UNIVERSIDAD PÚBLICA DE EL ALTO

CARRERA INGENIERÍA DE SISTEMAS

PROYECTO DE GRADO

SISTEMA DE INFORMACION WEB, PARA EL SEGURO SOCIAL UNIVERSITARIO.

CASO: UPEA

Para optar al título de Licenciatura en Ingeniería de Sistemas

Mención: INFORMATICA Y COMUNICACIONES

Postulante: **Filberto Mamani Ato**

Tutor Metodológico: **Ing. Marisol Arquedas Balladares**

Tutor Especialista: **Lic. Freddy Salguiero Trujillo**

Tutor Revisor: **Ing. Fanny Helen Pérez Mamani**

EL ALTO – BOLIVIA

2020

DEDICATORIA:

Dedico este proyecto a mi familia, por el constante cooperación y apoyo incondicional en el logro de mis metas personales y profesionales.

AGRADECIMIENTOS:

A la Universidad Pública De El Alto, a la Carrera de Ingeniería de Sistemas y a todos mis docentes por los valiosos conocimientos que me impartieron y que fueron la base para la realización del proyecto.

A la Ing. Marisol Arguedas Balladares, a la Ing. Ing. Fanny Helen Pérez Mamani y al Lic. Freddy Salguiero Trujillo, por su apoyo y asesoramiento en la realización del presente trabajo.

Al Dr. Cesar Hugo Suxo Escobar Jefe de Unidad de Seguro Social (UPEA), por brindarme su preciado tiempo y facilitarme la información necesaria para mi proyecto.

A mi querida familia por guiarme, ayudarme, y estar a mi lado en cada momento de mi vida y especialmente a lo largo de mi formación profesional.

RESUMEN

Tener la información actualizada y en el momento oportuno es una necesidad primordial en todas las empresas para que puedan cumplir con sus objetivos, dicha información puede ser manipulada desde cualquier dispositivo y cualquier explorador que tenga una conexión a internet.

El proyecto “SISTEMA DE INFORMACION WEB, PARA EL SEGURO SOCIAL UNIVERSITARIO. Caso: UPEA”, tiene como principal objetivo ayudar al personal de la unidad en la obtención y manipulación de información de los estudiantes.

En el desarrollo del sistema web se usa la metodología UWE.

UWE detalla el proceso de las aplicaciones, cuenta con cinco modelos que son el análisis de requerimientos, modelo de contenidos, modelo navegacional, modelo de presentación y modelo de procesos.

Para la calidad del sistema web se utiliza la norma ISO 9126 y para la seguridad del sistema se realiza por niveles, tanto del lado del cliente como del lado del servidor.

ÍNDICE DE CONTENIDO

Pág.

Contenido

CAPITULO I

1.1 INTRODUCCIÓN

En la actualidad los se han establecido como factores cruciales para la toma de decisiones, así como para la aceleración de procesos en distintas entidades, logrando que sus tareas sean más rápidos y eficientes en el manejo de grandes cantidades de datos e información, ya que las tareas que tradicionalmente eran realizadas manualmente antes, ahora son realizadas por medio de sistemas de información.

En toda institución es importante tener un control fiable de los datos e información ya que este constituye el eje fundamental que mueve a todos los sistemas, además es esencial para el logro y alcance de los propósitos, las instituciones de carácter educativo no son la excepción, ya que reciben una creciente cantidad de nuevos estudiantes cada gestión por lo cual se debe presentar especial atención en el manejo de datos.

La Universidad Pública de El Alto está experimentando un crecimiento significativo en la cantidad de estudiantes matriculados en sus distintas carreras, los mismos requieren nuevas formas de atención a sus requerimientos en cuanto a los procesos de carácter académico, infraestructura y otros.

El presente trabajo hace énfasis en la Dirección de Seguro Social, es uno de los departamentos con alta responsabilidad de la Universidad Pública de El Alto, es encargado de administrar la afiliación al seguro universitario al estamento estudiantil, en ese marco el proyecto tiene por objeto el diseño, desarrollo de un sistema de registro, para mejorar el manejo de datos que permitirá tomar decisiones estratégicas.

