pieza súper fundamental para poder lograr esta meta. Su constancia, perseverancia, paciencia, cariño, amor y dedicación fueron algunas de las tantas cosas que hicieron por mí para poder lograrlo.

A ese ser tan único y especial que es Magly Mendez que durante todo este proceso de trabajo especial de grado, estuvo a mi lado dándome apoyo, fuerza, consejos y sobre todo mucho amor. Por permitirme estar a tu lado, por hacerme tan feliz y poder compartir este éxito contigo. ¡Te Amo!

A mi amigo y compañero de trabajo Lenys Varguillas el cual fue participe en gran parte para que todo esto fuese posible. El único en verdad que puede entender por todo lo que pasamos para llegar hasta aquí, difícil pero no imposible. A su familia por tanta amabilidad, hospitalidad y cariño.

A mis amigos los cuales pusieron su grano de arena para que mis estudios fuesen inolvidables y únicos, para aprender y vivir experiencias que solo se dan rodeadas de gente como ustedes.

A mis compañeros de trabajo, especialmente a Ricardo Ubieta, por su tiempo y ganas de apoyarme cada vez que necesitaba de una ayuda o de una idea "simpática" que programar.

Para todos ustedes...

¡GRACIAS!

**Marco Pascale.**

## **Resumen**

El presente trabajo especial de grado se basó en el desarrollo de una aplicación web para la cotización, compra y manejo de pólizas de seguros para dispositivos móviles, la cual permite gestionar de manera eficiente, rápida y económica, todo lo relacionado al proceso de negocio de los seguros para dispositivos móviles y esto se logra porque al ser un sistema automatizado reduce la utilización de recursos y permite que todo el proceso sea manejado con la menor cantidad de personas.

El proyecto fue desarrollado usando la metodología de Programación Extrema XP (eXtreme Programming); el correcto uso de las metodologías definidas para su implementación facilitan la planificación y desarrollo de la aplicación a lo largo de todo su ciclo de vida, así como la documentación necesaria de la misma, para suministrar información a cualquier tipo de trabajos a futuro que se basen sobre esta aplicación.

Adicionalmente, para el desarrollo de la aplicación se utilizaron las herramientas, frameworks y lenguajes que son utilizados ampliamente a nivel mundial tales como CakePHP, RubyOnRails, html5, javascript, JQuery, Amazons3, css3 y otros, lo que asegura el acceso a una extensa documentación a la cual consultar. Esto ayuda a la creación de los distintos módulos planteados en este trabajo, al igual que su posterior mantenimiento y extensión.

Para cualquier empresa que maneje seguros le será muy beneficiosa una aplicación a la cual acceder desde cualquier dispositivo que soporte el estándar HTML5, la cual permita:

- Cotizar y comprar un seguro (en este caso para dispositivos móviles).
- Manejar pagos automatizados a través de la aplicación.
- Gestionar el proceso de reclamos permitiendo al usuario colocar un reclamo y a la empresa analizar el reclamo y darle una solución.
- Administrar todos los dispositivos, reclamos y configuración de la aplicación.

**Palabras Claves:** Cotización, Pólizas, Aplicación Web, Programación Extrema.

|                                                             |    |
|-------------------------------------------------------------|----|
| <b>Índice</b>                                               |    |
| Dedicatorias.....                                           | 1  |
| Resumen .....                                               | 5  |
| Índice .....                                                | 6  |
| Índice de Figuras .....                                     | 8  |
| Índice de tablas.....                                       | 10 |
| CAPÍTULO I. Problema de Investigación .....                 | 12 |
| 1.1 Planteamiento del Problema.....                         | 12 |
| 1.2 Objetivos.....                                          | 13 |
| 1.2.1 General .....                                         | 13 |
| 1.2.2 Específicos .....                                     | 13 |
| 1.3 Alcance .....                                           | 13 |
| 1.4 Importancia y justificación.....                        | 14 |
| CAPÍTULO II. Marco Conceptual .....                         | 15 |
| 2.1 Póliza de Seguros .....                                 | 15 |
| 2.2 Elementos relacionados con una Póliza de seguros.....   | 15 |
| 2.2.1 Compañía de seguros .....                             | 15 |
| 2.2.2 Accidente .....                                       | 15 |
| 2.2.3 Asegurado.....                                        | 15 |
| 2.2.4 Asegurador .....                                      | 15 |
| 2.2.5 Aviso de Siniestro .....                              | 15 |
| 2.2.6 Prima de seguro.....                                  | 16 |
| 2.3 Aplicaciones Web .....                                  | 16 |
| 2.3.1 Modelo Cliente-Servidor .....                         | 17 |
| 2.3.2 Patrón de arquitectura Modelo-Vista-Controlador ..... | 18 |
| 2.4 Tecnologías utilizadas para el desarrollo.....          | 19 |
| 2.4.1 Tecnologías del lado del cliente .....                | 19 |
| 2.4.2 Tecnologías del lado del servidor .....               | 23 |
| CAPÍTULO III. Marco Metodológico.....                       | 33 |
| 3.1 Programación Extrema (Extreme programming).....         | 33 |
| 3.1.1 Características .....                                 | 33 |
| 3.1.2 Iteraciones .....                                     | 35 |
| 3.1.3 Historias de Usuarios .....                           | 36 |

|                                                      |    |
|------------------------------------------------------|----|
| CAPÍTULO IV Marco Aplicativo.....                    | 38 |
| 4.1    Adaptación del proceso de desarrollo XP ..... | 38 |
| 4.1.1    Actores y Responsabilidades .....           | 38 |
| 4.1.2    Metáfora del Sistema .....                  | 39 |
| 4.1.3    Adaptación de las tareas XP .....           | 40 |
| 4.2    Requerimientos Generales del sistema .....    | 41 |
| 4.2.1    Requerimientos funcionales .....            | 41 |
| 4.3    Requerimientos Específicos del sistema .....  | 42 |
| 4.4    Desarrollo de la aplicación .....             | 42 |
| 4.4.1    Iteración 0 .....                           | 42 |
| 4.4.2    Iteración 1 .....                           | 44 |
| 4.4.3    Iteración 2 .....                           | 49 |
| 4.4.4    Iteración 3 .....                           | 55 |
| 4.4.5    Iteración 4 .....                           | 65 |
| 4.4.6    Iteración 5 .....                           | 69 |
| 4.4.7    Iteración 6 .....                           | 71 |
| 4.4.8    Iteración 7 .....                           | 77 |
| Conclusiones.....                                    | 83 |
| Trabajos Futuros .....                               | 85 |
| Referencias bibliográficas.....                      | 86 |

## Índice de Figuras

|                                                                                                    |    |
|----------------------------------------------------------------------------------------------------|----|
| Figura 1. Arquitectura Cliente - Servidor.....                                                     | 17 |
| Figura 2. Modelo Vista Controlador .....                                                           | 19 |
| Figura 3. JavaScript. Autor: Álvarez - 2009.....                                                   | 22 |
| Figura 4. Arquitectura de la aplicación.....                                                       | 32 |
| Figura 5. Ciclo de Vida de XP. Jaspe (2012) .....                                                  | 35 |
| Figura 6. Metáfora de la aplicación .....                                                          | 40 |
| Figura 7. Desarrollo de XP .....                                                                   | 40 |
| Figura 8. Visión del modelo relacional representando las tablas principales de la aplicación. .... | 43 |
| Figura 9. Modelo inicial del diagrama de clases.....                                               | 45 |
| Figura 10. Esquema de layout de la aplicación. ....                                                | 46 |
| Figura 11. Fragmento de código del login de usuario. ....                                          | 47 |
| Figura 12. Fragmento de código sobre la creación de usuario. ....                                  | 47 |
| Figura 13. Caso de prueba 1 .....                                                                  | 48 |
| Figura 14. Caso de prueba 2 .....                                                                  | 48 |
| Figura 15. Caso de prueba 3 .....                                                                  | 49 |
| Figura 16. Modelo mejorado del diagrama de clases de la aplicación. ....                           | 50 |
| Figura 17. Interfaz de cotización de póliza.....                                                   | 51 |
| Figura 18. Interfaz del dashboard del sistema. ....                                                | 52 |
| Figura 19. Apis publicas usadas por el sistema .....                                               | 52 |
| Figura 20. Apis públicas para la validación de los imei y el formato del número de teléfono. ....  | 53 |
| Figura 21. Controlador que maneja el dashboard del sistema. ....                                   | 54 |
| Figura 22. Extracto de código que muestra la presentación de los estatus de las pólizas. ....      | 54 |
| Figura 23. Caso de prueba 4 .....                                                                  | 55 |
| Figura 24. Código HTML que contiene la estructura del botón de pago. ....                          | 57 |
| Figura 25. Interfaz de pagos de la aplicación. ....                                                | 58 |
| Figura 26. Extracto de código donde se realiza la encriptación de la data. ....                    | 58 |
| Figura 27. Función que redirección al cliente a la pasarela de pago. ....                          | 59 |
| Figura 28. Api publica que recibe el post de la pasarela de pago y lo procesa.....                 | 59 |
| Figura 29. Segmento de configuración del servidor de mandrill. ....                                | 60 |
| Figura 30. Función que envía los emails del sistema. ....                                          | 61 |
| Figura 31. Función que convierte las vistas en pdf para su envío por email. ....                   | 62 |
| Figura 32. Certificado de póliza.....                                                              | 63 |
| Figura 33. Caso de prueba 5 .....                                                                  | 64 |
| Figura 34. Caso de prueba 6 .....                                                                  | 65 |
| Figura 35. Interfaz de reporte de siniestros. ....                                                 | 66 |
| Figura 36. Extracto de que genera el formulario de la colocación de siniestro. ....                | 67 |
| Figura 37. Función que permite subir y manejar archivos en los servidores de Amazon S3. ....       | 68 |
| Figura 38. Caso de prueba 7 .....                                                                  | 69 |
| Figura 39. Función que ejecuta el worker para la revisión diaria de las pólizas. ....              | 70 |
| Figura 40. Interfaz del dashboard de la aplicación administrativa. ....                            | 73 |
| Figura 41. Menú de configuración del sistema. ....                                                 | 74 |

|                                                                                                |    |
|------------------------------------------------------------------------------------------------|----|
| Figura 42. Extracto de código donde se configuran las fotos que muestra el sistema.....        | 75 |
| Figura 43. Extracto de código donde se configura la asignación de data de la aseguradora. .... | 76 |
| Figura 44. Caso de prueba 9.....                                                               | 77 |
| Figura 45. Interfaz del dashboard del administrador de la aseguradora.....                     | 78 |
| Figura 46. Interfaz de la sección de reportes. ....                                            | 78 |
| Figura 47. Implementación página principal del módulo de aseguradora. ....                     | 79 |
| Figura 48. Extracto de código de la generación de reportes. ....                               | 80 |
| Figura 49. Interfaz de análisis de siniestros. ....                                            | 81 |
| Figura 50. Caso de prueba 10.....                                                              | 82 |
| Figura 51. Caso de prueba 11.....                                                              | 82 |

## Índice de tablas

|                                                                |    |
|----------------------------------------------------------------|----|
| Tabla 1. Iteraciones e historias de usuario.....               | 36 |
| Tabla 2. Esquema de tabla para las historias de usuario. ....  | 37 |
| <b>Tabla 3: Actores y roles.</b> .....                         | 39 |
| Tabla 4. Historia de usuario para la iteracion 0. ....         | 43 |
| Tabla 5. Historia de usuario para la iteración 1. ....         | 44 |
| <b>Tabla 6: Historia de usuario para la iteración 2.</b> ..... | 49 |
| Tabla 7: Historia de usuario para la iteración 3. ....         | 56 |
| Tabla 8. Historia de usuario para la iteración 4. ....         | 65 |
| Tabla 9. Historia de usuario para la iteracion 5 .....         | 70 |
| Tabla 10. Historia de usuario para la iteración 6. ....        | 71 |
| Tabla 11. Funcionalidad del administrador.....                 | 72 |
| Tabla 12. Perfiles de usuario. ....                            | 73 |
| Tabla 13. Historia de usuario de la iteracion 7.....           | 77 |

## Introducción

Los desarrollos tecnológicos en la actualidad se han extendido y son de vital importancia para la sociedad, con el gran avance de la tecnología, las empresas se ven en la necesidad de automatizar en la medida de lo posible gran parte de sus procesos, por otro lado las empresas de seguro actualmente están comenzando a ofrecer pólizas de seguros para celulares y Tabletas; lo cual llevó a desarrollar una aplicación web que permitió a los usuarios gestionar ellos mismos sus pólizas y así gestionar sus siniestros desde dicha aplicación.

El presente trabajo tiene como objetivo desarrollar una aplicación Web para la cotización, contratación y gestión de pólizas de seguros de celulares, Smartphone y tabletas en Venezuela para lograr mejoras en el proceso de negocio y en la generación de documentos.

Con este trabajo especial de grado, se busca una mejoría en los procesos para la adquisición de pólizas, esto se logra ampliando el público que podrá acceder al servicio, mejorando los tiempos de respuesta para la adquisición de pólizas y el procesamiento de los siniestros, y reduciendo costos al minimizar el uso de recursos.

Para llevar a cabo el objetivo anteriormente mencionado se presenta el siguiente Trabajo Especial de Grado el cual está estructurado en cuatro capítulos:

**Capítulo 1 Problema de investigación:** en este capítulo se especifica el planteamiento del problema, los objetivos, el alcance, la importancia y la justificación de este trabajo

**Capítulo 2 Marco Conceptual:** se presentan los fundamentos conceptuales que sustentan este trabajo especial de grado donde se describe la arquitectura y herramientas utilizadas para el desarrollo de la aplicación web.

**Capítulo 3 Marco Metodológico:** se describen los aspectos más relevantes de la metodología de desarrollo Programación Extrema (XP).

**Capítulo 4 Marco Aplicativo:** en el que se especifican las actividades realizadas en cada una de las iteraciones que conforman el desarrollo del sistema siguiendo la metodología seleccionada previamente.

Finalmente se presentan las conclusiones y las referencias bibliográficas.

## **CAPÍTULO I. Problema de Investigación**

### **1.1 Planteamiento del Problema**

Con el transcurso del tiempo la tecnología avanza, las empresas de seguros se sienten con la necesidad y responsabilidad de adquirir nuevas tecnologías para el mejoramiento de sus sistemas y a la vez de sus procedimientos, con el fin de dar versatilidad a la divulgación de la información y fortalecer el sistema de comunicación con sus clientes, de tal manera que la información pueda llegar de manera más rápida, y sencilla a todos sus usuarios.

Las aplicaciones web ofrecen numerosas ventajas en cualquier ámbito de trabajo donde se apliquen, ya que por medio de estas se permiten divulgar, mantener y mejorar la estructura de información que se muestra en el sitio web.

Las empresas de seguros hoy en día tienden a emitir pólizas de seguros mediante una forma presencial en la que el asegurado debe asistir a las oficinas de atención al cliente o por vía de un CallCenter o bien sea a través de un corredor de seguros.

Debido a esto nace la iniciativa de desarrollar una aplicación web que permita tramitar la adquisición de pólizas para teléfonos celulares, Smartphone y tabletas. La misma contará con un módulo de registro de usuario en el cual deberá proveer una serie de datos que serán llenados en un formulario de la aplicación, un módulo de cotización donde el usuario podrá seleccionar un modelo de dispositivo móvil y la respuesta será una cotización con el monto mensual que deberá cancelar para la adquisición de la póliza. Un módulo de pago que cuenta con una serie de opciones que le permita llevar a cabo dicho pago según sea la opción seleccionada, y un módulo de generación de documentos en formato PDF para la creación de pólizas que serán enviadas a los usuarios digitalmente.

A través del desarrollo de este TEG se desarrolló una aplicación web para mejorar la calidad del servicio ofrecido por las empresas de seguros específicamente para asegurar teléfonos celulares, Smartphone y tabletas.

## **1.2 Objetivos**

### **1.2.1 General**

El objetivo general de este TEG es:

Desarrollar una aplicación Web para la cotización, compra y manejo de pólizas de seguros de celulares, Smartphone y tabletas en Venezuela para lograr mejoras en el proceso de negocio y en la generación de documentos asociados al proceso.

### **1.2.2 Específicos**

A continuación se muestra una lista de los objetivos específicos de este TEG:

- Analizar los requerimientos del usuario de la aplicación web a desarrollar.
- Diseñar interfaz gráfica del módulo de cotización, registro y consulta de pólizas y módulos de administración interna que facilite el proceso de contratación de pólizas para celulares, Smartphone y tabletas.
- Diseñar una base de datos que brinde soporte a los módulos de cotización, registro y seguimiento de pólizas, y la base de datos donde se guardará la información de los logs del sistema.
- Desarrollar los módulos administrativos, de ventas y atención al cliente del sistema.
- Diseñar y aplicar pruebas de aceptación para cada módulo desarrollado en la aplicación para comprobar su funcionamiento.

## **1.3 Alcance**

El alcance de esta investigación abarca los siguientes puntos:

- Desarrollo de los módulos de cotización, registro y consulta de pólizas y de administración.
- Establecimiento de plantillas de correos que permitan su reutilización dentro del sistema.
- Generación de contratos de pólizas en formato PDF por parte del administrador.
- El proyecto se enfocó en Pólizas de seguros de celulares, Smartphone y tabletas.

#### **1.4 Importancia y justificación**

A nivel nacional las empresas de seguros no cuentan con una aplicación web que le permita a sus clientes la fácil ejecución de procesos tan básicos como lo es el adquirir una póliza de seguro, por el contrario todo se efectúa de manera manual y presencial, siendo una gran desventaja para las empresas no aprovechar los recursos tecnológicos que existen hoy en día. Por tal razón nace la iniciativa de crear una Aplicación Web que permita automatizar el proceso de adquisición de pólizas específicamente para teléfonos, Smartphone y tabletas; logrando que los clientes puedan realizar la adquisición de pólizas vía internet.

Se considera sumamente importante el mejorar esos procesos para que de esa manera se logre un ahorro de tiempo, se disminuye el esfuerzo humano y se minimizan los errores que puedan existir durante la adquisición de una póliza en un proceso manual, también es importante mencionar que de esta manera se agilizan tareas administrativas y de carga laboral al personal, además que se puede acceder desde cualquier lugar donde exista una conexión y acceso a internet logrando abarcar un mayor mercado de clientes sin depender de tener una oficina o sucursal en diferentes estados del país.

## **CAPÍTULO II. Marco Conceptual**

### **2.1 Póliza de Seguros**

La póliza es un documento que representa al Contrato de Seguros, y es el que refleja tanto las normas que lo regulan de forma general, particular o especial como la relación convenida entre el Asegurado y el Asegurador. (San Antonio, 1991).

### **2.2 Elementos relacionados con una Póliza de seguros**

#### **2.2.1 Compañía de seguros**

Es la institución económica que elimina o reduce los perjuicios que en el patrimonio de una persona producen determinados acontecimientos fortuitos, distribuyendo aquellos perjuicios sobre una serie de personas en las cuales gravita el mismo riesgo, aunque no se haya cumplido. (San Antonio, 1991).

#### **2.2.2 Accidente**

Es el acontecimiento inesperado, repentino e involuntario que puede ocasionar daños a personas (reportando consecuencias económicas o personales) o cosas independientemente de su voluntad. (San Antonio, 1991).

#### **2.2.3 Asegurado**

Es el titular del interés asegurado, o la persona que está expuesta al riesgo de pérdida sobre ella misma, sus bienes o intereses económicos. Persona que mediante el pago de una prima traspasa el riesgo que recae sobre ella a una compañía aseguradora. Cuando contrata el seguro pasa a tener la doble calidad de Asegurado o Tomador (San Antonio, 1991).

#### **2.2.4 Asegurador**

Es la empresa que asume la cobertura del riesgo, previamente autorizada a operar como tal por la Superintendencia de Seguros de la Nación, la empresa (sociedad anónima especial) constituida en conformidad a la ley, que asume el riesgo que terceras personas le traspasan, a cambio de una prima. (San Antonio, 1991).

#### **2.2.5 Aviso de Siniestro**

Comunicación efectuada por el Asegurado, donde le informa al asegurador, la ocurrencia de determinado accidente, cuyas características guardan relación, en principio, con las circunstancias previstas en la póliza para que se efectúe la indemnización. (San Antonio, 1991).

### **2.2.6 Prima de seguro**

Aportación económica que ha de satisfacer el contratante o asegurado a la entidad aseguradora en concepto de contraprestación por la cobertura de riesgo que este le ofrece. Desde un punto de vista jurídico, es el elemento real más importante del contrato de seguro, porque su naturaleza, constitución y finalidad lo hacen ser esencial y típico de dicho contrato. (San Antonio, 1991).