1.2 ANTECEDENTES

1.2.1 Antecedentes de la Institución

En 1989 instituciones sociales de El Alto firmaron convenios con la Universidad Mayor de San Andrés (UMSA) para crear una facultad con carreras técnicas. La población de El Alto quería que la universidad contase con carreras de formación

profesional, no sólo a nivel técnico. Para esto, los habitantes de esta ciudad iniciaron una serie de movilizaciones para conseguir una universidad que tenga una mayor oferta académica.

Los pobladores y juntas sociales de El Alto no consideraban pertinente tener una universidad que respondiera a gobierno y administración de otra ciudad. Además, las organizaciones sociales solicitaban carreras como medicina y la Universidad Mayor de San Andrés (UMSA) sólo proponía carreras técnicas, a lo cual respondieron con manifestaciones solicitando una universidad pública y autónoma.

Creación

El 5 de septiembre de 2000 se promulgó la ley 2115 que determinó la creación de la Universidad Pública de El Alto, la misma determina que la UPEA tendría autonomía en 5 años, tiempo durante el cual estaría a cargo de un consejo formado por el Ministerio de Educación de Bolivia y otros organismos gubernamentales. De acuerdo a la ley de su creación, el ente de mayor decisión en la universidad sería el Consejo de Desarrollo Institucional (CDI), mismo en el que estaban insertos miembros de organizaciones sociales que tenían poca relación con el quehacer académico. Sin embargo, la UPEA comenzó a institucionalizarse, se reinstauró el Consejo Universitario presidido por Edwin Callejas, luego de deponer a Javier Tito nombrado por el ministerio de educación y haciéndose cabeza del gobierno de la UPEA.

Autonomía

En noviembre de 2003 durante el gobierno de Carlos Mesa se pone en vigencia la ley que garantiza la autonomía universitaria de la UPEA.2 La universidad ha sido un actor principal de las revueltas sociales durante los últimos años.

La dirección de Seguro Social, es un área autónoma de la Universidad Pública de el Alto, encargada de gestionar toda la información.

Para el desarrollo del presente estudio se tomaron los siguientes antecedentes:

1.2.2 Antecedentes

Para el desarrollo del presente estudio se tomaron los siguientes antecedentes:

1.2.3 Internacional

Título: Sistema de información para la gestión de proyectos.

Año: Bogotá, agosto 2016, fundación universitaria los libertadores.

Autor: Paola Andrea Blanco Y Mauricio Hernández.

Metodología: SCRUM.

Herramientas: Microsoft Visual Studio 2015 ASP.NET, Internet Information Services, el lenguaje de modelado unificado o UML, Y como base datos SQL Server 2012.

Descripción del proyecto: Se desarrolló un sistema confiable y estable basado en programación web bajo tecnología asp.net; el cual tiene administración y seguridad que permita gestionar los diferentes roles que se van a manejen en el sistema, dando la opción a los profesores y directivos de la institución tener un control sobre los proyectos de grado y las investigaciones desarrolladas por los diferentes alumnos.

Título: Sistema de información para el registro y control de averiguaciones disciplinarias de la defensa pública.

Año: Venezuela, caracas mayo de 2012. universidad de Católica Andrés Bello

Autor: por Ing. Leonardo David Gomes.

Metodología: XP (extreme programing).

Herramientas: son HTML, RUBY, UML, ruby on rails, y como base de datos posygreSQL.

Descripción del proyecto: El sistema es orientado en colaborar en la solución de la problemática que se genera en procesos judiciales.

Título: sistema de información para la gestión de los trabajos de grado

Año: Cartagena de Indias, 2013, Universidad de Cartagena de indias

Autor: Marcelo Solís y Julio Cesar Deavila.

Metodología: metodología de desarrollo RUP.

Herramientas: PHP, base de datos MySQL, servidor Web apache HTTP 2.2 (opensource), entorno de desarrollo Adobe Dreamweaver CS6 Versión 12.0 Build 5808 y Navicat Premium Versión 9.1.11.

Descripción del proyecto: Se realizó con el fin de poner una solución a los problemas que presenta el proceso de entrega y evaluación de trabajos de grado realizados por los estudiantes próximos a graduarse.