### **2.3 Aplicaciones Web**

Un conjunto de herramientas de software o un programa informático, que permite al usuario procesar una tarea en particular, dicha aplicación se encuentra alojada en un servidor web y es accedida a través de Internet mediante un navegador web (Internet Explorer, Opera, Mozilla Firefox, Google Chrome, Safari y otros), o una Intranet (red entre computadores que sirve para compartir recursos y tiene uso exclusivo dentro del sitio donde fue creada). (Alegsa, 2010).

De acuerdo a los aportes de Diccionario de Informática (2012), (Enubes, 2013) y (Pereda, 2007) las características principales de una aplicación web son:

- Reside en un Servidor Web.
- Se puede acceder a ella desde cualquier lugar y en cualquier momento siempre y cuando se disponga de conexión a Internet.
- Es escrita en lenguajes soportados por navegadores web, pues son ellos quienes la ejecutan independientemente del sistema operativo en el que se encuentre.
- Puede ser accedida por miles de usuarios a la vez. En el caso de que llegase a actualizarse la aplicación, todos los usuarios estarán al tanto de esta actualización de manera inmediata.
- No requiere de la instalación de programas adicionales. Para su acceso, simplemente se requiere hacer uso de un navegador web y conexión a Internet como se comentó anteriormente.
- Es Portable, es decir que es capaz de ejecutarse sin importar la plataforma en la que se encuentre.
- Es usable, una aplicación web es sencilla y entendible para todos los usuarios.

### 2.3.1 Modelo Cliente-Servidor

Esta arquitectura cliente/servidor se basa en el procesamiento compartido de la información por medio de un conjunto de procesadores, en el cual múltiples clientes, distribuidos en cualquier parte solicitan requerimientos a uno o más servidores centrales por medio de una red amplia como internet o compartida como intranet. Existen 3 elementos fundamentales sobre los cuales se desarrollan e implementan los sistemas cliente/servidor: **(Alegsa, 2010)**.

- El proceso cliente que es quien inicia el dialogo.
- El proceso servidor que es quien espera pasivamente las peticiones de servicio hechas por los clientes.
- El sistema intermedio que corresponde a la interfaz gráfica, que provee la conectividad entre el cliente y el servidor para poder intercambiar mensajes.

En la siguiente figura 1 Arquitectura Cliente-Servidor. Se observa el proceso de una petición básica de un cliente a un servidor.

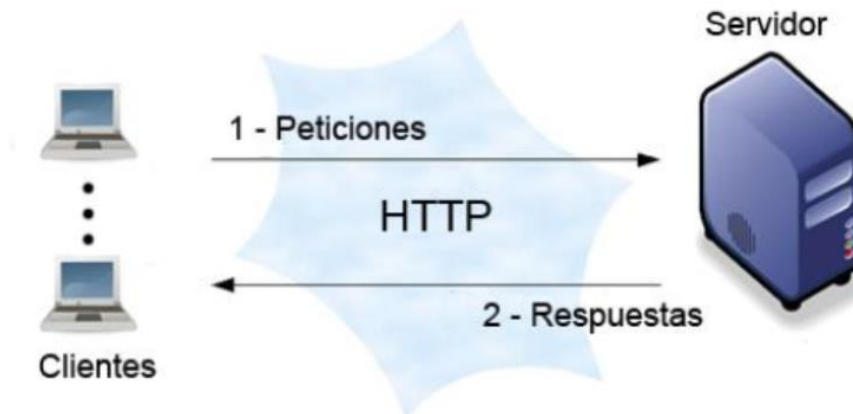


Figura 1. Arquitectura Cliente - Servidor

La Arquitectura cliente/servidor proporciona un intercambio de información entre clientes y servidores obteniendo ciertas ventajas, destacándose:

Centralización del Control: La información, los recursos y la integridad de los datos a solicitar, ubicados en el servidor, son controlados de forma que un cliente externo o defectuoso pueda dañar el sistema.

Interfaz gráfica que resalte en cuanto a la manipulación de la data.

Topología Distribuida: Es caracterizada por lograr satisfacer a clientes ubicados en cualquier parte (Tanenbaum, 2002).

1. Servidor web: Es un almacén donde se alojan las aplicaciones web y programas informáticos escritos en lenguajes que son reconocidos, ejecutados e interpretados por él, genera código HTML y lo envía al cliente. Su función es atender las peticiones del cliente web y responderlas en un formato comprensible por él (código HTML).
2. Cliente web: Un cliente web es conocido generalmente como el navegador que se utiliza en el equipo del usuario. Se encarga de hacer peticiones al servidor web y una vez recibida la respuesta de éste realiza la interpretación y ejecución de las aplicaciones web escritas fundamentalmente en Lenguaje de Marcado Hipertextual o HTML, por sus siglas en inglés HyperTextMarkupLanguage.

### 2.3.2 Patrón de arquitectura Modelo-Vista-Controlador

Modelo Vista Controlador (MCV) es un patrón de diseño, comúnmente utilizado en aplicaciones web, que separa los datos de la aplicación, la interfaz de usuario y la lógica de control en tres (3) componentes distintos los cuales pueden apreciarse en la Figura 2. (Alegsa, 2010).

Las actividades asociadas a los componentes se describen a continuación:

- **Modelo:** Define las reglas del negocio (funcionalidad de la aplicación) y representa las estructuras de los datos.
- **Controlador:** Gestiona las acciones del usuario actuando como intermediario entre las vistas y el modelo.
- **Vistas:** es la información presentada al usuario del sistema.

A continuación se presenta una secuencia de tareas del patrón Modelo-Vista-Controlador las cuales se combinan para ofrecer una respuesta satisfactoria y coherente con respecto a la acción realizada por el usuario de la aplicación:

1. El usuario interactúa con la interfaz e introduce un evento cualquiera.
2. El controlador recibe y gestiona el evento introducido por el usuario.
3. El controlador se comunica con el modelo para realizar la tarea solicitada por el usuario. En algunas situaciones se deben modificar los datos del modelo o simplemente actualizarlos.

4. Después de recibir la respuesta del modelo el controlador la entrega a la vista para que sea mostrada al usuario. En algunos casos el modelo se comunica con la vista para entregarle los resultados de la solicitud.
5. La vista espera nuevas interacciones por parte del usuario, si éstas ocurren se repite el ciclo nuevamente.

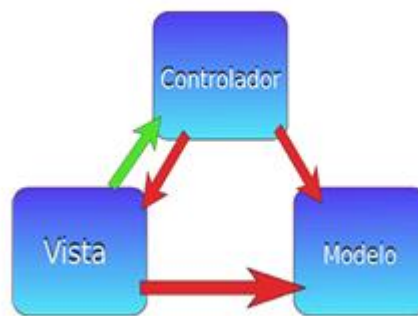


Figura 2. Modelo Vista Controlador

## 2.4 Tecnologías utilizadas para el desarrollo

Las tecnologías web son aquellas que proporcionan al desarrollador un conjunto de herramientas que facilitan la elaboración de aplicaciones web. Se dividen en tecnologías del lado: del cliente (lenguajes digeridos por el navegador web sin ningún tratamiento previo) y del servidor (lenguajes reconocidos, ejecutados e interpretados del lado del servidor y enviados al cliente en un formato comprensible para él). Es importante mencionar que tanto los lenguajes del cliente como los del servidor son independientes.

### 2.4.1 Tecnologías del lado del cliente

#### 2.4.1.1 Lenguaje de Marcado de Hipertexto (HTML)

Según (Alvarez, Desarrollo Web, 2010) y (W3C, 2013) el Lenguaje de Marcado Hipertextual o HTML, por sus siglas en inglés HyperTextMarkupLanguage: lenguaje de marcas de hipertexto es el lenguaje que permite la generación de hipertextos en la World Wide Web. Utilizado del lado del cliente se usa para la creación de páginas web. Es sencillo, de etiquetas que indican al navegador web la forma y estructura (presentación) que tendrá el texto, las imágenes y los videos colocados en la página web.

A diferencia de HTML se encuentra XHTML, acrónimo inglés de eXtensibleHyper Text MarkupLanguage (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. XHTML es la versión XML de HTML, por lo que tiene básicamente, las mismas funcionalidades, pero cumple las especificaciones más estrictas de XML. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información, y la forma de presentarla estén claramente separadas a diferencia de HTML.

#### **2.4.1.2 Hoja de estilos en cascada**

De acuerdo al aporte de Chesco (2007) y W3C (2013) se concluye que la Hoja de Estilo en Cascada o CSS, por sus siglas en inglés Cascading Style Sheet: es un lenguaje de programación utilizado del lado del cliente que permite controlar la estructura y forma (presentación) que van a tener los documentos escritos en HTML o XHTML (páginas web) de una manera más fácil, limpia y sencilla. Por lo tanto, permite al desarrollador crear la presentación de sus páginas de una forma mucho más eficiente y sin ensuciar el código HTML.

Es una tecnología que permite crear páginas web de una manera más exacta, pudiendo incorporar colores, imágenes, formato de texto, fondos, entre otros a los documentos web escritos en HTML y XML, así también, es posible definir estilos en un archivo externo a las páginas y de querer cambiar alguno de ellos, automáticamente se actualizarían todas las páginas vinculadas de nuestro sitio (**Alvarez, 2008**).

#### **2.4.1.3 AJAX**

AJAX (JavaScript Asíncrono y XML), no es una tecnología independiente, sino es la técnica que detalla el desarrollo web para aplicaciones interactivas. Estas aplicaciones se ejecutan del lado del cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano (Nerfites, 2009). Entre estas están:

- *HTML* o *XHTML* + *CSS* para crear una aplicación basada en estándares.
- *DOM* para la creación y manipulación dinámica de la aplicación.
- *XML* y *JSON* para el intercambio y la manipulación de la información que será mostrada en la aplicación.
- Objeto *XMLHttpRequest* para el intercambio asíncrono de información.
- *JavaScript* para combinar todas las tecnologías anteriores.

Esto hace posible la creación de aplicaciones muchos más rápidas y con mejor respuesta a las acciones del usuario, ya que cada acción ejecutada por el usuario genera una petición al servidor; una vez procesada esta, se crea una nueva página para el usuario. *Ajax* posibilita que el usuario nunca se encuentre frente a una ventana del navegador vacía esperando una respuesta tardía del servidor.

#### **2.4.1.4 JavaScript**

El programa en *JavaScript* tiene la particularidad de que está insertado dentro del mismo documento *HTML*, que lo presenta al usuario y no es por ello un programa aparte. Permite crear aplicaciones similares a las *CGI* (*Common Gateway Interface*). El *CGI* es un mecanismo que se ha utilizado en los servidores web para implementar las páginas web activas. El funcionamiento de los *CGI* es el siguiente: lee los datos provenientes de un formulario de una página web, procesa la información y lo escribe sobre el canal de salida estándar que es la pantalla del ordenador (Alvarez, 2009).

El programa en *JavaScript* reconoce eventos, que no son más que acciones de *JavaScript* (uno de los eventos es *onClick*, que se refiere, que al momento de pulsar con el cursor sobre la zona a la cual está asociada ese evento realice una determinada acción), creados por el usuario, definiendo así un sistema interactivo. Se puede por ello crear formularios que verifiquen la validez de la información e interpreten está en el mismo programa contenido en el documento *HTML*, sin necesidad de comunicación por la red. También se permite por medio de un código *JavaScript* realizar acciones particulares como ejecutar un archivo de audio, ejecutar una ventana, entre otros.

El lenguaje *HTML* permite dotar las páginas web de una atractiva información visual, no obstante le falta cierto grado de interactividad para el usuario. Para llenar el vacío existente entre la sencillez y la poca interactividad de *HTML*, y el grado de interactividad de *Java*, las empresas de software se dispusieron a desarrollar órdenes fáciles de utilizar. De esta manera *Netscape Communications* introdujo el lenguaje *LiveScript*, poco después la empresa *Sun Microsystems* (creadora del lenguaje *Java*), se unió a *Netscape* para conseguir que el *LiveScript* fuese adoptado como el lenguaje estándar de internet para la escritura de órdenes acerca de la web. Puesto que el *LiveScript* tenía muchas semejanzas con *Java*, el lenguaje fue renombrado *JavaScript*.

*JavaScript* es un lenguaje compacto, y basado en objeto, diseñado para el desarrollo de aplicaciones cliente servidor a través de internet. En una aplicación cliente para un navegador, la sentencia *JavaScript* contenida en un documento *HTML* puede reconocer

y responder a eventos generados por el usuario como *clic* del mouse, información de formularios y navegación de documento a documento.

Por ejemplo se puede escribir una función *JavaScript* que verifique que la información introducida por el usuario sea correcta. Un documento *HTML* con *JavaScript* contenido es capaz de interpretar la información insertada por el usuario, verificar que sea correcta y alertar al usuario de que no lo sea. *JavaScript* es un lenguaje de programación dirigido a los programadores de páginas web.

*JavaScript* fue diseñado para ser un lenguaje de elaboración de *scripts* que pudieran incrustarse en archivos *HTML*. No es compilado si no que a través de ello es interpretado y leído por el navegador como código fuente.

Es necesario resaltar que hay dos tipos de *JavaScript*: por un lado está el que se ejecuta en el cliente, este es el *JavaScript* propiamente dicho, aunque técnicamente se denomina *Navigator JavaScript*. Pero también existe un *JavaScript* que se ejecuta en el servidor, y se denomina *SS JavaScript (Server - Side)*, que permite a los *scripts* ejecutarse del lado del servidor para ser mezclados directamente con el contenido web. Cada vez que un documento contiene código *JavaScript* a ser ejecutado del lado del servidor y es solicitado por el cliente, el servidor ejecuta el *script* o *scripts* del documento, y devuelve el documento resultante (este puede generarse parcialmente estático y parcialmente dinámico). En la figura 3 se muestra un gráfico de la interacción de *JavaScript*.

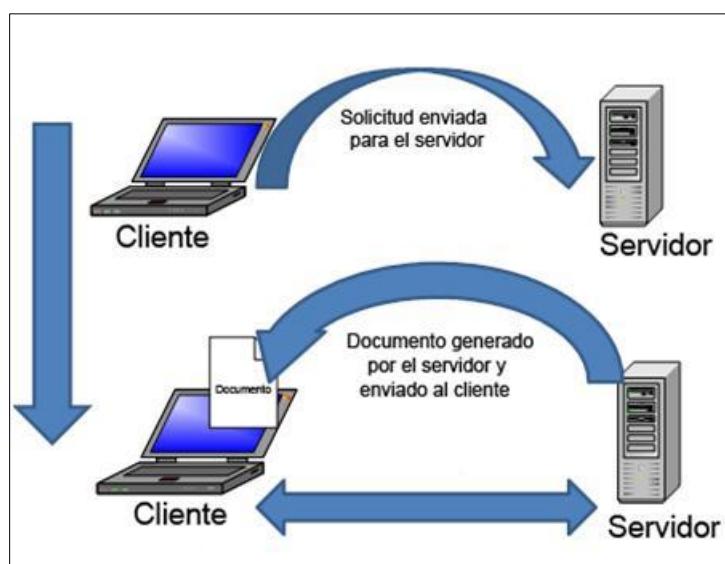


Figura 3. JavaScript. Autor: Álvarez - 2009.

#### **2.4.1.5 JQuery**

*JQuery* es una librería de *JavaScript* que permite simplificar la manera de poder interactuar con los documentos en *HTML*. Entre muchas de sus funciones, permite agregar interacción con la tecnología *AJAX* en páginas web y blogs, crear y desarrollar animaciones y manejar eventos, proporciona una serie de funcionalidades basadas en *JavaScript* que de otra manera requerirían de mucho más código como por ejemplo: menú desplegables utilizando listas, *drag and drop*, efectos de los métodos *SHOW* y *HIDE*, búsquedas dinámicas en la base de datos, entre otros.

Con las funciones propias de *JQUERY* se logran grandes resultados en menos tiempo y escribiendo menos (Alvarez, Desarrollo web, 2009).

#### **2.4.1.6 Bootstrap**

Bootstrap, es un framework originalmente creado por Twitter, que permite crear interfaces web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Es decir, el sitio web se adapta automáticamente al tamaño de una PC, una Tablet u otro dispositivo. Esta técnica de diseño y desarrollo se conoce como "*responsive design*" o diseño adaptativo.

El beneficio de usar *responsive design* en un sitio web, es principalmente que el sitio web se adapta automáticamente al dispositivo desde donde se acceda. Lo que se usa con más frecuencia es el uso de media queries, que es un módulo de CSS3 que permite la representación de contenido para adaptarse a condiciones como la resolución de la pantalla y si trabajas las dimensiones de tu contenido en porcentajes, puedes tener una web muy fluida capaz de adaptarse a casi cualquier tamaño de forma automática.

### **2.4.2 Tecnologías del lado del servidor**

#### **2.4.2.1 Preprocesador de hipertexto (PHP)**

Es un lenguaje de programación de código abierto, interpretado de alto nivel, se utiliza del lado del servidor, lo que significa que el código es ejecutado del lado del servidor, generando código HTML y enviándolo al cliente, está diseñado originalmente para la creación de páginas web de contenido dinámico de una manera rápida y sencilla.

Soporta la mayoría de servidores web usados en la actualidad, el de uso común hoy día es Apache; y puede ser utilizado en cualquiera de los principales sistemas operativos como UNIX (Linux, Solaris, Debian, Ubuntu.) y Microsoft Windows, entre otros, sin costo alguno.

Cuando un cliente hace una petición al servidor para que le envíe una página web, sucede lo siguiente del lado del servidor:

1. El servidor web recibe el mensaje, comprueba que se trata de una petición válida, y al ver que la extensión es "php" solicita al intérprete de PHP (que es otro programa que se ejecuta en el servidor web) que le envíe el archivo.
2. Una vez el intérprete PHP termina de ejecutar el código contenido en el archivo y ha recibido toda la información necesaria del gestor de base de datos, envía los resultados al servidor web.
3. El servidor web envía la página al cliente que la había solicitado y el navegador muestra en pantalla la información obtenida del servidor web.

Este lenguaje de programación orientado a objetos, a procedimientos o incluso una mezcla de ambos, puede estar incluido en el código ".html" o puede ser un archivo externo que debe ser guardado con extensión ".php".

#### **2.4.2.2 CakePHP**

Es un framework open-source orientado al desarrollo rápido y mantenimiento de aplicaciones Web sobre el lenguaje PHP. Este framework utiliza patrones de diseños habituales como MVC (Mode-View-Controller) y ORM (Object-relationalmapping), que permiten reducir los costos de desarrollo.

#### **Características del Framework:**

A continuación se listan las características del framework y se destacan algunas técnicas que se pueden aplicar para desarrollar aplicaciones incorporaran con el framework (Scaffolding, Dispathcher, helpers):

- Licencia Flexible.
- Compatible con PHP4 y PHP5.
- CRUD integrado para la interacción con la Base de Datos.
- Soporte de Aplicaciones [Scaffolding]
- Generación de Códigos.

- Arquitectura Modelo Vista Controlador (MVC).
- Despachador de peticiones [Dispatcher], con URLs y rutas personalizadas y limpias.
- Validación integrada.
- Plantillas rápidas y flexibles (Sintaxis de PHP, con ayudantes [helpers]).
- Ayudantes para AJAX, JavaScript, formularios HTML y más.
- Componentes de Email, Cookies, Seguridad, Sesión, y Manejo de Solicitudes.
- Listas de control de acceso flexibles.
- Limpieza de datos.
- Caché flexible.
- Localización.
- Funciona en cualquier subdirectorío del sitio Web, con poca o ninguna configuración de Apache.

Los archivos y carpetas que se generan a la hora de la instalación:

- App: es donde se hará la mayor parte del desarrollo de su aplicación. Se muestra un poco más de cerca las carpetas dentro de App:
  - Config: mantiene los (pocos) archivos de configuración que CakePHP usa. Detalles de conexión a bases de datos, arranque [bootstrapping], archivos de configuración del núcleo y más deberían ser almacenados aquí.
  - Controllers: contiene los controladores de tu aplicación y sus componentes.
  - Locale: almacena archivos de cadenas para la internacionalización.
  - Models: contiene los modelos de su aplicación, comportamientos [behaviors], y orígenes de datos [datasources].
  - Plugins: contiene los paquetes de plugins diseñados para cakephp.
  - Tmp: aquí es donde CakePHP almacena información que actualmente se almacena depende en cómo haya configurado a CakePHP, pero esta carpeta es usualmente usada para almacenar descripciones de modelos, registros (logs), y algunas veces información de sesiones.
  - Vendors: cualquier grupo de clases o librerías de terceros debería ser ubicado aquí.
  - Views: los archivos de presentación son ubicados aquí: elementos [elements], páginas de error, ayudantes [helpers], layouts y archivos de vistas.

- Webroots: en una configuración de producción, esta carpeta debería servir como la raíz del sitio [documentroot] para su aplicación. Las carpetas aquí también sirven como lugares de almacenamiento para hojas de estilos en cascada [CSS stylesheets], imágenes y archivos JavaScript.
- Cake: Este directorio es del núcleo del framework cakephp.
- Htaccess: Es usado para modificar o agregar funciones a directorios, estableciendo un control de acceso
- Index.php: Archivo por defecto en la ruta inicial y carpetas de la aplicación, es una de las páginas de inicio de la aplicación web.
- Readme.

#### **2.4.2.3 Ruby**

Ruby es un lenguaje de programación de alto nivel, reflexivo y orientado a objetos, donde su sintaxis viene inspirada en lenguajes como Python y Perl. Ruby es un lenguaje de programación interpretado en una sola pasada y su implementación oficial es distribuida bajo una licencia de software libre (Mazt, 2003).

En Ruby, todo es un objeto. Se le puede asignar propiedades y acciones a toda información y código. La orientación a objetos es un paradigma de programación que facilita la creación de software de calidad por sus factores que potencian el mantenimiento, la extensión y la reutilización del software generado bajo este paradigma.

La programación orientada a objetos trata de amoldarse al modo de pensar del hombre y no al de la máquina. Esto es posible gracias a la forma racional con la que se manejan las abstracciones que representan las entidades del dominio del problema, y a propiedades como la jerarquía o el encapsulamiento.

En muchos lenguajes, los números y otros tipos primitivos no son objetos. Ruby tiene mucho en común del lenguaje Smalltalk<sup>1</sup> pudiendo poner métodos y variables de instancia a todos sus tipos de datos. Esto facilita el uso de Ruby, porque las reglas que se aplican a los objetos son aplicables a todo Ruby.

Ruby es considerado un lenguaje flexible, ya que permite a sus usuarios alterarlo libremente.

Las partes esenciales de Ruby pueden ser quitadas o redefinidas a placer. Se puede agregar funcionalidad a partes ya existentes. Ruby intenta no restringir al desarrollador.

#### **2.4.2.4 RubyOnRails (RoR)**

Rails es un framework para el desarrollo de aplicaciones web, software libre por naturaleza, está basado en el patrón de diseño Modelo Vista Controlador (MVC), presenta una gran ventaja cuando se desarrolla un software para distintos dispositivos, ya que los cambios requeridos son mínimos y sencillos. Trata de combinar la simplicidad con la posibilidad de desarrollar aplicaciones del mundo real escribiendo menos código que con otros entornos de trabajo y con mínimas configuraciones (Santos, 2006).

Uno de los principales objetivos de RoR es la reutilización de código para no perder tiempo programando algo que ya está creado. Además, el framework también está pensado para que trabajar con bases de datos SQL sea de lo más sencillo.

Sin embargo, es importante conocer las convenciones de programación que existen en Rails por ejemplo: el nombre de las tablas de las bases de datos. Rails recomienda nombrar las tablas con el nombre de la clase que la representa en plural y en minúsculas (la clase 'Forum' si tuviera una tabla se debería llamar 'forums'). Como éste hay varios convenios más que a simple vista parecen poco productivos, pero luego permiten ahorrar una cantidad de tiempo impresionante cuando luego se quiere hacer la documentación o editar código, etc.

RoR tiene algunas filosofías que es recomendable mencionar, como son:

- No lo repitas (del inglés Don't repeat yourself, DRY).
- Convención sobre configuración.
- Uso de patrones de diseño.
- Generación de código (HELPERS).
- Menos código, menos errores.

- Tests integrados (Unitarios y Funcionales).

Para desarrollo y pruebas, se utiliza Mongrel o WEBrick, incluido con Ruby. Para utilizar Rails en servidores en producción se está extendiendo el uso de Passenger, una suerte de mod\_rails para Apache desarrollado en 2008 por la empresa holandesa Phusion. Otras opciones para producción son Nginx, Mongrel, Apache, lighttpd con FastCGI o alguna combinación de ambos (por ejemplo utilizando Apache como proxy para los procesos Mongrel). Sobre Apache, mod\_ruby puede mejorar considerablemente el rendimiento, aunque su uso no se recomienda porque no es seguro utilizar múltiples aplicaciones RoR sobre Apache.

#### **2.4.2.5 Active\_admin**

Es una librería de Ruby que permite la generación de una interfaz anexa a la de la aplicación, teniendo como objetivo poder administrar todo el contenido de manera más dinámica y eficaz, además, brindando apoyo del manejo de la información. Tiene como objetivo hacer que las tareas por parte del administrador sean fáciles y seguras de implementar.

Dentro de las funcionalidades más resaltantes que ofrece Active\_admin se tienen:

- Se establecen filtros de búsqueda por cada campo de cada tabla de la base de datos.
- Implementación de una función de encriptación MD5 para claves de acceso al sistema.
- Permite la descarga en formatos CVS, XML Y JSON del contenido de una vista desplegada en tablas o bien sea de una tabla en particular de la base de datos.
- Es posible implementar botones de acciones que permitan la ejecución de una acción o función que facilite las funcionalidades del sistema.
- Fácil distribución del contenido y manejo de la información.
- Aporta una vista global, denominada "Inicio", donde permite la visualización de temas o datos relevantes que se necesiten mostrar al ingresar al sistema.

#### **2.4.2.6 Sistema Manejador de Base de Datos**

Un Sistema Manejador de Base de Datos es un software que tiene como propósito general facilitar el proceso de definir, construir y manipular bases de datos que se utilizan para diferentes tipos de aplicaciones (Rodríguez, 2006).

Entre sus principales procesos sobresalen:

- Definir la base de datos significa la declaración de:
  - Los tipos de datos
  - La estructura
  - Las restricciones de los datos a ser almacenados en la base de datos.
- Crear o construir la base de datos: es el proceso de almacenar los datos en algún medio de almacenamiento, esto es controlado por el Sistema Manejador de Base de Datos (SMBD).
- Manipular una base de datos incluye funciones como:
  - Consultar la base de datos para obtener algunos datos específicos.
  - Actualizar la base de datos para reflejar cambios.
  - Generar reportes de los datos.
  - Eliminar algunos datos.

##### **2.4.2.6.1 MySQL**

De acuerdo a (MySQL, 2013) se define MySQL como un sistema de administración de bases de datos relacional (SMBDR), que almacena y distribuye una gran cantidad de datos, típicos de una aplicación.

Está basado en la arquitectura cliente-servidor, por lo que el servidor de BD puede estar asociado a múltiples clientes. Utiliza SQL para el acceso y manipulación de los datos.

MySQL es software de fuente abierta, lo que significa que es posible para cualquier persona usarlo y modificarlo. Cualquier persona puede bajar el código fuente y usarlo sin pagar, o estudiarlo y ajustarlo a sus necesidades. Este SMBD usa la Licencia Pública General (GPL, del inglés General Public License) GNU para definir qué puede hacer y qué no puede hacer con el software en diferentes situaciones. Si usted no se ajusta al

GPL o requiere introducir código del estilo de este manejador en aplicaciones comerciales entonces puede comprar una versión comercial licenciada.

Según (MySQL, 2013) sus características son:

- Velocidad de acceso: está basado en una arquitectura multihilos mediante hilos del kernel, por lo que diversos clientes pueden acceder simultáneamente de forma concurrente (multiusuarios).
- Velocidad de respuestas: MySQL implementa un almacenamiento parcial de los resultados de las consultas comunes (caché), mediante tablas hash en memorias temporales, por lo que una vez que se ejecuta una consulta, su resultado es almacenado de forma temporal, por si se requieren esos datos nuevamente, aumentando la velocidad de respuesta. Adicionalmente, usa tablas en disco en forma de árboles binarios (b-tree) para búsquedas rápidas con compresión de índice, por los que los tiempos de búsqueda son muy bajos.
- Fácil uso: provee una interfaz de línea de comandos. Además cuenta con muchas herramientas gráficas para su manejo, desde aplicaciones de escritorio hasta aplicaciones web.
- Puede ejecutarse en diversos sistemas operativos sin ningún problema, mediante las herramientas GNU Automake, Autoconf, y Libtool.
- Internacionalización: brinda soporte para varios idiomas y diversos juegos de caracteres.
- Código abierto: cualquier desarrollador tiene acceso al código libremente.
- Soporte: el código MySQL se prueba con Purify (un detector de memoria perdida comercial) así como con Valgrind, una herramienta GPL. Tiene soporte completo para operadores y funciones en cláusulas Select y Where, además de un completo soporte para cláusulas groupby y orderby, soporte de funciones de agrupación.
- Seguridad: posee un buen enfoque en cuanto a seguridad, ya que ofrece un sistema de contraseñas y privilegios seguro mediante verificación basada en el host y el tráfico de contraseñas está cifrado al conectarse a un servidor. Soporta gran cantidad de datos. MySQL Server tiene BD de hasta cincuenta (50) millones de registros.

Los clientes se conectan al servidor MySQL usando sockets TCP/IP en cualquier plataforma. En sistemas Windows se pueden conectar usando named pipes y en sistemas Unix usando ficheros socket Unix.

#### 2.4.2.7 Manejo de Versiones

Se llama manejo de versiones a la gestión de los diversos cambios que se realizan sobre los elementos de algún producto o una configuración del mismo. Una versión, revisión o edición de un producto, es el estado en el que se encuentra dicho producto en un momento dado de su desarrollo o modificación. Aunque un sistema de control de versiones puede realizarse de forma manual, es muy aconsejable disponer de herramientas que faciliten esta gestión dando lugar a los llamados sistemas de control de versiones o SVC (del inglés System Versión Control) (Versión Control with Subversión. (O'Reilly, 2004).

Para este trabajo de investigación se utilizó el siguiente:

- ***GIT***

*Git* es un sistema de control de versiones diseñado para manejar proyectos muy grandes con velocidad y eficiencia, pero igual de apropiado para repositorios pequeños; es especialmente popular con la comunidad *Open Source*, sirviendo como plataforma de desarrollo para proyectos como el Kernel Linux, *Ruby on Rails*, WINE o X.org.

*Git* cae en la categoría de herramientas de manejo de código fuente distribuido, similar por ejemplo a *Mercurial* o *Bazaar*. Cada directorio de trabajo de *Git* es un repositorio completo con historial y capacidades totales de tracking de revisiones, independiente de acceso de red o un servidor central. Aun así, *Git* es extremadamente rápido y eficiente con el espacio.

*Git* es un proyecto *Open Source* cubierto por la *GNU General Public License v2*. Originalmente escrito por *Linus Torvalds* y mantenido por (Hamano, 2008).

En la figura 4 se muestra la arquitectura de la aplicación.

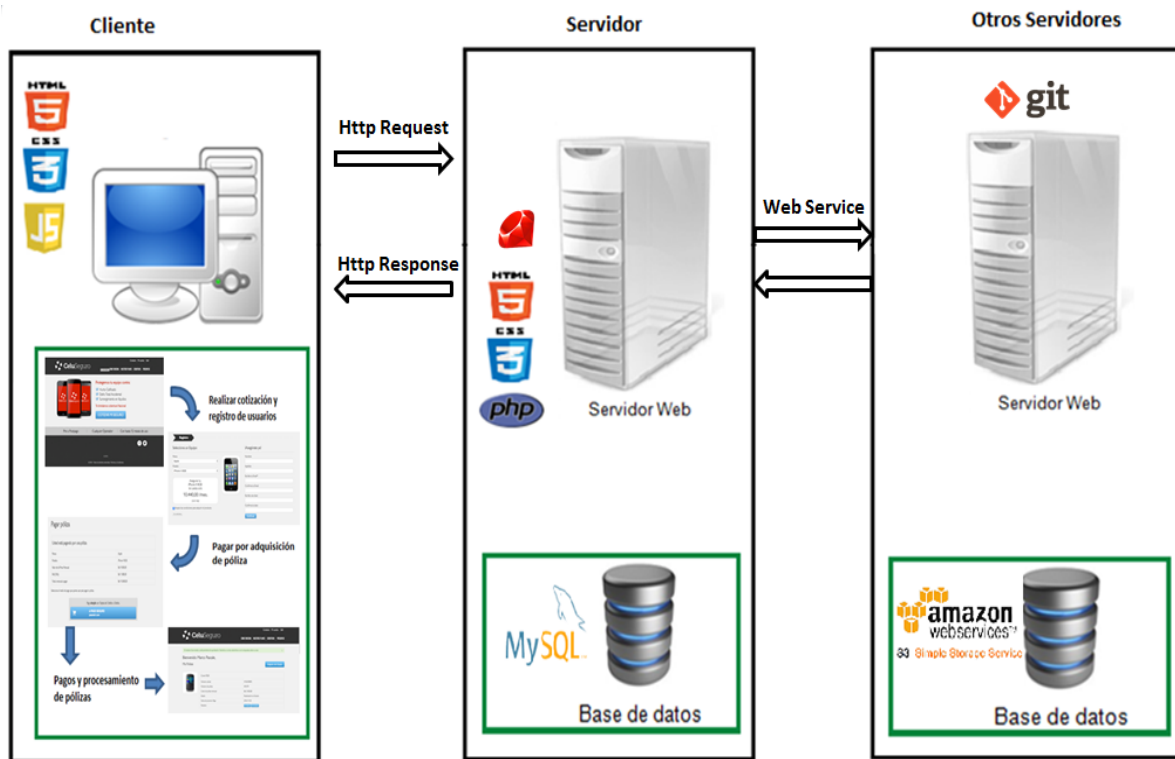


Figura 4. Arquitectura de la aplicación

### **CAPÍTULO III. Marco Metodológico**

En el presente capítulo se describe la metodología usada para el proceso de desarrollo de la Aplicación web de cotización, compra y manejo de pólizas para celulares, Smartphone y tabletas; la metodología ágil "Programación Extrema" (XP), la cual se fundamenta en la simplicidad, la comunicación, la retroalimentación y la refactorización de código (Jeffries, Anderson, y Hendrickson, 2000).

A continuación se especifican los aspectos más resaltantes de la Programación Extrema (XP).

#### **3.1 Programación Extrema (Extreme programming)**

XP (eXtreme Programming) se trata de una metodología ágil en contraposición a las metodologías pesadas como RUP. Se basa en la simplicidad, la comunicación y la retroalimentación o reutilización del código desarrollado. No se enfoca en la documentación sino en los requerimientos comunicados por el cliente.

El objetivo principal que se persigue es la satisfacción del cliente [1], por eso tiene mucha importancia la comunicación con los usuarios o clientes. Esta comunicación se va a soportar principalmente en las historias de usuario (UserStories) cuando proviene desde el cliente, y de las entregas y versiones parciales del sistema cuando la comunicación es hacia el cliente (*Beck, 1997*).

##### **3.1.1 Características**

De acuerdo al aporte de (Jeffries, Anderson, y Hendrickson, 2000) junto con las contribuciones de Pressman (2007) se concluye que las principales características de XP son:

Antes de comenzar un proyecto XP se deben analizar cuatro variables: costo, tiempo, calidad y ámbito del proyecto. La metodología propone hacer un análisis en donde se pondere la importancia de cada variable, debido al impacto que cada una puede ejercer sobre las otras.

Se basa en la mejora continua del código fuente mediante prácticas de comunicación entre los actores participantes y reutilización de artefactos. Se busca mantener una calidad alta en los artefactos producidos mientras estos se generen rápidamente.

Se realiza una programación en parejas, se supone que la mayor calidad del código escrito de esta manera es más importante que la posible pérdida de productividad inmediata, manteniendo siempre una comunicación constante con el cliente.

El proceso de desarrollo es iterativo e incremental, sin introducir nuevas funcionalidades al sistema antes que estas sean necesarias.

El proceso está orientado a los desarrolladores y al usuario, los cuales interactúan entre sí para definir el alcance del proyecto y corregir errores conceptuales a la brevedad posible.

El desarrollo está estrictamente guiado por pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se recomienda altamente escribir el código de la prueba antes de la codificación.

Mediante esta metodología se libera rápidamente a producción un sistema sencillo y, de igual manera, se liberan continuamente nuevas versiones en periodos cortos. Tanto los jefes de proyecto, los clientes y programadores, son parte del equipo y están involucrados en el desarrollo del software.

Según (Kent, 2007) XP se divide principalmente en iteraciones las cuales contienen cuatro actividades (Planificación, Diseñar, Codificación y Pruebas), explicadas a continuación:

- Planificar: La actividad de planificación comienza creando una serie de historias de usuario, en las cuales se describen los requerimientos del sistema en terminología del cliente, proporcionando a su vez una estimación del tiempo necesario para el desarrollo de dicha iteración. (el tiempo para el desarrollo de cada iteración debe oscilar entre una (1) y tres (3) semanas.
- Diseñar: El diseño también debe ser incremental y debe estar reflejado en el software, lo cual quiere decir que la estructura de éste debe ser clara. Hay que diseñar lo que las necesidades del problema requieren.
- Codificación: Significa que, todo el software se produce mediante la elaboración de pequeñas versiones incrementales de producción corta. XP promueve el uso de código existente con la condición que sea refactorizado (reestructurado, alterando su estructura interna sin cambiar su comportamiento externo) antes de ser incluido en el proyecto. Cuando se desarrolla usando Extreme Programming, existe un énfasis fuerte en que el código fuente sea claro y fácil de entender, para que cualquier programador del grupo de trabajo pueda modificar y extender el código existente. Esto se hace con la intención de repartir equitativamente las tareas en el grupo y no hacer a alguna persona particular indispensable.

- Pruebas: Se debe asegurar que todo lo que se hace funcione correctamente. Para ello, lo mejor es desarrollar la o las pruebas desde el momento que se conocen las historias del usuario, sin embargo se pueden elaborar en cualquier momento del proceso de desarrollo.

El método XP tiene como premisa primordial, dividir el trabajo en iteraciones, las cuales se enfocan en versiones parciales del sistema hasta llegar mediante iteraciones al producto final. Los nuevos requerimientos son recibidos progresivamente y son incluidos en cada nueva iteración.

En la Figura 5 se muestra el ciclo de vida de XP y algunas de las ideas y tareas claves asociadas con cada actividad del marco de trabajo:

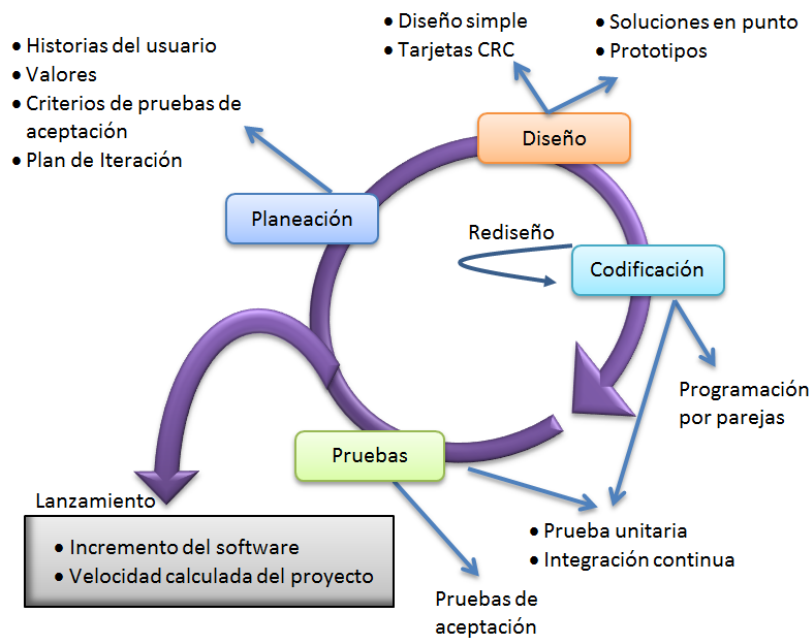


Figura 5. Ciclo de Vida de XP. Jaspe (2012)

### 3.1.2 Iteraciones

El desarrollo está basado o compuesto principalmente por iteraciones. La idea de las iteraciones es ir trabajando sobre versiones pequeñas o parciales del sistema hasta llegar al producto final. En el desarrollo de la aplicación se reciben los requerimientos y la retroalimentación progresivamente. En éste sistema en las primeras iteraciones se trata de realizar una arquitectura que pueda ser utilizada durante el resto del proyecto.