1.2.4 Nacional:

Título: “sistema de gestión hospitalario modulo consulta externa caso: seguro social universitario”

Año: La Paz, Bolivia 2013, universidad mayor de san Andrés facultad de ciencias puras y naturales.

Autor: Univ. Marco Aurelio avalos

Metodología: metodología de desarrollo RUP

Herramientas: SCRUM, motor base de datos SQL server, servidor web apache, xampp, lenguaje de programación php, lenguaje de modelado unificado (UML).

Descripción del proyecto: El proyecto se realizó para brindar, al médico la sistematización en la redacción de diagnósticos y tratamientos a los pacientes.

1.2.5 Local:

Título: sistema de información automatizado para el control de la gestión y administración de bienes y servicios

Año: el Alto Bolivia de 2013 UPEA.

Autor: Walter Emilio Paco Siles

Metodología: metodología web que es UWE (UML-Based WEB Engineering)

Herramientas: MagigDraw, JQuery, php5, servidor apache, PostgresSQL.

Descripción del proyecto: El proyecto está desarrollado para funcionar en una arquitectura cliente servidor lo que significa que todas las gestiones que se realizan se concentran en el servidor de manera que en él se disponen los requerimientos provenientes de los usuarios.

1.3 PLANTEAMIENTO DEL PROBLEMA

La Universidad Pública De El Alto, en la dirección de Seguro Social, confronta una diversidad de problemas en su proceso de control y seguimiento de datos, por falta de información actualizada, ya que la misma área se encuentra en una etapa de implementación.

También el incremento de la población estudiantil hace que la información sea excesiva y difícil de manejar adecuadamente.

Por tanto, todo esto deriva a una ineficiencia en el control y administración del estamento estudiantil por parte de la Dirección del Seguro Social.

1.3.1 Problema General

La Dirección de seguro Social, no cuenta con algún sistema de información para el registro de los estudiantes, específicamente en esta área, lo que dificulta a la hora de tomar decisiones.

1.3.2 Problemas Específicos

- El almacenamiento de la información es descentralizado lo que genera redundancia y demora en el manejo de datos.
- Los reportes que se emiten de los procesos de seguimiento se lo hacen en forma manual lo cual demanda mucho tiempo.
- El manejo de datos e información está sujeto a ser eliminada o perdida puesto que los mismos son manejadas de forma manual.
- La información y el registro de los estudiantes no es actual ni oportuna y dificulta a la toma de decisiones.

De acuerdo a estos problemas surge la siguiente pregunta o interrogante.

¿de qué forma se podría mejorar el registro de los beneficiarios del seguro universitario?

1.4 OBJETIVOS

1.4.1 Objetivo General

Desarrollar un sistema Web de registro y acceso de servicios, de los estudiantes al seguro universitario, para la Dirección de seguro de salud de la Universidad Pública de El Alto, a fin de lograr una eficiente administración en el manejo y gestión de la información.

1.4.2 Objetivos Específicos

- Diagnosticar la situación actual de la Dirección de seguro social.
- Realizar el análisis del sistema para comprender las necesidades.
- Crear una base de datos.
- Implementar el diseño del sistema.
- Diseñar y desarrollar la plataforma en base a requerimientos obtenidos.
- Registrar información de cada uno de los estudiantes que gozan del beneficio de seguro social.
- Registrar información de los parentescos de cada uno de los estudiantes.
- Reportar incidentes.
- Generar reportes que ayuden a la toma de decisiones.

1.5 JUSTIFICACIÓN

1.5.1 Justificación Técnica

La Dirección de seguro Social de la Universidad Pública de el Alto cuenta con recursos y herramientas como equipos de computación con buena capacidad en memoria, procesamiento, conexiones de red, servicio de internet y ambiente adecuado, para poder llevar a cabo el desarrollo del sistema. En la parte de software se tienen instalado el sistema operativo Windows, con programas como: Office 2007, PDF, antivirus y otros programas.

El sistema de administración le permitirá a la Dirección de Seguro Social, a contar con un recurso tecnológico que facilite el desempeño de sus actividades en el manejo de la información, sirviendo así de una herramienta útil y fiable.

1.5.2 Justificación Económica

La información es un recurso de consideración en toda universidad, más aún si esta es manipulada en grandes volúmenes de datos, en la medida, en que las mismas sean automatizadas implica ahorro en tiempo, reducción en material de escritorio, minimiza el trabajo manual.

El sistema será desarrollado bajo la premisa de software libre, que implica la no erogación de gastos en licencias de uso.

1.5.3 Justificación Social

La dirección de seguro Social, recibe cada año a nuevos postulantes para el seguro universitario, los datos de los mismo son manipulados en forma manual, almacenadas en material de escritorio, hojas de cálculo, en ese sentido el sistema que se propone coadyuvar con la dirección de seguro social, mostrando confiabilidad en el manejo de datos e información, reducirá el trabajo excesivo del personal, reducirá los tiempos de respuesta en la emisión de certificados, consultas, reportes, brindando a la comunidad universitaria del seguro social, una adecuada atención.

1.6 METODOS Y TECNICAS

Las metodologías y técnicas que se utilizaran para el desarrollo del presente proyecto son los siguientes:

1.6.1 Método Científico

Debido a que el presente trabajo tiene carácter formal, se utilizara el método científico durante todo el desarrollo del mismo, desde el planteamiento del problema hasta la verificación de los resultados obtenidos.

1.6.2 Metodología de Desarrollo

Para el desarrollo del software se hará uso de la metodología UWE (UML-Based Web Engineering, en español Ingeniería Web Basada en UML) es una metodología que permite modelar de mejor manera una aplicación Web, para el proceso de creación de aplicaciones detalla ésta, con una gran cantidad de definiciones, en el proceso de diseño lista que debe utilizarse. Procede de manera iterativa e incremental, coincidiendo con UML, incluyendo flujos de trabajo y puntos de control.

Modelo de aplicación web según la metodología UWE

- Modelo de Casos de Uso: se modela requisitos funcionales de la aplicación Web para ver como interactúa cada uno de ellos.

- Modelo Conceptual: Materializa en un modelo de dominio, considerando los requisitos reflejados en los casos de uso.
- Modelo Navegación: Especifica el entorno en la cual se realizará el aspecto de navegación de la aplicación Web.
- Modelo de presentación: Representa las vistas del interfaz del usuario mediante modelos estándares de interacción UML.

En cuanto a los requisitos, UWE los clasifica dependiendo del carácter de cada uno. Además, distingue entre las fases de captura definición y validación de requisitos (Engineering, 2012).

1.6.3 Métricas de Calidad de Software

Se evaluará la calidad del software usando la ISO 9126.

ISO 9126 es un estándar internacional para la evaluación del Software, fue originalmente desarrollado en 1991 para proporcionar un esquema para la evaluación de calidad del software (CUATRORIOS, 2011).

La normativa define seis características de la aplicación, estas seis características son dividas en un número de sub- características, las cuales representan un modelo detallado para la evaluación de cualquier sistema informático las cuales son:

- ✓ Funcionalidad: Adecuación, Exactitud, Interoperatvidad y Seguridad
- ✓ Confiabilidad: Madurez, Tolerancia a fallas y Recuperabilidad
- ✓ Usabilidad: Entendibilidad, Capacidad de Aprendizaje y Operabilidad
- ✓ Eficiencia: Comportamiento en tiempos y Comportamiento de recursos
- ✓ Mantenibilidad: Analizabilidad, Modificabilidad, Estabilidad y Capacidad de prueba
- ✓ Portabilidad: Adaptabilidad, Instalabilidad y Remplazabilidad

1.6.4 Métodos de Estimación de Costos

Para la estimación de costos del software se aplicará el Modelo COCOMO.

El modelo COCOMO original se convirtió en uno de los modelos de estimación más ampliamente utilizados y estudiados en el mundo (Pressman R. S., 2010).

Está compuesto por tres modelos que corresponden a distintos niveles de detalle y precisión. Mencionados en orden creciente son: Modelo Básico, Intermedio y Detallado.

La estimación es más precisa a medida que se toman en cuenta mayor cantidad de factores que influyen en el desarrollo de un producto de software

1.6.5 Técnicas de Recopilación de Datos

En el desarrollo del presente proyecto, se utiliza las técnicas de: observación, entrevista, cuestionarios y encuestas a los usuarios finales lo que nos permite de forma directa, obtener la información necesaria concerniente al movimiento y flujo de información en la unidad de Seguro Social.