Las iteraciones pueden ser de dos tipos principalmente: por objetivos o por lapsos de tiempo. En el desarrollo de este sistema las iteraciones están basadas en lapsos de tiempo estimados a una semana. Durante el tiempo fijado para cada iteración se realizan las implementaciones indicadas en las historias de usuario que abarque.

### 3.1.3 Historias de Usuarios

Las historias de usuario son la técnica utilizada para especificar los requisitos del software, éstas tienen el propósito de describir los requerimientos de los clientes. El contenido de las historias de usuario proviene del cliente.

En cada iteración se lleva a cabo un grupo de historias, es decir, en cada iteración se trabaja sobre uno o más requerimientos, tal como se puede apreciar en la tabla 1.

Tabla 1. Iteraciones e historias de usuario

|             |                                                              |
|-------------|--------------------------------------------------------------|
| Iteración 0 | ...<br>Historia de usuario 1<br>Historia de usuario 2<br>... |
| Iteración 1 | ...<br>Historia de usuario 7<br>Historia de usuario 8<br>... |

El tratamiento de las historias de usuario es muy dinámico y flexible, en cualquier momento las historias de usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en un tiempo corto.

Cabe destacar que se tienen tres variantes de historia de usuario para los requerimientos:

- Nueva.
- Corrección.
- Mejora.

Con respecto a la información contenida en la historia de usuario, existen varias plantillas sugeridas pero no un único formato a seguir.

En tabla 2 se muestra el esquema de tabla a utilizar para las historias de usuarios debido a que hay muchas variantes se ha tomado la siguiente a utilizar en nuestro proyecto.

Tabla 2. Esquema de tabla para las historias de usuario.

| Id | Fecha | Requerimiento | Tipo |
|----|-------|---------------|------|
|    |       |               |      |

## **CAPÍTULO IV Marco Aplicativo**

### **4.1 Adaptación del proceso de desarrollo XP**

Las iteraciones simbolizan los cambios incrementales generados a través de las pruebas y retroalimentaciones repetidas que a futuro dan como resultado un sistema estable pero en evolución. Las iteraciones pueden ser de dos tipos principalmente: por objetivo o por lapsos de tiempo. En nuestro caso las iteraciones se realizan por objetivo, donde el cumplimiento del mismo, comprende la puesta en marcha de un conjunto de requerimientos funcionales (historias de usuario) para cada iteración.

A continuación se describen los aspectos relacionados a la adaptación del proceso XP que se realizó durante el desarrollo de la Aplicación Web para la gestión de pólizas de seguros para dispositivos móviles.

#### **4.1.1 Actores y Responsabilidades**

Los actores son todas las personas involucradas en el desarrollo del proyecto, los cuales a su vez cumplen distintos roles o responsabilidades según su importancia y nivel de participación. A continuación se destacan los roles existentes en el presente proceso de desarrollo para la aplicación web de gestión de pólizas:

**Cliente:** Determina la funcionalidad que se pretende alcanzar en cada iteración y define las prioridades de implementación según el valor de negocio que aporta cada historia.

**Programador:** Es responsable de implementar las historias solicitadas por el cliente. Además, estima el tiempo de desarrollo de cada historia para que el cliente pueda asignarle prioridad dentro de alguna iteración.

**Seguidor:** Su función se centra en realizar las pruebas de integración al sistema del código provisto por los programadores y de verificar el correcto funcionamiento de la aplicación.

**Administrador:** Tiene la responsabilidad de implementar, configurar, mantener, monitorizar, documentar y asegurar el correcto funcionamiento de la aplicación.

En el tabla 3 se muestra el esquema de actores y los roles que representan.

Tabla 3: Actores y roles.

|                     | <b>Cliente</b> | <b>Programador</b> | <b>Seguidor</b> | <b>Administrador</b> |
|---------------------|----------------|--------------------|-----------------|----------------------|
| Marco Pascale       |                | X                  | X               | X                    |
| Lenys Varguillas    |                | X                  | X               | X                    |
| Luis Davila         | X              |                    |                 |                      |
| Andres Figueira     | X              |                    |                 |                      |
| Concettina Di Vasta | X              |                    |                 |                      |

#### 4.1.2 Metáfora del Sistema

Metáfora es la forma de transmitir cómo funciona el sistema. Se encontraron metáforas en frases sencillas o diagramas. Las metáforas ayudan a cualquier persona a entender el objeto del programa. Es sistema a desarrollar en general es una aplicación web que va a permitir a un cliente realizar la adquisición de pólizas de seguro para un dispositivo móvil, reportar un siniestro y manejar su información de manera remota.

Principalmente se permite a los usuarios realizar actividades como: Cotización de costo de pólizas, compra de pólizas de seguro, colocación de reclamos, carga de documentos.

Adicionalmente, el Administrador puede realizar: revisión de pólizas vendidas, revisión de usuarios, configuración de variables del sistema, manejo de reclamos de siniestro, revisión de documentación enviada por el cliente, revisión de reportes del sistema, aprobación y rechazo de reclamos es decir que tiene un perfil de usuario administrador.

La metáfora de la Aplicación Web de cotización, compra y manejo de pólizas para celulares, Smartphone y tabletas se muestra en la figura 6.

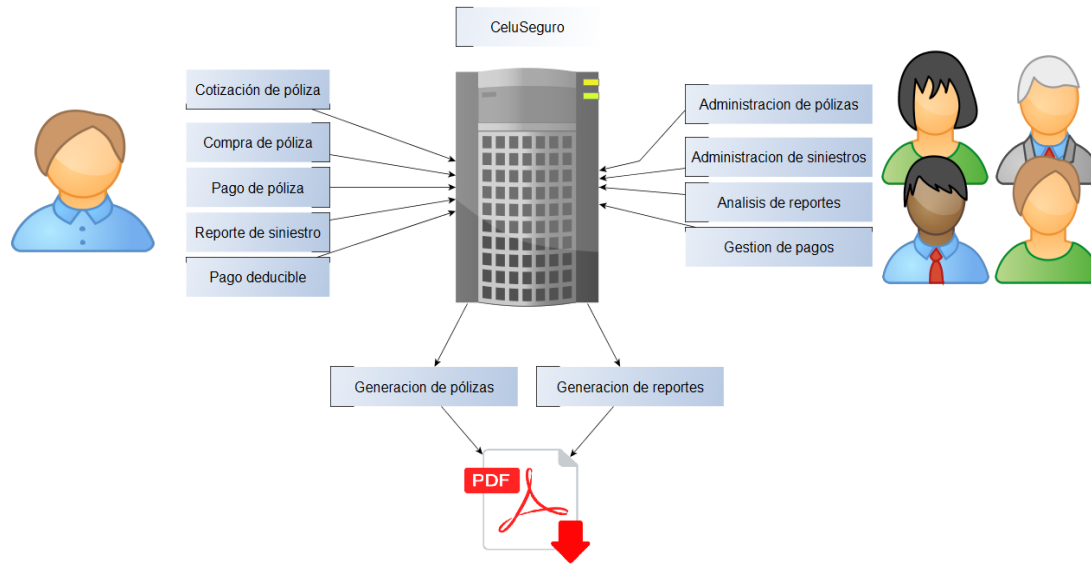


Figura 6. Metáfora de la aplicación

#### 4.1.3 Adaptación de las tareas XP

Se adaptó las 4 tareas fundamentales al desarrollo de nuestra aplicación web.

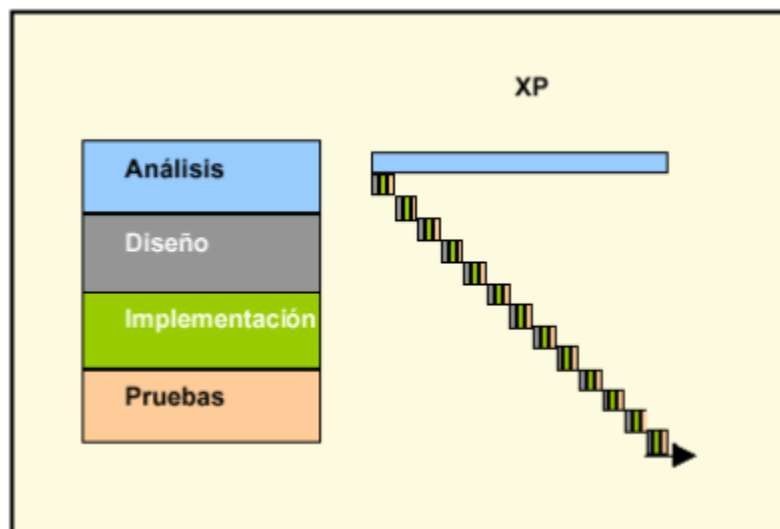


Figura 7. Desarrollo de XP

Se tienen cuatro tareas fundamentales (ver figura 7) durante las iteraciones las cuales serán adaptadas de la siguiente manera: esta explicación general debe hacerse en el

capítulo anterior, aquí va el análisis, diseño etc. con el caso de la aplicación web de ustedes

- **Análisis:** Para el análisis se utilizan las historias de usuario adaptado según ha sido explicado anteriormente.
- **Diseño:** Por cada iteración se crearon los prototipos de vistas, diagramas de clase y diagramas de base de datos. Que permitan resolver los requerimientos de la iteración.
- **Codificación o Implementación:** El código de implementación del sistema se irá registrando y sincronizando constantemente en un servidor de control de versiones (git).
- **Pruebas:** Las pruebas serán de aceptación en donde el usuario o cliente pone a prueba el sistema y verifica que hayan quedado cubiertos los requerimientos. También se realizaran pruebas unitarias y de funcionalidad.

Es posible que en algunas iteraciones no se tengan desarrolladas todas las cuatro tareas. Esto se debe a las actividades que involucran determinadas historias de usuario durante una iteración.

## **4.2 Requerimientos Generales del sistema**

La aplicación que se va a realizar es principalmente para permitir a los usuarios de la aplicación web una cotización sobre un bien como por ejemplo un teléfono celular o una Tableta.

La aplicación web debe contener un módulo que permita la cotización de celulares o Tablet seleccionando la marca y el modelo de una lista de opciones. También deberá contar con otro módulo que permita hacer un registro de usuario para el nuevo cliente. Módulo de autenticación en la aplicación para acceso al dashboard.

### **4.2.1 Requerimientos funcionales**

Entre los requerimientos funcionales más notables y generales del sistema destacaron los siguientes:

- Capacidad de almacenar documentos digitales.
- Cotizar varias marcas de dispositivos móviles.
- Generar certificado de póliza.
- Presentar las páginas con un buen grado de usabilidad.

### **4.2.2 Requerimientos no funcionales**

Los requerimientos no funcionales básicamente están determinados por la eficiencia para este tipo de sistema.

- Que funcione sobre una plataforma robusta, eficiente y escalable.
- El sistema debe tener eficiencia y velocidad de respuesta.

Estos requerimientos están cubiertos por las tecnologías referenciadas en la propuesta el principio de este trabajo.

### **4.3 Requerimientos Específicos del sistema**

- Identificar tecnologías para el procesamiento de datos.
- Diseñar interfaz gráfica del módulo de cotización, registro y consulta de pólizas.
- Diseñar interfaz gráfica del módulo de administración interna de la entidad que emite las pólizas
- Codificar módulos de cotización, administración generador de documentos.
- Codificar módulo de simulación mediante pasarelas de pagos.
- Realizar pruebas de funcionamiento.

### **4.4 Desarrollo de la aplicación**

La presente sección está dividida en iteraciones, durante las cuales se implementan las soluciones a los requerimientos.

#### **4.4.1 Iteración 0**

En la iteración 0 se definen e identifican las estructuras y los modelos que compongan el sistema.

Sobre estas estructuras se va a basar el sistema, donde se definirá el modelo de datos para los usuarios y considerando que se va a trabajar con data sensible se le aplicara un mecanismo de encriptación para proteger la misma.

##### **4.4.1.1 Planificación**

En la tabla 4 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 4. Historia de usuario para la iteracion 0.

| ID   | Fecha      | Requerimiento                                   | Tipo  |
|------|------------|-------------------------------------------------|-------|
| 1.00 | 15/02/2016 | Definir estructuras y tablas del aplicación web | Nueva |
| 2.00 | 15/02/2016 | Definir los modelos de la aplicación            | Nueva |

#### 4.4.1.2 Diseño

En la figura 8 se muestra el diagrama de tablas que conforma parte fundamental de la aplicación. En él se puede apreciar la data esencial de la aplicación.

El diagrama expresa de manera sencilla la naturaleza de la solución.

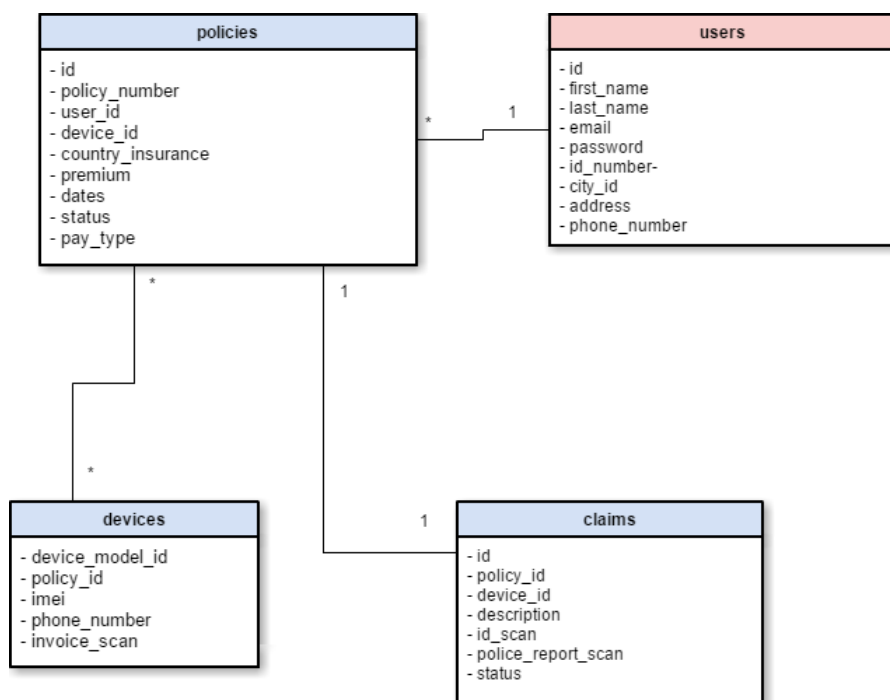


Figura 8. Visión del modelo relacional representando las tablas principales de la aplicación.

Es importante destacar que los campos de la tabla de users van a estar encriptados para mejorar la seguridad de los datos.

#### 4.4.2 Iteración 1

En esta iteración se crean las primeras versiones de las interfaces para manejo de cuentas de usuario, registro y la página del home.

Para esta iteración realizó todo lo que tiene que ver con los registros y acceso de los usuarios a una pantalla personal.

Es importante destacar que como clave primaria para la parte de acceso al sistema y del registro de usuario se usó el email del usuario.

##### 4.4.2.1 Planificación

En la tabla 5 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 5. Historia de usuario para la iteración 1.

| Id   | Fecha      | Requerimiento                                            | Tipo   |
|------|------------|----------------------------------------------------------|--------|
| 3.00 | 18/02/2016 | Hacer las interfaces de esta iteración                   | Nueva  |
| 4.00 | 18/02/2016 | Hacer el funcionamiento del registro y acceso al sistema | Nueva  |
| 5.00 | 19/02/2016 | Definir estructuras y tablas del sistema                 | Nueva  |
| 1.01 | 19/02/2016 | Estructuras del sistema                                  | Mejora |

##### 4.4.2.2 Diseño

El modelado inicial de la aplicación se visualiza a través del diagrama de clases presentado en la figura 9.

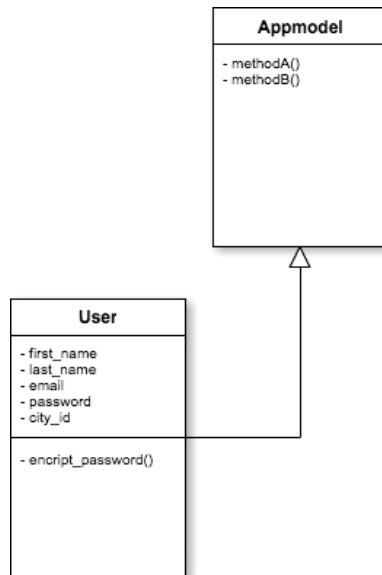


Figura 9. Modelo inicial del diagrama de clases.

Las interfaces siguen un patrón similar debido a que usan el mismo Layout, o sea todas utilizan una misma plantilla y una misma hoja de estilo.

En la figura 10 se da una muestra de una vista correspondiente a la sección de contacto de la aplicación.

The image shows a web application's contact page. At the top, there is a grey navigation bar with links: 'COMO FUNCIONA', 'COBERTURA', 'PREGUNTAS', and 'CONTACTO'. On the right side of this bar are buttons for 'Mi cuenta' and 'Salir'. Below the navigation bar, the page title 'Contáctanos' is displayed. The main content area is divided into two columns. The left column contains a contact form with fields for 'Nombre', 'Email', 'Telefono' (with a placeholder 'Para ser contactado Ejemplo: xxx-xxxxxxx'), and 'Mensaje'. A blue 'Enviar' button is at the bottom of the form. The right column contains text: 'Para preguntas relacionadas con la adquisición de pólizas ó reclamaciones, por favor llame al', followed by the phone number '555.55.55' in large blue font, and the word 'ó'. Below this is a red button that says 'Haz click AQUÍ'. Further down, it lists 'UCV' and 'Caracas - Venezuela'. At the bottom of the page is a dark grey footer. On the right side of the footer are the text 'Venezuela' with a dropdown arrow, and icons for Facebook and Twitter. In the center of the footer, it says 'Universidad Central de Venezuela', 'Facultad de Ciencias', and '© 2016 - Escuela de Computacion'.

Contáctanos

Nombre

Email

Telefono

Mensaje

Enviar

Para preguntas relacionadas con la adquisición de pólizas ó reclamaciones, por favor llame al

555.55.55

ó

Haz click AQUÍ

UCV  
Caracas - Venezuela

Venezuela ▾  

Universidad Central de Venezuela  
Facultad de Ciencias  
© 2016 - Escuela de Computacion

Figura 10. Esquema de layout de la aplicación.

### 4.4.2.3 Codificación

La implementación del login y logout puede verse en el fragmento de la figura 11.

```
197 public function login() {
198     // User is already in?
199     if ($this->Session->read('Auth.User')) {
200         // $this->Session->setFlash('You are logged in!');
201         $this->redirect(array('action' => 'dashboard'), null, false);
202     }
203
204     // Login user
205     if ($this->request->is('post')) {
206         if ($this->Auth->login()) {
207             // debug($SESSION);
208             // Check signup step to avoid too many redirects
209             $this->checkCurrentSignupStep();
210             $this->redirect($this->Auth->redirectUrl());
211         } else {
212             $this->Session->setFlash('Usuario o contraseña Invalida', 'default', array('class' => 'alert alert-error'), 'error-login');
213         }
214     }
215 }
216
217 public function logout() {
218     // Reseller
219     if ($this->Session->read('reseller') != null){
220         $this->Session->delete('channel');
221         $this->Session->delete('reseller');
222     }
223     // Tracking de Pago
224     if ($this->Session->read('PayTracking') != null){
225         $this->Session->delete('PayTracking');
226     }
227
228     return $this->redirect($this->Auth->logout());
229 }
230 }
```

Figura 11. Fragmento de código del login de usuario.