1.7 HERRAMIENTAS

1.7.1 MagicDraw 18.0

MagicDraw, herramienta CASE desarrollada por No Magic. Es compatible con el estándar UML 2.3, desarrollo de código para diversos lenguajes de programación (Java, C++ y C#, entre otros) así como para modelar datos. Cuenta con capacidad para trabajar en equipo y es compatible con varios entornos de desarrollo (IDEs) Diseñada para los analistas del negocio, los analistas del software, los programadores, los ingenieros y los escritores de la documentación.

Facilita el análisis y el diseño de los sistemas Orientado a Objetos (OO) y de las bases de datos orientados objeto (EcuRed, 2007).

Disponible en magicdraw.com, permite el diseño UML de las aplicaciones Web, de manera estandarizada.

1.7.2 JQuery

JQuery es un Framework JavaScript, nos ofrece una infraestructura con la que tendremos mucha mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Por ejemplo, con JQuery obtendremos ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax, etc. Cuando programemos JavaScript con JQuery tendremos a nuestra disposición una interfaz para programación que nos permitirá hacer cosas con el navegador que estemos seguros que funcionarán para todos nuestros visitantes. Simplemente debemos conocer las librerías del Framework y programar utilizando las clases y métodos para la consecución de nuestros objetivos (Miguel Angel, 2012).

1.7.3 php7

PHP, es un lenguaje de programación interpretado de alto nivel al lado del servidor para internet, muy similar en su sintaxis al lenguaje C, con algunas diferencias, no compila como al igual que C, ya que es un intérprete, por lo tanto, cada vez que se debe ejecutar un programa, lo interpreta verificando toda su sintaxis.

PHP nos brinda la posibilidad de realizar tareas de forma automatizadas, mejorando la productividad de nuestro sitio web y dando la posibilidad de añadir gran cantidad de funcionalidades que con HTML no podemos hacerlo, ya que HTML no es un lenguaje de programación.

PHP7 mejorara los mecanismos de POO para solucionar las carencias de las anteriores versiones. Un paso necesario para Conseguir que PHP sea un lenguaje apto para todo tipo de aplicaciones y entornos, incluso los más exigentes (Alvarez, López, & Gutierrez, 2013).

1.7.3 Framework CodeIgniter 3.11

CodeIgniter es un **framework** para el desarrollo de aplicaciones en php, que utiliza el MVC. Esto permite a los programadores o desarrolladores Web mejorar su forma de trabajar, además de dar una mayor velocidad a la hora de crear páginas Webs.

1.7.4 Servidor Apache

Apache es el Servidor Web hecho por excelencia por su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa. La licencia es una descendiente de la licencia BSD (*Berkeley Software Distribution*), no es GPL (General Public License**)**. Esta licencia permite hacer lo que quieras con el código fuente siempre que les reconozcas su trabajo.

Apache es una tecnología gratuita de código fuente abierta. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto.

Apache permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor las características de Apache se pueden extender hasta donde nuestra imaginación y conocimientos lleguen (Cooper, 2004).

1.7.4 Bases de datos

Tabla 1.2: Cuadro comparativo de gestores de base de datos.

Oracle

Características	Ventajas	desventajas
• El SGBD Oracle, es fabricado por Oracle Corporation. • Utiliza la arquitectura cliente/servidor.	• Gestor de base de datos completo. • Multiplataforma.	• Privado. • Costes de licencia y mantenimiento muy alto.

MySQL

Características	Ventajas	desventajas
• MySQL es un manejador de base de datos principalmente utilizado en las aplicaciones web en conjunto con php.	• Multiplataforma. • Usa licencia GLP. • Velocidad al realizar operaciones.	• Un gran porcentaje de las utilidades no están documentadas. • No es intuitivo, como otros programas (ACCESS).

MariaDB

Características	Ventajas	desventajas
• MariaDB nace como una alternativa libre a MySQL tras la compra de está por Oracle.	• En general, muchas mejoras para aumentar el rendimiento y la eficiencia con respecto a MySQL.	• La migración de un sistema muy testado y fiable como es MySQL.

Fuente [elaboración propia]

En base a la comparación se opta por el gestor de base de datos **MariaDB.**

MariaDB: Es un derivado del sistema de gestión de base de datos MySQL. El propósito de este sistema es proveer capacidades similares y extendidas en relación a MySQL, con un foco especia en mantener el software de forma libre a través de la licencia GPL. Con aplicaciones muy amplias, puede ser usado por equipos de desarrollo de software, testing de aplicaciones, sitios web de empresas y bloggers, servidores vps o locales y de hosting compartido.

1.8 LÍMITES Y ALCANCES

1.8.1 Limites

La presente propuesta será desarrollada para la dirección de seguro social de acuerdo a su necesidades, especificaciones y requerimientos.
El sistema no se enfocará en el ámbito financiero.
Por otra parte, el sistema es escalable en el tiempo.

1.8.2 Alcances

El sistema tendrá los siguientes módulos:

- Módulo de administración de usuarios, con distintos niveles de acceso al sistema y con técnicas de seguridad.
- Módulo de registro y control de estudiantes con el beneficio.
- Módulo de validación y revisión de documentos.
- Módulo de búsqueda de los estudiantes.
- Módulo de información detallada de los estudiantes.
- Módulo de reportes.
- Módulo de Atención al paciente.
- Módulo de sacar fichas.
- Módulo de Registro.
- Módulo de sacar fichas.

1.9 APORTES

Académico:
El sistema Web de registro de estudiantes al seguro social, es un importante aporte para mejorar el manejo de la información transparente, organizada y reducción de tiempo de respuesta a distintas peticiones ya que permitirá tener información centralizada en la base de datos, dando lugar a un incremento en la funcionalidad y desempeño dentro de la Dirección de Seguro Social, mediante un apropiado flujo de información.

Institucional:

La Universidad Pública de Alto y la Dirección de Seguro Social se beneficiará con un sistema confiable y seguro.

CAPITULO II

MARCO TEÓRICO

En este capítulo se expone los fundamentos teóricos que fueron necesarios para el Desarrollo del SISTEMA DE INFORMACION WEB PARA EL REGISTRO UNIVERSITARIO AL SEGURO SOCIAL.

2.1 Sistemas De Información

Un sistema de información es un conjunto de datos que interactúan entre sí con un fin común. En informática, los sistemas de información ayudan a administrar, recolectar, recuperar, procesar, almacenar y distribuir información relevante para los procesos fundamentales y las particularidades de cada organización.

2.2 Registro

Un registro informático es un tipo o conjunto de datos almacenados en un sistema.

2.3 INGENIERIA DE SOFTWARE

La ingeniería de software es una tecnología con varias capas como se aprecia en la figura 2.1 cualquier enfoque de ingeniería (incluso la de software) debe basarse en un compromiso organizacional con la calidad. El fundamento en el que se apoya la ingeniería de software es el compromiso con la calidad (Pressman R. S., 2010)

Figura 2.1: Capas de la Ingeniería de Software

Fuente (Pressman R. S., 2010, pág. 12)

2.3.1 Proceso

El fundamento de la ingeniería del software es la capa de proceso. El proceso de la ingeniería del software es la unión que mantiene juntas las capas de tecnología y que permite un desarrollo racional y oportuno de la ingeniería del software. El proceso define un marco de trabajo para un conjunto de Áreas clave de proceso (ACPs) que se deben establecer para la entrega efectiva de la tecnología de la ingeniería del software. Las áreas claves del proceso forman la base del control de gestión de proyectos del software y establecen el contexto en el que se aplican los métodos técnicos, se obtienen productos del trabajo (modelos, documentos, datos, informes, formularios, etc.), se establecen hitos, se asegura la calidad y el cambio se gestiona adecuadamente.

2.3.2 Métodos

Los métodos de la Ingeniería del Software indican «cómo» construir técnicamente el software. Los métodos abarcan una gran gama de tareas que incluyen análisis de requisitos, diseño, construcción de programas, pruebas y mantenimiento. Los métodos de la ingeniería del software dependen de un conjunto de principios básicos que gobiernan cada área de la tecnología e incluyen actividades de modelado y otras técnicas descriptivas.