La implementación de la creación del usuario puede verse en el fragmento de la figura 12.

```
272 // Registro de Nuevo Usuario
273 if($this->User->validates()) {
274     // Validate device data
275     if(isset($this->data['Device']['device_model_id'])) {
276         $this->User->create();
277         if($this->User->save($this->data)) {
278             // Create and save the device
279             $this->request->data['Device']['user_id'] = $this->User->id;
280             $this->User->Device->create();
281             if($this->User->Device->save($this->data['Device'])) {
282                 $this->Session->write('Signup.device', $this->User->Device->id);
283                 // Welcome email
284                 // $this->EmailSender->sendWelcomeEmail($this->data['User']['email'], $this->data['User']['email']);
285                 if($this->Auth->login()) {
286                     //***** Add to celuseguro-log*****
287                     $HttpSocket = new HttpSocket();
288                     $data = array('id' => $this->User->id, 'otype' => 'User', 'rid' => 1, 'rtype' => 'System', 'etype' =>
289                         "Registro de usuario");
290                     $results = $HttpSocket->post('http://celulog.local:3005/logs.json', $data);
291
292                     //*****
293                     $campaign = $this->Campaign->findByEmail($email);
294                     if(isset($campaign['Campaign'])) {
295                         $this->Campaign->create();
296                         $this->Campaign->set('first_name', $this->data['User']['first_name']);
297                         $this->Campaign->set('last_name', $this->data['User']['last_name']);
298                         $this->Campaign->set('email', $email);
299                         $this->Campaign->save();
300                     }
301
302                     // Redirect to signup wich sends user to step 2
303                     $this->redirect('/signup'); // array('action' => 'signup')
304                 }
305             }
306         }
307     }
308 }
309 else {
310     $this->Session->setFlash(__('Selecciona un modelo de teléfono.'), 'default', array('class' => 'alert alert-error'),
311         'error-phoneModel');
312 }
```

Figura 12. Fragmento de código sobre la creación de usuario.

#### 4.4.2.4 Pruebas

Las pruebas de aceptación para la interfaz se hicieron observando el funcionamiento del módulo al haber sido terminado y verificando la veracidad de la información almacenada al crear el usuario.

Por otro lado también se realizaron pruebas colocando usuarios no existentes y usuarios existentes con password erróneo.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 13, 14, 15).

| Caso de prueba                                         |                                                                                                          |
|--------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 1                            | Número de historia de usuario: 2,3,4                                                                     |
| Nombre del caso de prueba: Verificación de formularios |                                                                                                          |
| Descripción                                            | Se van a realizar pruebas al formulario de registro                                                      |
| Condiciones de ejecución                               | El cliente ingresa tanto datos correctos, como incorrectos y se verificará la respuesta de la aplicación |
| Resultado esperado                                     | El sistema debe validar y mostrar los errores                                                            |
| Evaluación                                             | Satisfactorio                                                                                            |

Figura 13. Caso de prueba 1

| Caso de prueba                                               |                                                                                                 |
|--------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 2                                  | Número de historia de usuario: 2,3,4                                                            |
| Nombre del caso de prueba: Verificación de acceso al sistema |                                                                                                 |
| Descripción                                                  | Se va a realizar la autenticación del usuario en la aplicación                                  |
| Condiciones de ejecución                                     | El cliente ingresa datos correcto y se verificará si posee credenciales validas para el acceso. |
| Resultado esperado                                           | El sistema debe validar y mostrar los errores                                                   |
| Evaluación                                                   | Satisfactorio                                                                                   |

Figura 14. Caso de prueba 2

| Caso de prueba                                                  |                                                                                 |
|-----------------------------------------------------------------|---------------------------------------------------------------------------------|
| Número de caso de prueba: 3                                     | Número de historia de usuario: 2,3,4                                            |
| Nombre del caso de prueba: funcionalidad de recordar contraseña |                                                                                 |
| Descripción                                                     | Se va a realizar la recuperación de la contraseña                               |
| Condiciones de ejecución                                        | Se ingresaran los datos en el sistema para recuperar la contraseña de la cuenta |
| Resultado esperado                                              | Se debe recibir un email con los datos para la recuperación                     |
| Evaluación                                                      | Satisfactorio                                                                   |

Figura 15. Caso de prueba 3

#### 4.4.3 Iteración 2

En esta iteración se va a continuar con el desarrollo del módulo de cotización del seguro, y el módulo de venta de pólizas, también se va a desarrollar la sección de "mi cuenta" que es la página principal donde se va a desplegar toda la información referente a las pólizas que posee un cliente, y las acciones que puede realizar sobre ellas.

##### 4.4.3.1 Planificación

En la tabla 6 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 6: Historia de usuario para la iteración 2.

| Id   | Fecha      | Requerimiento                       | Tipo   |
|------|------------|-------------------------------------|--------|
| 3.00 | 22/02/2016 | Asegurar equipo en cuenta existente | Nueva  |
| 4.00 | 25/02/2016 | Espacio mi cuenta para clientes     | Nueva  |
| 1.02 | 25/02/2016 | Estructuras del aplicación          | Mejora |

#### 4.4.3.2 Diseño

El sistema va a contemplar el concepto de póliza y se creará un modelo póliza para poder manejarla.

En la figura 16 se tiene el diagrama de clase de los elementos que ahora pertenecen a la aplicación web.

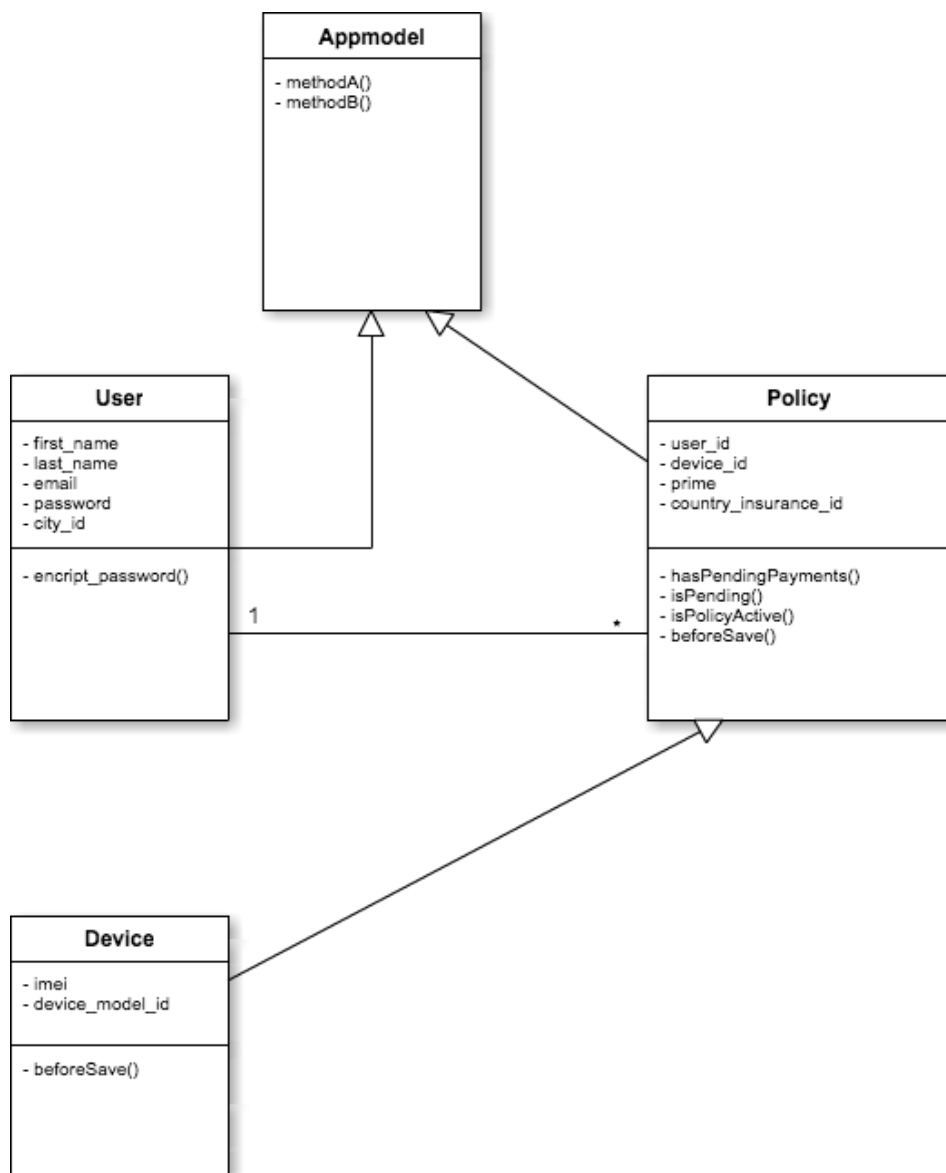


Figura 16. Modelo mejorado del diagrama de clases de la aplicación.

En la figura 17 se muestra cómo será la interfaz para la cotización y compra de pólizas desde una cuenta existente.

Registro

## Selecciona un Equipo

Marca

Apple

Modelo

iPhone 5 32GB

Asegurar tu  
**iPhone 5 32GB**  
te cuesta solo  
**20.237,61 /mes.**  
(con iva)



☐ Acepto las condiciones para adquirir el producto.

[Ver condiciones](#)

Figura 17. Interfaz de cotización de póliza.

En la figura 18 se muestra como es la interfaz para el dashboard de la aplicación mostrando un equipo que fue asegurado y las opciones que le dará el sistema.

Bienvenido Laura Lopez,

Mis Polizas

Asegurar otro Equipo

|                                                                                   |                                                                                            |  |
|-----------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------|--|
|  | iPhone 4S 64GB                                                                             |  |
| Número celular                                                                    | 3000300022                                                                                 |  |
| Número de póliza                                                                  | 21-17-900000482                                                                            |  |
| Costo de póliza mensual                                                           | Bs. 15.775,01                                                                              |  |
| Estado                                                                            | Asegurado                                                                                  |  |
| Fecha de próximo Pago                                                             | 2016-04-15                                                                                 |  |
| ¿Presenta algún siniestro?                                                        | <input type="button" value="Reclamar Hurto"/> <input type="button" value="Reclamar daño"/> |  |

Figura 18. Interfaz del dashboard del sistema.

#### 4.4.3.3 Codificación

Para realizar todo lo que tiene que ver con el registro del equipo y la cotización de pólizas se crearon unas Apis públicas para poder consumirlas desde toda la aplicación.

Estas Apis van a servir para llenar los formularios como los que involucran la dirección, también van a permitir validar si un teléfono ya está registrado en el sistema, si el IMEI ingresado es correcto.

Por otro lado para esta iteración la compra va a ser directa sin la parte correspondiente al módulo de pago.

En las figuras 19 y 20 se mostró el código de varias Apis públicas usadas.

```
64 public function policyRanges(){
65     $this->loadModel('PolicyRange');
66     $data = $this->PolicyRange->find('all');
67     echo json_encode(array('data' => $data));
68 }
69
70 public function randompass(){
71     $key = new DateTime();
72     $key = ($key->format('UYmU')*rand());
73     $pass = AuthComponent::password($key);
74     echo json_encode(array('data' => $pass));
75 }
76
```

Figura 19. Apis publicas usadas por el sistema

```

77 // Duplicidad de IMEI en BD
78 public function imeiValidation(){
79     $imei = $this->request->data('imei');
80     $this->loadModel('Device');
81     $device = $this->Device->findByImei($imei);
82
83     if(isset($device['Device'])){
84         $response = true;
85     }else{
86         $response = false;
87     }
88     echo json_encode(array('response' => $response));
89 }
90
91 // Duplicidad de Telefono en BD
92 public function phoneValidation(){
93     $phone = $this->request->data('phone');
94     $this->loadModel('Device');
95     $device = $this->Device->findByPhoneNumber($phone);
96
97     if(isset($device['Device'])){
98         $response = true;
99     }else{
100         $response = false;
101     }
102
103     echo json_encode(array('response' => $response));
104 }

```

Figura 20. Apis públicas para la validación de los imei y el formato del número de teléfono.

Para la parte del dashboard se colocó una sección donde se listan todos los teléfonos asegurados asociados al cliente y todas las opciones que ese cliente tiene con esa póliza.

En las figuras [2118](#) y [2219](#) se muestra parte del código de la sección del dashboard del usuario.

```
161 public function dashboard() {
162
163     // $this->loadModel('Devices');
164     $this->checkCurrentSignupStep();
165     $this->loadModel('Transaction');
166     $this->loadModel("Tax");
167     $this->loadModel("Claim");
168
169
170     $tax=$this->Tax->find('first');
171     $tax=$tax['Tax']['tax'];
172     $this->set('tax',($tax/100)+1);
173
174     $this->set('currentUser',$this->User->findById($this->Auth->user('id')));
175     // $this->User->Device->contain(array('Policy', 'Claim' => array('Replacement')));
176     $this->set('devices', $this->User->Device->findAllByUserId($this->Auth->user('id'),
177         array(),array('Device.id'=>'desc'),3));
178     $this->set('device_info',$this->User->Device->DeviceModel->find('all'));
179
180     $devices = $this->User->Device->findAllByUserId($this->Auth->user('id'),array(),array('Device.id'=>'desc'),3);
181     $this->set('samsung_devices',$samsung_devices);
182
183     $this->set('transactions',$this->User->Transaction->find('all',array(
184         'conditions' => array('Transaction.user_id' => $this->Auth->user('id'), 'Transaction.code' => 0),
185         'order' => array('Transaction.created DESC'),
186         'limit' => 3
187     )));
188
189     $claims = $this->Claim->findAllByUserId($this->Auth->user('id'),array(),array('Claim.created'=>'desc'),3);
190
191     $this->set('claims',$claims);
192     // debug($claims);
193
194
195 }
```

Figura 21. Controlador que maneja el dashboard del sistema.

```
69
70
71 if(!isset($devices[0])) $devices = array($devices);
72 foreach($devices as $device) {
73
74     $deviceImage = $device['DeviceModel']['image_path'];
75     $policyStatus = ($device['Policy']['transaction_id'] != '') ? __('Asegurado') : __('Procesando Pago');
76     if($device['Policy']['status']=='Active'){
77         $policyStatus = __('Asegurado');
78     }elseif($device['Policy']['status']=='Pending'){
79         $policyStatus = __('Procesando Pago');
80     }elseif($device['Policy']['status']=='suspended'){
81         $policyStatus = __('Reclamación en Estudio');
82     }elseif($device['Policy']['status']=='canceled'){
83         $policyStatus = __('Cancelada');
84     }elseif($device['Policy']['status']=='error'){
85         $policyStatus = __('Hubo un error con el cobro de su póliza');
86     }elseif($device['Policy']['status']=='cancel'){
87         $policyStatus = __('Cancelada');
88     }elseif($device['Policy']['status']=='suspended_pay'){
89         $policyStatus = __('Debe pagar su mensualidad');
90         $simple_boton = $this->Html->link(__('Pagar Mensualidad'), array('controller' => 'claims', 'action' => 'add', '
91             stolen', $device['Device']['id']), array('class' => 'btn btn-small'));
92
93     }
94
95     $actions = '';
96
97     $allowAdd = true;
98     $sending = false;
```

Figura 22. Extracto de código que muestra la presentación de los estados de las pólizas.

#### 4.4.3.4 Pruebas

Las pruebas de aceptación para estas interfaces fueron la verificación de que las vistas tuvieran los campos correctos en los formularios y que esos datos hayan sido enviados y tratados como se esperaba por la aplicación.

Las pruebas de aceptación para las Apis públicas se realizaron mediante la aplicación web ARS, para lo cual usando ARS se consultaron a las Apis y se dio seguimiento los resultados obtenidos, y para la vista del dashboard se colocó información para llenar la vista y se revisó que se desplegara de manera correcta y rápida toda la información.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 23).

| Caso de prueba                                                      |                                                                                                                          |
|---------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 4                                         | Número de historia de usuario: 5,6,7                                                                                     |
| Nombre del caso de prueba: Asegurar dispositivo en cuenta existente |                                                                                                                          |
| Descripción                                                         | Se va a realizar la prueba de cotización y compra de póliza en una cuenta existente                                      |
| Condiciones de ejecución                                            | El cliente usara el botón que dice Asegurar otro equipo y llenara la información y luego le dara paso a pagar la póliza. |
| Resultado esperado                                                  | El sistema debe validar y redirigirlo al botón de pago                                                                   |
| Evaluación                                                          | Satisfactorio                                                                                                            |

Figura 23. Caso de prueba 4

#### 4.4.4 Iteración 3

En esta iteración se va a realizar el módulo de pago, para esto se va a manejar unas pasarelas de pago simuladas con el objetivo de simular una integración de tipo rest, y se va a crear la lógica para el envío de correos al cliente.

#### 4.4.4.1 Planificación

En la tabla 7 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 7: Historia de usuario para la iteración 3.

| Id   | Fecha      | Requerimiento                    | Tipo   |
|------|------------|----------------------------------|--------|
| 5.00 | 01/03/2016 | Integración con pasarela de pago | Nueva  |
| 6.00 | 05/03/2016 | Manejo de documentos vía email   | Nueva  |
| 1.03 | 05/03/2016 | Estructuras del sistema          | Mejora |

#### 4.4.4.2 Diseño

Para la integración con la pasarela de pago se define primero el conjunto de campos de información que se enviaron a nuestra pasarela, de tal manera que esta procese la información y regrese la información referente a si el cliente completo el pago o no.

En la figura 24 se muestra el modelo del botón de pago.

```

102 <div class="boton">
103 <!-- <form action="https://secure.interpagos.net/apps/simplepay/" method="post"> -->
104 <form action="https://secure.interpagos.net/gateway/" method="post">
105 <input name="IdClient" type="hidden" id="IdClient" value="1733">
106 <input name="Token" type="hidden" id="Token" value="<?=sha1("1733-Q2BUQJ27HNE7TFRN-".$reference."-".
107 $this->Number->format($total, array('places' => 2,'decimals' => '.', 'thousands' => '', 'before' => ''));?>" />
108 <input type="hidden" name="IDReference" id="IDReference" value="<?=$reference?>" />
109 <input name="Reference" type="hidden" id="Reference" value="<?=$reference?>" />
110 <input name="Currency" type="hidden" id="Currency" value="COP" />
111 <input name="BaseAmount" type="hidden" id="BaseAmount" value="<?php echo($this->Number->format($policyCost,
112 array('places' => 2,'decimals' => '.', 'thousands' => '', 'before' => ''));?>" />
113 <input name="TaxAmount" type="hidden" id="TaxAmount" value="<?php echo($this->Number->format($taxValue,
114 array('places' => 2,'decimals' => '.', 'thousands' => '', 'before' => ''));?>" />
115 <input name="TotalAmount" type="hidden" id="TotalAmount" value="<?php echo($this->Number->format($total,
116 array('places' => 2,'decimals' => '.', 'thousands' => '', 'before' => ''));?>" />
117 <input name="LenguajeInterface" type="hidden" id="LenguajeInterface" value="SP" />
118 <input name="PayMethod" type="hidden" id="PayMethod" value="1" />
119 <input name="RecurringBill" type="hidden" id="RecurringBill" value="0" />
120 <input name="RecurringBillTimes" type="hidden" id="RecurringBillTimes" value="0" />
121 <input name="Test" type="hidden" id="Test" value="0" />
122 <input name="ExtraData1" type="hidden" id="ExtraData1" value="<?=$policy['Policy']['id']?>" />
123 <input name="ExtraData2" type="hidden" id="ExtraData2" value="<? echo($_SERVER['REMOTE_ADDR']);?>" />
124
125 <input name="PageAnswer" type="hidden" id="PageAnswer"
126 value="https://www.celuseguro.com/policies/payinterpagos/<?=$policy['Policy']['id']?>"
127 <input name="PageConfirm" type="hidden" id="PageConfirm"
128 value="https://www.celuseguro.com/policies/payinterpagos/<?=$policy['Policy']['id']?>"
129
130 <button type="submit" class="pay_button btn btn-large btn-celu pago" href="#">
131 <div class="row-fluid">
132 <div class="span3"><i class="icon-shopping-cart icon-2x pull-left"></i></div>
133 <div class="span9"><div class="text"><i class="icon-lock"></i> PAGO SEGURO<br/><span>Interpagos.net</span></div></div>
134 </div>
135 </button>
136 </form>
137 </div>
138 </div>
139 </div>
140
141 <div class="clear"></div>

```

Figura 24. Código HTML que contiene la estructura del botón de pago.

Como se está realizando una integración con un sistema de pagos, se le agrega varios sistemas de seguridad para verificar que la conexión sea segura. Como primer paso se agregaran campos encriptados al post enviado al comercio con la información que permita validar que la conexión es legítima para esto se usó una encriptación SHA1 y para la encriptación del mensaje se utilizó 3DES.

Para el diseño de las interfaces de pago se crea una vista que permita al usuario elegir el tipo de pago, y luego en la siguiente vista puede revisar el pago y realizar el pago con la opción de su preferencia.

En la figura 25 se muestra la vista correspondiente a la sección de pagos donde se desglosa la información correspondiente que se va a facturar.

## Pagar póliza

Usted está pagando por una póliza

|                            |                |
|----------------------------|----------------|
| Marca:                     | Apple          |
| Modelo:                    | iPhone 4S 16GB |
| Valor de la Prima Mensual: | Bsf. 9.000,00  |
| IVA(12%):                  | Bsf. 1.080,00  |
| Total a mensual a pagar:   | Bsf. 10.080,00 |

Selecciona el medio de pago que quieras usar para pagar tu póliza.

Pago **simple** con Tarjeta de Crédito o Débito.



**PAGO SEGURO**  
pasarela1.com

Figura 25. Interfaz de pagos de la aplicación.

### 4.4.4.3 Codificación

Parte de la codificación se muestra en la figura 26, específicamente la parte donde es encriptada la data que se enviará a la pasarela, esto se va a realizar en un controlador que va a recibir el post realizado por el usuario y lo va a adaptar al que debe recibir la pasarela seleccionada. Esto se hace de esta manera para poder tener integración con varias pasarelas de pago.

```
689
690     if($this->request->host() == 'gateway.segurati.com'){
691         $key = '2C105C9969393B90348DA8EC0A1C1528'; //Produccion
692     }else{
693         $key = 'F0BC3BF03A21C128EB9EE9A563AA1CF5'; //Staging
694     }
695     $rrr = str_replace(' ', '+', $payment['json']);
696     $decode=$this->decrypt3DES($this->tobin($key),str_replace(' ', '+', $payment['json']));
697     $decode = implode(' ', array_filter(str_split($decode, 1), 'ctype_print'));
698     $obj = json_decode($decode);
699     $codRespuesta=$obj->{'codRespuesta'};
700     $idConcepto=$obj->{'idConcepto'};
701
702     list($Ppolicy, $ref, $monto, $spr, $currency, $user_ip) = split('~', $idConcepto);
703     $check=$this->PayTransaction->find('first',array('conditions' => array('PayTransaction.reference' => $ref)));
704
705     $data = array(
706         'policy' => $Ppolicy,
707         'status' => $codRespuesta
708     );//fin del arreglo
709     //seteo variables para el tracking pixel
710     $this->set('policy_id',$check['PayTransaction']['policy_id']);
711     $this->set('reference',$check['PayTransaction']['reference']);
712     $this->set('amount',$check['PayTransaction']['amount']);
713
```

Figura 26. Extracto de código donde se realiza la encriptación de la data.

Luego que la data es encriptada, se correrá un script que armara el post nuevamente y lo enviara automáticamente a la pasarela. En la figura 27 se puede ver el script que se va a utilizar.

```

1676 private function redirect2($url, $data) {
1677     $html = "<html><body><form id='form' action='$url' method='post'>";
1678     foreach ($data as $key => $value) {
1679         $html .= "<input type='hidden' name='$key' value='$value'>";
1680     }
1681     $html .= "</form><script>document.getElementById('form').submit();</script>";
1682     $html .= "</body></html>";
1683     print($html);
1684 }

```

Figura 27. Función que redirección al cliente a la pasarela de pago.

Luego que el cliente realice su pago, desde el lado de la aplicación se crea un api para poder recibir la respuesta de la pasarela de pago. Para poder desde nuestra aplicación procesar el pago.

En la figura 28 se muestra el api que va a recibir los post de las pasarelas

```

523 public function payaddcel() {
524     $this->autoRender = false;
525
526     if($this->request->is('post')) {
527         $payment = $this->data;
528         // CREO ENTRADA EN LOG
529         if(!isset($payment['json'])){
530             $this->badPost('addcel',$payment,'Transaction','badPost Transaction');
531             return false;
532         }
533
534         if($this->request->host() == 'gateway.segurati.com'){
535             $key = '2C105C9969393B90348DA8EC0A1C1528'; //Produccion
536         }else{
537             $key = 'F0BC3BF03A21C128EB9EE9A563AA1CF5'; //Staging
538         }
539
540
541         $rrr = str_replace(' ','+',$payment['json']);
542
543         //Decodifico el mensaje
544         $decode=$this->decrypt3DES($this->tobin($key),str_replace(' ','+',$payment['json']));
545         $decode = implode('', array_filter(str_split($decode, 1), 'ctype_print'));
546         $obj = json_decode($decode);
547         $concepto=$obj->{'concepto'};
548         $codRespuesta=$obj->{'codRespuesta'};
549         $idTransaccion=$obj->{'idTransaccion'};
550         $descRespuesta=$obj->{'descRespuesta'};
551         $idConcepto=$obj->{'idConcepto'};
552         $numAprobacion=$obj->{'numAprobacion'};
553         $fechaTransaccion=$obj->{'fechaTransaccion'};
554
555         $r = $this->payAddcelApi($concepto,$codRespuesta,$idTransaccion,$descRespuesta,
556             $idConcepto,$numAprobacion,$fechaTransaccion,$obj);
557         echo json_encode(array('data' => 'all good here'));
558     }
559 }

```

Figura 28. Api publica que recibe el post de la pasarela de pago y lo procesa.

Luego que el pago es procesado, la aplicación se encargara de enviar un email al cliente, en el cual si el pago fue exitoso se le enviara todos los datos de su póliza y los documentos en pdf que la acreditan. Para el envío de emails se usó un servicio online llamado mandrill, en la figura 29 se muestra la configuración del api de mandrill y en la figura 30 la función encargada de enviar los emails.

```
52 public $mandrill = array(  
53     'transport' => 'Mandrill',  
54     'uri' => 'https://mandrillapp.com/api/1.0/',  
55     'key' => 'ZtFto9rd9HuL1hpgzQHqKA'  
56 );  
57  
58 public $smtp = array(  
59     'transport' => 'Smt',  
60     'from' => array('site@localhost' => 'My Site'),  
61     'host' => 'localhost',  
62     'port' => 25,  
63     'timeout' => 30,  
64     'username' => 'user',  
65     'password' => 'secret',  
66     'client' => null,  
67     'log' => false,  
68     //'charset' => 'utf-8',  
69     //'headerCharset' => 'utf-8',  
70 );  
71
```

Figura 29. Segmento de configuración del servidor de mandrill.

```

850 public function createEmail($policyID){
851     $this->loadModel('Policy');
852     $this->loadModel('Device');
853     $this->loadModel('City');
854     $this->Policy->contain('User', 'Device', 'Transaction');
855     $policy = $this->Policy->findById($policyID);
856     $user = $policy['User'];
857     $device = $this->Device->findById($policy['Device']['id']);
858     $city = $this->City->findById($policy['User']['city_id']);
859
860     // GENERO CLAUSULADO
861     $clausulado = file_get_contents('http://www.web.com/pdf/clausulado.pdf');
862     $clausulado_encoded = base64_encode($clausulado);
863
864     // GENERO PDF POLIZA
865     $poliza_pdf = $this->createPdf($policyID);
866
867     // ENVIDIO CORREO
868     $this->EmailSender->sendPolicyCreatedEmail(
869         $user['email'], // destination email
870         array_merge($user,$device,$policy['Device'],$policy['Transaction'],$policy['Policy']),
871         array(
872             'Policy.pdf' => array(
873                 'name' => 'Poliza_CeluSeguro.pdf',
874                 'type' => 'application/pdf',
875                 'content' => base64_encode($poliza_pdf)
876             ),
877             'Clausulado.pdf' => array(
878                 'type' => "application/pdf",
879                 'content' => $clausulado_encoded,
880                 'name' => 'clausulado.pdf'
881             )
882         ) // attachment
883     );
884
885 }

```

Figura 30. Función que envía los emails del sistema.

Para la creación de los documentos en formato pdf primero se crearon las vistas en la aplicación y luego se generó el pdf con una función que podemos ver en la figura 31.

```

821 // Crear PDF
822 public function createPdf($policyID){
823     $this->loadModel('Policy');
824     $this->loadModel('Device');
825     $this->loadModel('City');
826     $this->loadModel('DeviceModel');
827     $this->Policy->contain('User', 'Device', 'Transaction');
828     $policy = $this->Policy->findById($policyID);
829     $user = $policy['User'];
830     $device = $this->Device->findById($policy['Device']['id']);
831     $city = $this->City->findById($policy['User']['city_id']);
832     $deviceModel = $this->DeviceModel->findById($device['Device']['device_model_id']);
833
834
835     ini_set('memory_limit', '512M');
836     $poliza_pdf = $this->doc('policy', array(
837         'PolicyReference' => Configure::read('Policies.ref') + $policyID,
838         'Device' => $policy['Device'],
839         'User' => $user,
840         'Brand' => $deviceModel['Brand'],
841         'City' => $city['City'],
842         'DeviceModel' => $deviceModel['DeviceModel'],
843         'Policy' => $policy['Policy'],
844     ));
845
846     return $poliza_pdf;
847 }
848

```

Figura 31. Función que convierte las vistas en pdf para su envío por email.

A continuación se muestra un ejemplo en la figura 32 del documento pdf generado por la aplicación que hace referencia a un ejemplo del certificado de póliza que se le envía al cliente.

Compañía de 555 555-5555  
 Seguros C.A. 555 555-5555  
 Rif. J-XXXXXXX-X 555 555-5555  
 Ucv. 555 555-5555  
 Caracas www.companiadeseguros.com  
 Venezuela

**CERTIFICADO INDIVIDUAL DE SEGURO**  
**PÓLIZA N° «401376»**  
**CERTIFICADO N°2016101475448360**

|                                      |                              |                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------------------------------|------------------------------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Seguro por cuenta tomado por:</b> | Celuseguro Tesis             | <b>Rif:</b>      | J-XXXXXXX-X                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Asegurado / Beneficiario:</b>     | Marco Pascale                | <b>CC/:</b>      | 15379286                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Ciudad:</b>                       | Barranco de Loba             | <b>Email:</b>    | r4ncor@gmail.com                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|                                      |                              | <b>Teléfono:</b> | 1231231231                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>Información del Celular</b>       |                              |                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Marca:</b>                        | Apple                        | <b>Modelo:</b>   | iPhone 4S 32GB                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>No. de celular:</b>               | 1231231231                   | <b>IMEI:</b>     | 013842137708624                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Programa Aceptado</b>             |                              |                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <b>Coberturas</b>                    | <b>Plan Mensual</b>          |                  | <b>Vigencia</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Daño Total Accidental                | Seguro de Terminales Móviles |                  | Desde las 00 horas del: 2016-10-04                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|                                      | Vr Prima: Mensual            | Bsf. 9.000,00    | <b>Hasta:</b><br>La cobertura de seguro a la que hace referencia este certificado estará vigente hasta que se incurra en alguna de las causales de terminación del contrato de seguro establecidas en las condiciones generales o en la ley. El monto de la prima, indicada en este certificado, se incrementarán en cada aniversario de este documento con base en el índice anual de precios al consumidor, IPC, establecido al 31 de diciembre del año anterior al aniversario. |
|                                      | IVA:                         | Bsf. 1.440,00    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Hurto Calificado                     | Total: Mensual               | Bsf. 10.440,00   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|                                      | Deducible Aplicado:          | 20% 1er evento   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|                                      |                              |                  | 30% 2do evento                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

**INFORMACIÓN IMPORTANTE**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Laudantium fugit dolorum accusamus cumque, animi iusto maiores optio similique cum atque!  
 Veniam eveniet illum ratione atque, labore provident, sint quasi porro.  
 Lorem ipsum dolor sit amet, consectetur adipiscing elit. Magnam veritatis magni quibusdam modi eius cumque hic necessitatibus placeat est praesentium quam natus, fuga maiores aperiam fugit, temporibus rem! Nobis, nihil? Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quasi neque eum, minus, quam quis iusto debitis reiciendis omnis! Nam assumenda repellat, ad consectetur quis laudantium vitae alias blanditis quod esse?.

**DECLARACIONES Y AUTORIZACIONES**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Distinctio totam quis expedita, molestias in, suscipit. Voluptas quibusdam veniam quidem commodi praesentium quia consequatur neque, nihil molestiae cum? Nihil, unde laudantium?

Teléfono: 555.5555. Caracas-Venezuela  
 Correo electrónico: compania@seguros.com  
 Página Web: http://www.companiadeseguros.com\*

**Figura 32. Certificado de póliza.**

#### 4.4.4.4 Pruebas

Para las pruebas de aceptación, se realizó el envío de data controlada a la pasarela y se vio la respuesta de la misma, luego se revisó el correcto funcionamiento del procesamiento de la respuesta. También se realizó pruebas de encriptación y de verificación de la data enviada y recibida para el envío de emails, se comprobó la correcta conexión con el servicio de envío de emails y se hicieron pruebas adquiriendo pólizas y revisando que el pdf y el email correspondan con la información correcta de la aplicación.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 33,34).

| Caso de prueba                                                        |                                                                                                                     |
|-----------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 5                                           | Número de historia de usuario: 5,6,7                                                                                |
| Nombre del caso de prueba: Revisión de la póliza en la pagina de pago |                                                                                                                     |
| Descripción                                                           | A la hora de comprar una póliza luego de finalizado los formularios de registros se debe mostrar la página de pago. |
| Condiciones de ejecución                                              | El cliente comprara una poliza y debe llegar a la ejecución del pago                                                |
| Resultado esperado                                                    | El sistema debe generar correctamente los datos                                                                     |
| Evaluación                                                            | Satisfactorio                                                                                                       |

Figura 33. Caso de prueba 5

| Caso de prueba                                               |                                                                                            |
|--------------------------------------------------------------|--------------------------------------------------------------------------------------------|
| Número de caso de prueba: 6                                  | Número de historia de usuario: 8,9                                                         |
| Nombre del caso de prueba: Recepción de documentos vía email |                                                                                            |
| Descripción                                                  | Al momento de comprar una póliza se debe recibir los emails adecuados.                     |
| Condiciones de ejecución                                     | El cliente comprara una póliza y automaticamente le llegara un email a su cuenta de correo |
| Resultado esperado                                           | El sistema debe generar correctamente los datos                                            |
| Evaluación                                                   | Satisfactorio                                                                              |

Figura 34. Caso de prueba 6

#### 4.4.5 Iteración 4

Para esta iteración se dispone de un sistema que maneja tanto usuarios como pólizas de seguro, ahora se desarrolló todo lo que tiene que ver con el reporte de un siniestro y el manejo de documentos para revisión.

En esta iteración también se desarrollada la conexión de la aplicación con el servidor de Amazon s3 para el almacenaje de documentos en la nube.

##### 4.4.5.1 Planificación

En la tabla 8 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 8. Historia de usuario para la iteración 4.

| Id   | Fecha      | Requerimiento                              | Tipo   |
|------|------------|--------------------------------------------|--------|
| 7    | 18/03/2016 | Colocar un reclamo en una póliza existente | Nueva  |
| 8    | 18/03/2016 | Almacenaje de documentos en la nube        | Nueva  |
| 1.04 | 19/03/2016 | Estructuras del sistema                    | Mejora |

#### 4.4.5.2 Diseño

Para la creación del diseño se definieron los campos que debería llevar la interfaz de notificación de siniestros, en la figura 35 se puede visualizar como sería el formulario.

### Información del Equipo

**Fecha de Compra**

28-05-2016Adjunte un archivo

**Invoice Number**

### Documentos Necesarios

**CÉDULA**

Adjunte un archivo

**FACTURA**

Adjunte un archivo

**DENUNCIA POLICIAL O ANTE AUTORIDAD COMPETENTE**  
(No se aceptan denuncias electrónicas realizadas vía internet)

Adjunte un archivo

### Información del siniestro

**Fecha del siniestro**

28-05-2016

Figura 35. Interfaz de reporte de siniestros.

#### 4.4.5.3 Codificación

Para la codificación creó el controlador que va a manejar los siniestros, y se agregaran los botones a la interfaz del dashboard para que se puedan reportar siniestros con el equipo seleccionado.

En las figuras 360 podemos observar un fragmento de código donde se manejan la planillas de reclamo.

```
576 public function claimform($claim_id){
577     $this->layout = 'clean';
578     $this->loadModel('Claim');
579     $this->loadModel('Device');
580     $this->loadModel('Brand');
581     $this->loadModel('City');
582
583     $this->Device->contain('Policy', 'Claim', 'DeviceModel', 'User');
584     // $this->Brand->contain('Brand');
585     // $this->City->contain('City');
586
587     $claim = $this->Claim->findById($claim_id);
588     $device = $this->Device->findById($claim['Claim']['device_id']);
589     $brand = $this->Brand->findById($device['DeviceModel']['brand_id']);
590     $city = $this->City->findById($device['User']['city_id']);
591
592     $data = array(
593         'Device' => $device['Device'],
594         'User' => $device['User'],
595         'Model' => $device['DeviceModel'],
596         'Brand' => $brand['Brand'],
597         'UCity' => $city['City'],
598         'Claim' => $device['Claim'],
599     );
600     $this->set('data', $data);
601     // debug($data);
602
603 }
```

Figura 36. Extracto de que genera el formulario de la colocación de siniestro.

Luego que se llene la información solicitada, toda la data suministrada será almacenada y los archivos que subió a la plataforma el cliente serán colocados en un servidor remoto que utiliza la plataforma de almacenamiento de archivos de Amazon S3, para hacer esto se realizara la integración con dicha plataforma como se muestra en la figura 37

```

2  #!/usr/local/bin/php
3  <?php
4  /**
5   * $Id$
6   *
7   * S3 class usage
8   */
9  if (!class_exists('S3')) require_once 'S3.php';
10 // AWS access info
11 if (!defined('awsAccessKey')) define('awsAccessKey', 'change-this');
12 if (!defined('awsSecretKey')) define('awsSecretKey', 'change-this');
13 $uploadFile = dirname(__FILE__).'/S3.php'; // File to upload, we'll use the S3 class since it exists
14 $bucketName = uniqid('s3test'); // Temporary bucket
15 // If you want to use PECL Fileinfo for MIME types:
16 //if (!extension_loaded('fileinfo') && @dl('fileinfo.so')) $ENV['MAGIC'] = '/usr/share/file/magic';
17 // Check if our upload file exists
18 if (!file_exists($uploadFile) || !is_file($uploadFile))
19     exit("\nERROR: No such file: $uploadFile\n\n");
20 // Check for CURL
21 if (!extension_loaded('curl') && !dl(PHP_SHLIB_SUFFIX == 'so' ? 'curl.so' : 'php_curl.dll'))
22     exit("\nERROR: CURL extension not loaded\n\n");
23 // Pointless without your keys!
24 if (awsAccessKey == 'change-this' || awsSecretKey == 'change-this')
25     exit("\nERROR: AWS access information required\n\nPlease edit the following lines in this file:\n\n".
26         "define('awsAccessKey', 'change-me');\n\ndefine('awsSecretKey', 'change-me');\n\n");
27 // Instantiate the class
28 $s3 = new S3(awsAccessKey, awsSecretKey);
29 // List your buckets:
30 echo "S3::listBuckets(): ".print_r($s3->listBuckets(), 1)."\n";
31 // Create a bucket with public read access
32 if ($s3->putBucket($bucketName, S3::ACL_PUBLIC_READ)) {
33     echo "Created bucket {$bucketName}";
34     // Put our file (also with public read access)
35     if ($s3->putObjectFile($uploadFile, $bucketName, basename($uploadFile), S3::ACL_PUBLIC_READ)) {
36         echo "S3::putObjectFile(): File copied to {$bucketName}/".basename($uploadFile).PHP_EOL;
37         // Get the contents of our bucket
38         $contents = $s3->getBucket($bucketName);
39         echo "S3::getBucket(): Files in bucket {$bucketName}: ".print_r($contents, 1);
40         // Get object info
41         $info = $s3->getObjectInfo($bucketName, basename($uploadFile));
42         echo "S3::getObjectInfo(): Info for {$bucketName}/".basename($uploadFile).": ".print_r($info, 1);
43     }
44     if ($s3->deleteObject($bucketName, basename($uploadFile))) {
45         echo "S3::deleteObject(): Deleted file\n";
46         // Delete the bucket we created (a bucket has to be empty to be deleted)
47         if ($s3->deleteBucket($bucketName)) {
48             echo "S3::deleteBucket(): Deleted bucket {$bucketName}\n";
49         } else {
50             echo "S3::deleteBucket(): Failed to delete bucket (it probably isn't empty)\n";
51         }
52     } else {
53         echo "S3::deleteObject(): Failed to delete file\n";
54     }
55 } else {
56     echo "S3::putObjectFile(): Failed to copy file\n";
57 }
58 } else {
59     echo "S3::putBucket(): Unable to create bucket (it may already exist and/or be owned by someone else)\n";

```

Figura 37. Función que permite subir y manejar archivos en los servidores de Amazon S3.

Luego que los archivos sean subidos al servidor se enviara una notificación vía email y por el sistema para que se procese los siniestros a través del módulo de aseguradora.

#### 4.4.5.4 Pruebas

Para probar las funciones que permiten subir archivos a S3, se realizaron pruebas individuales que permitieron ir subiendo varios archivos de prueba y revisando el correcto funcionamiento de la aplicación, para las pruebas de la interfaz de reporte de siniestro, se colocaron varios siniestros en el sistema y se verificó en base de datos el correcto almacenamiento de los datos.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 38).

| Caso de prueba                                                     |                                                                                                    |
|--------------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 7                                        | Número de historia de usuario: 10,11                                                               |
| Nombre del caso de prueba: Almacenamiento de documentos en la nube |                                                                                                    |
| Descripción                                                        | Cuando se reportan siniestros en la aplicación, se deben subir a la nube los archivos del reclamo. |
| Condiciones de ejecución                                           | El cliente colocara un reclamo, y luego subirá sus documentos a la plataforma                      |
| Resultado esperado                                                 | El sistema debe generar correctamente los datos                                                    |
| Evaluación                                                         | Satisfactorio                                                                                      |

Figura 38. Caso de prueba 7

#### 4.4.6 Iteración 5

En este sistema existen procesos que pueden tardar mucho tiempo en ejecutarse, los más destacados son el envío de correo masivos y la revisión diaria de los estatus de las pólizas, del caso de la revisión de las pólizas se debe revisar las fechas, estatus e información de pago estos procesos suele llevar mucho tiempo.

Para este tipo de problemas la tecnología web tiene una solución que es el manejo de workers que van a ser ejecutados en segundo plano, para realizar esto se usó un plugin de Rails que se llama Delayed::Job.

Delayed::Job según lo explican sus creadores, es un servidor y administrador de tareas para que estén sean ejecutadas en segundo plano, mientras el usuario y la aplicación Web pueden seguir interactuando síncronamente con otras acciones y servicios requeridos. Además la aplicación puede comunicarse con el hilo que ejecuta la tarea a través del administrador de tareas para consultar su estado de progreso, estatus, para detenerlo, etc.

#### 4.4.6.1 Planificación

En la tabla 9 se muestran las historias de usuario desarrolladas en esta iteración.

Tabla 9. Historia de usuario para la iteración 5

| Id | Fecha      | Requerimiento                                   | Tipo  |
|----|------------|-------------------------------------------------|-------|
| 9  | 05/04/2016 | Implementación de procesos largos en background | Nueva |

#### 4.4.6.2 Codificación

Básicamente Delayed::Job es un servidor que corre un demonio, y se accede a los hilos de ejecución de tareas (workers) a través de una clase llamada MiddleMan (Desde la aplicación Rails). Para cada tarea que se desea realizar con Delayed::Job se debe implementar una clase (worker) que herede de Delayed::Job::Rails y un método do\_work() que es donde estará el código de la tarea a ejecutarse.

En la figura 39 se tiene la implementación de un worker que realiza la tarea de

```
1 class PolicyValidation
2
3   @queue = :validation_queue
4
5   def self.perform()
6     ActiveRecord::Base.transaction do
7       today = Date.today
8       @billing_policies = []
9       @policies = Policy.all
10      @policies.each do |p|
11
12        if Date.today == p.expiration_date and p.status != 'canceled' and p.status != 'canceled_change_device' and p.status != 'null'
13          renovar = true
14          renovar = false if p.insurance.name.downcase.include? 'seguros del estado' and p.pending_claims?
15
16          if renovar
17            PolicyRenovation.perform(p.id, p.billing_date, p.expiration_date)
18            @billing_policies.push({ id: p.external_policy_number, type: 'Renovation' })
19            if p.renewed == true
20              registrar_log(p, 'Renovacion de la póliza')
21              Resque.enqueue(MailAlert, "renovation", p.id)
22            end
23          end
24        end
25      end
26
27      unless p.billing_date == p.expiration_date
28        if !p.transaction.nil? and (p.status == 'Active' or p.status == 'suspended' or p.status == 'suspended_pay')
29          unless p.billing_date.nil? or p.billing_date.blank?
30
31            # ALERTA DE COBRO SIMPLE 8 DIAS
32            if (((p.billing_date - today).to_i == 8) and p.pay_type == 'simple')
33              billing_alert("8days", p)
34              registrar_log(p, "ALERTA DE COBRO SIMPLE 8 DIAS")
35              @billing_policies.push({ id: p.external_policy_number, type: '8days' })
36            end
37
38            # ALERTA DE COBRO SIMPLE 4 DIAS
39            if (((p.billing_date - today).to_i == 4) and p.pay_type == 'simple')
40              billing_alert("4days", p)
41              registrar_log(p, "ALERTA DE COBRO SIMPLE 4 DIAS")
42              @billing_policies.push({ id: p.external_policy_number, type: '4days' })
43            end
44
45            # ALERTA DE COBRO SIMPLE VENCIDO
46            if (((p.billing_date - today).to_i == 0) and p.pay_type == 'simple')
47              simple_expire(p)
48              registrar_log(p, "ALERTA DE COBRO SIMPLE VENCIDO")
49              @billing_policies.push({ id: p.external_policy_number, type: 'Simple' })
50            end
51
52            # ALERTAS LUEGO DE VENCIDO EL COBRO
53
54            # ALERTA DE COBRO 3 DIAS DE VENCIDO
55            if ((p.billing_date - today).to_i == -3)
56              billing_alert("3days", p)
57              registrar_log(p, "ALERTA DE COBRO 3 DIAS DE VENCIDO")
58              @billing_policies.push({ id: p.external_policy_number, type: '3days' })
59            end
60          end
61        end
62      end
63    end
64  end
65 end
```

Figura 39. Función que ejecuta el worker para la revisión diaria de las pólizas.

procesar todas las pólizas y enviar correos dependiendo del caso.

Cabe señalar que para mantener el código del proceso en su lugar (ApplicationController), no se mudó la implementación y la lógica hacia el worker, sino que este último llama al método, y escribe el estado del proceso en el worker.

Es importante dejar claro que Delayed::Job cuando inicia es realmente un servicio aparte, cuyo código reside dentro del mismo proyecto.

#### 4.4.6.3 Pruebas

Para probar la ejecución de los procesos largos en segundo plano con Delayed::Job, se hizo una interfaz temporal donde se monitoreaba el progreso de la tareas, además en el servidor de Delayed::Job, se obtuvieron las mismas salidas por consola que las generadas en el servidor de aplicación principal con una invocación directa a los métodos. Por ejemplo el procesamiento de las pólizas se revisó el cambio de estatus y él envió de emails a los usuarios registrados.

#### 4.4.7 Iteración 6

Prácticamente desde el comienzo se desarrollaron unas funcionalidades que permiten a un usuario realizar todo tipos de acciones desde el registro de un dispositivo hasta el reporte de siniestros. Esto implica la creación de la figura del administrador y las funcionalidades de mantenimiento y monitoreo.

Mientras que cualquier usuario puede utilizar la aplicación como cliente. El administrador configura la aplicación y puede tomar decisiones que modifiquen su estado.

En esta iteración se agrupan las funcionalidades del administrador en una aplicación aparte creada en rails, todo esto para poder aprovechar una gema administrativa que se llama ActiveAdmin.

##### 4.4.7.1 Planificación

En la tabla 10 se muestran las historias de usuario desarrolladas en esta iteración:

Tabla 10. Historia de usuario para la iteración 6.

| Id | Fecha      | Requerimiento                                        | Tipo  |
|----|------------|------------------------------------------------------|-------|
| 10 | 06/04/2016 | Creación del sistema administrador                   | Nueva |
| 11 | 06/04/2016 | Creación de roles para la administración del sistema | Nueva |
| 12 | 06/04/2016 | Definir las vistas para la administración            | Nueva |

##### 4.4.7.2 Diseño

Uno de los beneficios de la utilización de la gema ActiveAdmin en el administrador es la facilidad para listar la información y poder generar todo tipo de reportes.

En la tabla 11 se muestra la funcionalidad del administrador junto con una breve descripción.

**Tabla 11. Funcionalidad del administrador.**

| <b>Función</b>         | <b>Descripción</b>                                                                                                                                                                                            |
|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Iniciar Sesión         | El administrador ingresa su login y password, luego se verifica y se activa la sesión.                                                                                                                        |
| Configurar sistema     | El administrador establece todos los datos necesarios para que el sistema funcione correctamente.                                                                                                             |
| Monitoreo del sistema  | El administrador debe monitorear el sistema y para ello puede revisar y consultar toda la data y a través del log del sistema puede encontrar errores para pasarlos al área de desarrollo para su corrección. |
| Agregar Roles          | El administrador principal puede crear otros administradores con diferentes roles para que estos realicen actividades administrativas, y no tengan acceso a todo el sistema.                                  |
| Eliminar un documento  | Esta opción está disponible para cada documento cuando el administrador está en la vista de resultados de una búsqueda.                                                                                       |
| Generación de reportes | El administrador del sistema también tiene la opción de generar distintos reportes para poder analizar las ventas de pólizas, cantidad de reclamos y otros.                                                   |

En la tabla 12 se muestra la definición del perfil de usuario con una breve descripción.

**Tabla 12. Perfiles de usuario.**

| Perfil de usuario         | Descripción                                                              |
|---------------------------|--------------------------------------------------------------------------|
| Administrador aseguradora | Acceso al módulo de la aseguradora.                                      |
| Administrador general     | Acceso a todos los módulos de la aplicación.                             |
| Administrador local       | Usuario con restricciones a módulos.                                     |
| Usuario                   | Usuario externo a la aplicación puede ser un visitante de la aplicación. |

En la figura 40 se muestra como es la interfaz de ActiveAdmin adaptado al sistema, se desplegaron las pólizas con toda la información, y enlazó cada campo con el elemento correspondiente, lo cual permitió la revisión de cualquier aspecto de la aplicación.

| Resumen    |                 |                       |                |                       |            |            |       |                     |                 |           |               |            |                                           |                      |
|------------|-----------------|-----------------------|----------------|-----------------------|------------|------------|-------|---------------------|-----------------|-----------|---------------|------------|-------------------------------------------|----------------------|
| Registro   | # Póliza        | Estatus               | Cliente        | Email                 | ID (CC)    | Teléfono   | Marca | Modelo              | IMEI            | Pago      | Medio de Pago | Recurrente | Transacción                               | Fecha de transacción |
| 13-05-2016 | 1006            | POR PAGAR             | Marco Pascate  | r4ncor111@gmail.com   | 15379285   | 8273645911 | Apple | iPhone 5C 8GB       | 501398994234526 | -         | -             | false      | -                                         | -                    |
| 15-03-2016 | 21-17-900000482 | PAGADA                | Laura Lopez    | llopez@aseguerosa.com | 1234       | 3000300022 | Apple | iPhone 4S 64GB      | 353308076614122 | 15775.01  | A D C E L     | No         | <a href="#">1407.401005.2016315102854</a> | 15/03/2016 - 02:33   |
| 07-03-2016 | 21-17-900000481 | CON REGLAMO PENDIENTE | Ricardo Ubieta | rubieta@celatam.com   | 333333     | 3333333373 | Apple | iPhone 4S 16GB      | 013851006460910 | 141250.69 | A D C E L     | No         | <a href="#">1404.401004.201638111440</a>  | 08/03/2016 - 04:15   |
| 07-03-2016 | 21-17-900000480 | PAGADA                | Brian Guacaran | bie47.delfa@gmail.com | 5622024139 | 4163464597 | Apple | iPhone 6 Plus 128GB | 012597108854481 | 663594.66 | A D C E L     | No         | <a href="#">1406.401003.201637183335</a>  | 07/03/2016 - 11:04   |
| 07-03-2016 | 21-17-900000479 | PAGADA                | Marco Pascate  | r4ncor@gmail.com      | 1212121212 | 1231231237 | Apple | iPhone 5C 8GB       | 497245026290354 | 314748.46 | A D C E L     | No         | <a href="#">1406.401002.201637181559</a>  | 07/03/2016 - 10:46   |
| 05-03-2016 | 21-17-900000478 | CANCELADA             | Marco Pascate  | m.pascate5@gmail.com  | 15379286   | 9898765434 | Apple | iPhone 6 16GB       | 330546576428760 | 461241.57 | A D C E L     | No         | <a href="#">1403.401001.201635172512</a>  | 05/03/2016 - 09:55   |

**Figura 40. Interfaz del dashboard de la aplicación administrativa.**

También el sistema administrador va a estar encargado de toda la configuración del sistema, variables de precio, logos de la página, texto del footer entre otros y todo eso se debe poder modificar, en la figura 41 se muestra el menú de configuración de la aplicación.

#### 4.4.7.3 Codificación

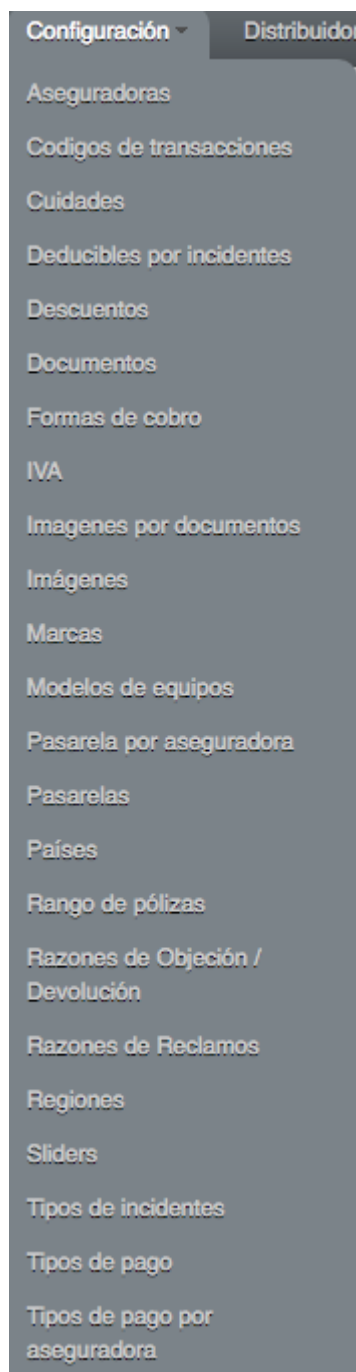


Figura 41. Menú de configuración del sistema.

En la figura 42 se tiene un extracto del código de configuración del sistema donde se implementa la asignación de las fotos que mostrara el sistema en la página de inicio.

```
1 # encoding: UTF-8
2 ActiveAdmin.register DocumentImage do
3
4   menu parent: 'Configuración', if: proc { current_admin_user.email.include? 'rubietta' or current_admin_user.email.include?
5
6   actions :all, except: [ :show ]
7
8   config.sort_order = 'country_insurance_id_asc'
9   config.batch_actions = false
10  config.comments = false
11
12  filter :country_insurance
13  filter :document
14
15  index do
16    column :country_insurance
17    column :document_name
18    column :image_name
19    default_actions
20  end
21
22  form do |f|
23    documents = options_for_select(params[:documents], selected: params[:selected_document])
24    images = options_for_select(params[:images], selected: params[:selected_image])
25    f.inputs "Datos del registro" do
26      f.input :country_insurance
27      f.input :document, as: :select, collection: documents
28      f.input :image, as: :select, collection: images
29    end
30    f.actions
31  end
32
33  controller do
34
35    def new
36      params[:documents] = []
37      params[:images] = []
38      super
39    end
40
41    def edit
42      di = DocumentImage.find(params[:id])
43      params[:documents] = Document.where(country_insurance_id: di.country_insurance_id).collect { |d| d.name }
44      params[:images] = Image.where(country_insurance_id: di.country_insurance_id).collect { |d| d.name }
45      params[:selected_document] = di.document_name
46      params[:selected_image] = di.image_name
47      super
48    end
49
50    def create
51      super do |format|
52        format.html do
53          redirect_to admin_document_images_path
54        end
55      end
56    end
57  end
```

Figura 42. Extracto de código donde se configuran las fotos que muestra el sistema.

En la figura 36–43 se mostrara otro extracto del código donde se maneja la configuración de las aseguradoras y la asignación de data.

```
1 class ApiController < ApplicationController
2
3   def gateway_insurance
4     params.delete(:controller)
5     params.delete(:action)
6     unless params.include? :country_insurance_id or params.include? :gateway
7       render json: { error: "Missing required params: 'country_insurance_id' or 'gateway'", status: 400 }
8     else
9       gateway_insurance = GatewayInsurance.joins(:gateway).where("LOWER(gateways.name) LIKE
10        '%#{params[:gateway].downcase}%'").where(country_insurance_id: params[:country_insurance_id])[0]
11       if gateway_insurance.blank?
12         render json: { error: "Record not found: country_insurance_id => #{params[:country_insurance_id]},
13          gateway_name => #{params[:gateway]}" }, status: 400
14       else
15         gateway = gateway_insurance.gateway
16         render json: { gateway_insurance: gateway_insurance, gateway: gateway }, status: 200
17       end
18     end
19   end
20
21   def gateway_insurance_all
22     params.delete(:controller)
23     params.delete(:action)
24     unless params.include? :gateway
25       render json: { error: "Missing required params: 'gateway'", status: 400 }
26     else
27       gateway_insurances = GatewayInsurance.joins(:gateway).where("LOWER(gateways.name) LIKE '%#{params[:gateway].downcase}%'")
28       if gateway_insurances.blank?
29         render json: { error: "Record not found: gateway_name => #{params[:gateway]}" }, status: 400
30       else
31         render json: { gateway_insurances: gateway_insurances }, status: 200
32       end
33     end
34   end
35
36   def gateway_empresas
37     params.delete(:controller)
38     params.delete(:action)
39
40     json = JSON.parse(params[:json])
41
42     policy = json['idConcepto'].split('~')[0]
43     reference = json['idConcepto'].split('~')[1]
44     user = reference.split('.')[0]
45     ar = AccountReceivable.create!(policy_id: policy, user_id: user, date: Time.now, currency: json['moneda'], amount: json['valor'], paid: false)
46
47     json['codRespuesta'] = '00'
48     json['descRespuesta'] = 'Aprobado'
49     json['idTransaccion'] = ar.id
50     json['numAprobacion'] = SecureRandom.hex(5)
51     json['fechaTransaccion'] = Time.now
52
53     url_confirm = json['urlConfirmacion']
54     url_response = json['urlRespuesta']
55
56     post_confirm = RestClient.post(url_confirm, { json: json.to_json.to_s })
57     post_response = RestClient.post(url_response, { json: json.to_json.to_s })
58     render text: post_confirm, status: 200
59   end
60
61 end
```

Figura 43. Extracto de código donde se configura la asignación de data de la aseguradora.

#### 4.4.7.4 Pruebas

Para probar las funciones del administrador, se probaron cada una de ellas por medio de sus vistas, verificando si se obtenían los mismos resultados que anteriormente daba el sistema, y en efecto los resultados eran los mismos.

Se probó intentar acceder a las funcionalidades del administrador como usuario visitante, y el sistema impidió todos los intentos, redirigiendo la petición a la vista de acceso a través de login y password.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 44).

| Caso de prueba                                                      |                                                                            |
|---------------------------------------------------------------------|----------------------------------------------------------------------------|
| Número de caso de prueba: 9                                         | Número de historia de usuario: 13,14                                       |
| Nombre del caso de prueba: Manejo de credenciales en administrador. |                                                                            |
| Descripción                                                         | Al entrar en el administrador se debe autenticar el usuario en el sistema. |
| Condiciones de ejecución                                            | El administrador usara sus credenciales para ingresar al sistema           |
| Resultado esperado                                                  | El sistema debe generar correctamente los datos                            |
| Evaluación                                                          | Satisfactorio                                                              |

Figura 44. Caso de prueba 9

#### 4.4.8 Iteración 7

En esta iteración se definirá el módulo donde el encargado de revisar los reportes de siniestro deberá analizar el siniestro y determinar si lo aprueba, lo rechaza o si devuelve la solicitud al cliente para que este pueda corregir algún error que tenga en la solicitud y volverla a enviar.

Anteriormente los documentos se tenían que presentar al corredor de seguros y este se encargaba de enviárselo a la aseguradora para que esta aseguradora los revise y luego actué sobre el mismo.

Ahora la aplicación va a proveer a la aseguradora un sistema donde va a poder revisar los documentos que se envían y tomar decisiones al momento.

##### 4.4.8.1 Planificación

En la tabla 12 se muestra las historias de usuario que se desarrollaran en esta iteración.

Tabla 13. Historia de usuario de la iteracion 7.

| Id | Fecha      | Requerimiento                            | Tipo  |
|----|------------|------------------------------------------|-------|
| 13 | 08/04/2016 | Implementación del módulo de aseguradora | Nueva |

#### 4.4.8.2 Diseño.

En la figura 45 se mostrara el diseño de la interfaz del módulo de aseguradora. Se presentaron 2 cuadros donde en el primero se mostraron las últimas pólizas vendidas con su dueño, y en el otro los últimos siniestros reportados con su estado.

Aseguradora

Inicio

Clientes

Pólizas

Reclamos

Reportes

mpascale@gmail.com

Salir

Inicio

Ingreso exitoso

Últimas pólizas

| #PÓLIZA                         | EQUIPO ASEGURADO                    | ESTATUS |
|---------------------------------|-------------------------------------|---------|
| <a href="#">21-17-900000482</a> | Apple iPhone 4S 64GB                | PAGADA  |
| <a href="#">21-17-900000480</a> | Apple iPhone 6 Plus 128GB           | PAGADA  |
| <a href="#">21-17-900000479</a> | Apple iPhone 5C 8GB                 | PAGADA  |
| <a href="#">21-17-900000477</a> | Apple iPhone 6 Plus 16GB            | PAGADA  |
| <a href="#">21-17-900000476</a> | Apple iPhone 5C 8GB                 | PAGADA  |
| <a href="#">21-17-900000471</a> | LG Optimus L3                       | PAGADA  |
| <a href="#">21-17-900000470</a> | Motorola Moto G 2DA Generación 16GB | PAGADA  |
| <a href="#">21-17-900000469</a> | Huawei Ascend Mate 7                | PAGADA  |
| <a href="#">21-17-900000467</a> | Blackberry Q10                      | PAGADA  |
| <a href="#">21-17-900000464</a> | Apple iPhone 4S 16GB                | PAGADA  |

Últimos reclamos

| #RECLAMO            | #PÓLIZA                         | CLIENTE          | MOTIVO | ESTATUS            |
|---------------------|---------------------------------|------------------|--------|--------------------|
| <a href="#">135</a> | <a href="#">21-17-900000481</a> | Ricardo Ubieta   | Hurto  | FALTA PAGO         |
| <a href="#">134</a> | <a href="#">21-17-900000306</a> | Lenys Varguillas | Hurto  | FALTA PAGO         |
| <a href="#">133</a> | <a href="#">21-17-900000480</a> | Brian Guacaran   | Hurto  | FALTA PAGO         |
| <a href="#">132</a> | <a href="#">21-17-900000311</a> | Ricardo Ubieta   | Hurto  | RECLAMO EN ESTUDIO |
| <a href="#">131</a> | <a href="#">21-17-900000478</a> | Marco Pascale    | Hurto  | APROBADO           |
| <a href="#">125</a> | <a href="#">21-17-900000308</a> | Ricardo Ubieta   | Hurto  | APROBADO           |
| <a href="#">124</a> | <a href="#">21-17-900000475</a> | Ricardo Ubieta   | Daño   | FALTA PAGO         |
| <a href="#">123</a> | <a href="#">21-17-900000468</a> | Guacarans Inc    | Daño   | APROBADO           |
| <a href="#">122</a> | <a href="#">21-17-900000391</a> | Ries Inc         | Hurto  | DEVUELTO           |
| <a href="#">121</a> | <a href="#">21-17-900000415</a> | Ries Inc 2       | Hurto  | APROBADO           |

Powered by [Active Admin](#) 0.8.8

Figura 45. Interfaz del dashboard del administrador de la aseguradora.

En la figura 46 se muestra la interfaz del menú de los reportes a utilizar, estos reportes se enviaran por email un documento en formato pdf con el reporte seleccionado.

|                       |               |          |
|-----------------------|---------------|----------|
| ADMIN /               |               | Reportes |
| Seleccione el reporte |               |          |
| REPORTE DE VENTAS     | ✓ Seleccionar |          |
| REPORTE DE SINIESTROS | ✓ Seleccionar |          |
| REPORTE DE USUARIOS   | ✓ Seleccionar |          |

Figura 46. Interfaz de la sección de reportes.

#### 4.4.8.3 Codificación

La implementación de la página principal del módulo de aseguradora puede verse en el fragmento de código de la figura 47.

```
1 # encoding : utf-8
2 ActiveAdmin.register Claim do
3
4   config.batch_actions = false
5
6   actions :show, :index, :update
7
8   statuses = [{"Falta planilla de Reclamo", "incomplete"}, {"En Revision", "pre_aproved"}, {"Falta pago", "aproved_pd"}, {"Aprobado", "aproved"},
9   [{"Finalizado", "closed"}, {"Cancelado", "canceled"}]
10  statuses << ["Devuelto", "returned"]
11  statuses << ["Rechazado", "rejected"]
12  filter :status, as: :select, label: "Buscar estatus", collection: statuses
13  filter :email_contains, as: :string, label: 'Email'
14  filter :phone_number_contains, as: :string, label: 'Teléfono'
15
16  index download_links: [:csv] do
17    column :id
18    column :status do |c|
19      case c.status
20      when "incomplete"
21        status_tag("Falta planilla de Reclamo", class: "incomplete border_tag")
22      when "pre_aproved"
23        status_tag("En revision", class: "pre_aproved border_tag")
24      when "aproved_pd"
25        status_tag("Falta pago", class: "aproved_pd border_tag")
26      when "aproved"
27        status_tag("Aprobado", class: "aproved border_tag")
28      when "closed"
29        status_tag("Finalizado", class: "close border_tag")
30      when "canceled"
31        status_tag("Cancelado", class: "close border_tag")
32      when "returned"
33        status_tag("Devuelto", class: "close border_tag")
34      else
35        status_tag("Rechazado", class: "rejected border_tag")
36      end
37    end
38    column(:created) { |c| c.created.strftime('%d/%m/%Y - %I:%M %p') }
39    column(:event_date) { |c| c.event_date.to_date.strftime('%d/%m/%Y') unless c.event_date.nil? rescue "" }
40    column :ctype do |c|
41      case c.ctype
42      when "stolen"
43        "Hurto"
44      else
45        "Daño"
46      end
47    end
48  end
49  default_actions
50 end
```

Figura 47. Implementación página principal del módulo de aseguradora.

En la figura 48 se puede ver un extracto del código que genera los reportes de la aplicación.

```
1 # encoding: UTF-8
2 ActiveAdmin.register_page 'reports' do
3
4   menu label: 'Reportes'
5
6   content title: 'Reportes' do
7     unless params[:insurance_id].blank?
8       insurance = CountryInsurance.find(params[:insurance_id])
9       panel "Seleccione el reporte (#{insurance.name})", id: 'reportes' do
10        table id: 'reportes' do
11          if insurance.name.downcase.include? 'estado'
12            tr do
13              th 'Reporte MUP'
14              th { link_to "#{icon(:check)} Seleccionar".html_safe, admin_reports_sde_mups_path(insurance_id: insurance.id), class: 'btn-white icon' }
15            end
16            tr do
17              th 'Reporte de Recaudos'
18              th { link_to "#{icon(:check)} Seleccionar".html_safe, admin_reports_sde_recaudos_path(insurance_id: insurance.id), class: 'btn-white icon' }
19            end
20            tr do
21              th 'Reporte de Ventas'
22              th { link_to "#{icon(:check)} Seleccionar".html_safe, admin_reports_sde_sales_path(insurance_id: insurance.id), class: 'btn-white icon' }
23            end
24          else
25            tr do
26              th 'Reporte de Ventas'
27              th { link_to "#{icon(:check)} Seleccionar".html_safe, admin_reports_ace_sales_path(insurance_id: insurance.id), class: 'btn-white icon' }
28            end
29          end
30        end
31      end
32    else
33      panel 'Seleccione la aseguradora', id: 'reportes' do
34        table id: 'reportes' do
35          CountryInsurance.all.each do |ci|
36            tr do
37              th ci.name
38              th { link_to "#{icon(:check)} Seleccionar".html_safe, url_for(insurance_id: ci.id), class: 'btn-white icon' }
39            end
40          end
41        end
42      end
43    end
44
45    span link_to "#{icon(:arrow_left)} Atrás".html_safe, nil, class: 'btn-white icon' if !params[:insurance_id].blank? and current_ace_user.country_insurances.size > 1
46  end
47
48  controller do
49
50    def index
51      insurances = current_ace_user.country_insurances
52      unless insurances.size > 1
53        params[:insurance_id] = insurances[0]
54      end
55      super
56    end
57  end
58
59 end
60 end
```

Figura 48. Extracto de código de la generación de reportes.

En la figura 49 se puede ver un extracto del código que implementa la vista para revisar los siniestros. Mediante jquery se validó a validar la lógica que debería tener el analista para que cumpla todos los requisitos a la hora de aprobarlo o rechazarlo.

**Reclamo #38**

▼ Reclamo

|                            |                                                                                                                                                       |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| ID                         | 38                                                                                                                                                    |
| DESCRIPCIÓN                | El día viernes 10 de Febrero de 2016 me encontraba en la ciudad de Caracas de camino a mi casa y se me acerco un sujeto en bicicleta con un cuchillo. |
| CÉDULA                     | Ver                                                                                                                                                   |
| DENUNCIA POLICIAL          | Ver                                                                                                                                                   |
| FACTURA                    | Ver                                                                                                                                                   |
| RECLAMO                    | Ver                                                                                                                                                   |
| ESTATUS                    | APROBADO                                                                                                                                              |
| FECHA DE RECLAMO           | jueves, 16 de Octubre de 2014 a las 05:39PM                                                                                                           |
| #FACTURA                   | 336276766                                                                                                                                             |
| FECHA DE COMPRA DEL EQUIPO | 10-06-2014                                                                                                                                            |
| FECHA DE LA OCURRENCIA     | 10-10-2014                                                                                                                                            |
| TIPO DE RECLAMO            | Hurto                                                                                                                                                 |
| OBSERVACIONES              | VACÍO                                                                                                                                                 |

► Poliza

► Usuario

► Equipo

Figura 49. Interfaz de análisis de siniestros.

**4.4.8.4 Pruebas**

Para realizar las pruebas para el módulo de aseguradora, se colocaron varios reclamos y se revisó el comportamiento de la aplicación ante cada uno, se aprobaron siniestros, se negaron siniestros y en cada caso se revisó la data de la aplicación para verificar que fuera correcta.

A continuación se mostraran algunas de las pruebas realizadas en esta iteración (Ver figuras 50, 51).

| Caso de prueba                                                                         |                                                                                              |
|----------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| Número de caso de prueba: 10                                                           | Número de historia de usuario: 15,16                                                         |
| Nombre del caso de prueba: Manejo de credenciales del analista en el modulo asegurador |                                                                                              |
| Descripción                                                                            | Al entrar en el administrador de la aseguradora se debe autenticar el usuario en el sistema. |
| Condiciones de ejecución                                                               | El cliente colocara un reclamo, y luego subirá sus documentos a la plataforma                |
| Resultado esperado                                                                     | El sistema debe generar correctamente los datos                                              |
| Evaluación                                                                             | Satisfactorio                                                                                |

Figura 50. Caso de prueba 10

| Caso de prueba                                                           |                                                                                                           |
|--------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| Número de caso de prueba: 11                                             | Número de historia de usuario: 15,16                                                                      |
| Nombre del caso de prueba: Analisis de reclamos en el modulo asegurador. |                                                                                                           |
| Descripción                                                              | El analista asegurador debe revisar la documentacion entregada por el cliente en su reporte de siniestro. |
| Condiciones de ejecución                                                 | Debe revisar todos los documentos enviados, y colocar la aprobacion de los mismos.                        |
| Resultado esperado                                                       | El sistema mostrara los documentos y luego al indemnizar cambiara el estado del mismo.                    |
| Evaluación                                                               | Satisfactorio                                                                                             |

Figura 51. Caso de prueba 11

## **Conclusiones**

El presente TEG tuvo como finalidad el desarrollo de una aplicación Web para la cotización, compra y manejo de pólizas de seguros de celulares, Smartphone y tabletas en Venezuela para lograr mejoras en el proceso de negocio y en la generación de documentos.

Se engloban las funcionalidades desarrolladas y se presentan como un modelo informático en el cual permite la reutilización de código, por ejemplo los documentos que pueden ser emitidos por la compañía de seguros, lo que implica una mejora para el sector asegurador, ya que los procesos manuales pasan a ser procesos ejecutados de manera online, destacando aspectos como lo son la rapidez y robustez que se desean con un sistema.

Se logró obtener un producto que cumple con los requerimientos planteados por los usuarios al principio de este trabajo, así como también las modificaciones requeridas a lo largo del desarrollo del mismo.

La metodología ágil XP facilitó considerablemente el desarrollo de la aplicación, gracias a la flexibilidad que provee para adaptarse a los cambios, así como la posibilidad de mantenerse en contacto con los clientes constantemente, permitiendo la retroalimentación y destacando el desarrollo en pareja como bien lo dicen los fundamentos de la metodología.

Gracias a las pruebas realizadas en cada iteración se verificaron los errores y fueron corregidos inmediatamente.

Este proyecto forma parte de la nueva manera de llevar los procesos en el ramo asegurador específicamente orientado a celulares, Smartphone y tabletas, y gracias a esta aplicación web se logró llevar a cabo la automatización de los procesos de cotización, compra y administración de pólizas de una manera más sencilla. Adicionalmente, el flujo de trabajo tanto de estos procesos como los de manejo de siniestros ha sido estandarizado, pudiéndose aplicar a otros servicios de pólizas ofrecidos, logrando que las actividades diarias sean más sencillas, con una reducción de tiempo y esfuerzo.

Por otro lado, se ha contribuido en la disminución de errores cometidos durante la realización de las tareas involucradas en los procesos abarcados, ya que la obtención y manejo de los datos se realiza a través de la Aplicación Web, a través de la cual se

valida su registro correcto, alcanzando una mayor satisfacción por parte de los involucrados, reduciendo el tiempo invertido en la realización de estos procesos y los costos asociados al material impreso utilizado.

## **Trabajos Futuros**

Es importante destacar que el sistema elaborado es un producto escalable, es por esto que se pueden

Agregar nuevas funcionalidades, entre las cuales se encuentran:

- A raíz de la implementación de esta aplicación donde se abarcan funcionalidades del lado del cliente y del lado de los administradores se podrían generar datos estadísticos referentes a la empresa con el fin de facilitar la toma de decisiones logrando una mejor comprensión del funcionamiento de la empresa.
- implementar y dar soporte a una versión móvil nativa de la aplicación.
- Implementar y dar soporte al envío y recepción de mensajes de texto.
- Módulos de auditoria de la aplicación.
- Hacer pruebas de seguridad y vulnerabilidad tanto del lado del servidor, como del lado del cliente.

## Referencias bibliográficas

- Lujan, S. (2002). Programación de aplicaciones web: historia, principios básicos y clientes web: Editorial Club Universitario.
- Valbuena, S., Cardona, S. y Villa, D. (2008) Programación Avanzada en Java. Colombia: Ediciones Elizcom.
- EcuRed, Framework, Recuperado de <http://www.ecured.cu/index.php/Framework>
- Mendoza, I., Definición de un Framework para aplicaciones Web con Navegación sensibles a concerns, Recuperado de [http://sedici.unlp.edu.ar/bitstream/handle/10915/4192/Documento\\_completo.pdf?sequence=1](http://sedici.unlp.edu.ar/bitstream/handle/10915/4192/Documento_completo.pdf?sequence=1)
- Diseño y Programación de Bases de Datos (2009) Ángel Cobo Yera
- Llanos, D. (2007). Fundamentos de informática y programación en C. España: Editorials S.A. EDICIONES PARANINFO.
- Galindez, A. (2010). Fundamentos de Bases de Datos: Instituto Tecnológico de Colima.
- HTML & CSS. (s.f.), Recuperado de <http://www.w3.org/standards/webdesign/htmlcss>.
- CSS|MDN.(s.f). Recuperado de <https://developer.mozilla.org/es/docs/Web/CSS>.
- Brito, K. (2009). Selección de Metodologías de Desarrollo para Aplicaciones Web en la Facultad de Informática de la Universidad de Cienfuegos, Edición electrónica gratuita. Texto completo en [www.eumed.net/libros/2009c/584/JavaScript, Mozilla Developer,\(s.f.\).](http://www.eumed.net/libros/2009c/584/JavaScript_Mozilla_Developer_(s.f.).) Recuperado de <https://developer.mozilla.org/es/docs/Web/JavaScript>
- jQuery.(s.f.). Recuperado de <http://jquery.com/>
- Bootstrap .The world's most popular mobile-first and responsive front-end framework. (s.f.). Recuperado de <http://getbootstrap.com/>
- Santos, D. (13 de Octubre de 2006). Maestro del Web. Recuperado de <http://www.maestrosdelweb.com/editorial/rubyonrails/>
- Coronel, C., Morris, S., & Rob, P. (s.f.). Bases de Datos, Diseño, Implementación y Administración.
- Valencia. (10 de Abril de 2008). Hooping. Recuperado de <http://www.hooping.net/glossary/aplicaciones-web-146.aspx>
- Grunt: The JavaScript TaskRunner. (s.f.)., Recuperado de <http://gruntjs.com/>

- PHP: Whatis PHP?. (s.f.). Recuperado de <http://php.net/manual/en/intro-what-is.php>
- HTTP Server Project. (s.f.). Apache, Recuperado de <http://httpd.apache.org/>
- Git, Wikipedia.(s.f.). Recuperado de <http://es.wikipedia.org/wiki/Git>
- PHPStorm, Wikipedia, Recuperado de <https://www.jetbrains.com/phpstorm/>
- BizagiProcessModelerUser'sGuide (Recuperado el 2 de Octubre del 2014) <http://help.bizagi.com/processmodeler/es/> 27. Sobre PostgreSQL, Recuperado de [http://www.postgresql.org.es/sobre\\_postgresql](http://www.postgresql.org.es/sobre_postgresql)
- Metodologías para el desarrollo de software.(s.f.). Recuperado de [http://wiki.monagas.udo.edu.ve/index.php/Metodolog%C3%ADas\\_para\\_el\\_desarrollo\\_de\\_software](http://wiki.monagas.udo.edu.ve/index.php/Metodolog%C3%ADas_para_el_desarrollo_de_software) 29.
- Selección de Metodologías de Desarrollo Para Aplicaciones Web .(s.f.). Recuperado de <http://www.eumed.net/librosgratis/2009c/584/Por%20que%20utilizar%20Scrum%20para%20desarrollar%20aplicaciones%20web.htm>
- Definición de API.(s.f.). Recuperado de <http://www.alegsa.com.ar/Dic/api.php>
- CSS mediaqueries. Mozilla Developer Network.(s.f.). Recuperado de [https://developer.mozilla.org/es/docs/CSS/Media\\_queries](https://developer.mozilla.org/es/docs/CSS/Media_queries)
- C. J. DATE. (2001). introducción a los sistemas de bases de datos (7ª ED.). Edo de México: S.A. alhambra mexicana.
- Alegsa, L. (2010). <http://www.alegsa.com.ar/Dic/aplicacion20web.php>. Argentina.
- Alvarez, M. A. (2008 йил 29-Junio). From <http://www.desarrolloweb.com/articulos/introduccion-css3.html>
- Alvarez, M. A. (2009). From <http://www.desarrolloweb.com/articulos/25.php>
- Alvarez, M. A. (2009 йил 19-Junio). *Desarrollo web*. From <http://www.desarrolloweb.com/articulos/introduccion-jquery.html>
- Alvarez, M. A. (2010). *Desarrollo Web*.
- Hamano. (2008). <http://picandocodigo.net>.
- John, T. O. (30 de 9 de 2005). <http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html>. Recuperado el 14 de junio de 2016, de <http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html>
- Nerfites. (2009 йил 19-Obtubre). *Autorneto*. From <http://autorneto.com/tecnologia/programacion/javascript-asincrono/>

- PCMAG. (2003). From [http://www.pcmag.com/encyclopedia\\_term/0,2542,t=Web+client&i=54284,00.asp](http://www.pcmag.com/encyclopedia_term/0,2542,t=Web+client&i=54284,00.asp)
- Piñol, C. M. (2003). *Desarrollo de aplicaciones web*. UOC.
- Rafael, J. (2008 йил 22-Junio). *Neleste 2.0*. From <http://www.neleste.com/acerca-de/>
- Rodríguez, H. A. (2006 йил 24-febrero). *MailxMail*. From <http://www.mailxmail.com/curso-procesamiento-datos-oracle/sistema-manejador-base-datos>
- San Antonio, M. C. (1991). *El sector Asegurador. Escuela de Hacienda Pública*. España.
- Tanenbaum, A. (2002). *Distributed Systems*. Prentice Hall.