2.3.3 HERRAMIENTAS

Las herramientas de la Ingeniería del Software proporcionan un enfoque automático o semiautomático para el proceso y para los métodos. Cuando se integran herramientas para que la información creada por una herramienta la pueda utilizar otra, se establece un sistema de soporte para el desarrollo del software llamado ingeniería del software asistida por computadora (CASE).

En el presente proyecto se hace un tratamiento de la estructura de las capas mencionadas en la figura 2.1 desarrollándolo de la siguiente forma:

- Enfoque de calidad se utiliza el modelo ISO-9126
- Capa de proceso se aplica la Ingeniería Web como marco de trabajo.

- Capa de método se hace el uso del Método de Lenguaje Unificado de Modelado Basado en Web (UWE).
- Capa de herramientas se hará uso de las Herramientas que se describió en el capítulo I en la sección 1.7 Herramientas las cuales se implementarán en una arquitectura cliente servidor en 3 capas.

2.4 METRICAS DE CALIDAD

En la mayoría de los desafíos técnicos, las métricas nos ayudan a entender tanto el proceso técnico que se utiliza para desarrollar un producto, como el propio producto.

El proceso para intentar mejorarlo, el producto se mide para intentar aumentar su calidad.

El principio, podría parecer que la necesidad de la medición es algo evidente después de todo es lo que nos permite cuantificar y por consiguiente gestionar de forma más efectiva.

Pero la realidad puede ser muy diferente. Frecuentemente la medición con lleva una gran controversia y discusión.

- ✓ ¿Cuáles son las métricas apropiadas para el proceso y para el producto?
- ✓ ¿Cómo se deben utilizar los datos que se recopilan?
- ✓ ¿Es bueno usar medidas para comparar gente, procesos o productos?

Estas preguntas y otras tantas docenas de ellas siempre surgen cuando se intenta medir algo que no sea medido en el pasado.

La medición es muy común en el mundo de la ingeniería. Medimos potencia de consumo, pesos, dimensiones físicas, temperaturas, voltajes, señales de ruidos por mencionar algunos aspectos. Desgraciadamente la medición se aleja de lo común en el mundo de la ingeniería del software. Encontramos dificultades en ponernos de acuerdo sobre que medir y como va evaluar las medidas.

Hay varias razones para medir un producto.

- ✓ Para indicar la calidad del producto
- ✓ Para evaluar la productividad de la gente que desarrolla el producto.
- ✓ Para evaluar los beneficios en términos de productividad y de calidad, derivados de uso de nuevos métodos y herramientas de la ingeniería de software.
- ✓ Para establecer una línea de base para la estimación
- ✓ Para ayudar a justificar el uso de nuevas herramientas o de formación adicional.

Las mediciones del mundo físico pueden englobarse en dos categorías: medidas directas y medidas indirectas.

Medidas directas. En el proceso de ingeniería se encuentran el costo, y el esfuerzo aplicado. Las líneas de código producidas, velocidad de ejecución, el tamaño de memoria y los defectos observados en un determinado periodo de tiempo.

Medidas Indirectas. Se encuentran la funcionalidad, calidad, complejidad, eficiencia, fiabilidad, facilidad de mantenimiento, entre otras.

2.4.1 Métricas de Calidad del Modelo de ISO-9126

El estándar ISO/IEC 9126 ha sido desarrollado en un intento de identificar los atributos clave de calidad para un producto de software (Pressman, 2002).

Este estándar es una simplificación del Modelo de McCall (Losavio, 2003), e identifica seis características básicas de calidad que pueden estar presentes en cualquier producto de software. El estándar provee una descomposición de las características en sub características, que se muestran en la Tabla 2.1

Tabla 2.2 Características de Calidad – Modelo ISO/IEC 9126

Características		Sub Características	
Funcionalidad	□	Adecuación	□ Seguridad
	□	Exactitud	□ Conformidad
	□	Interoperatvidad	
Fiabilidad	□	Madurez	□ Recuperabilidad
	□	Tolerancia a fallas	□ Conformidad
Usabilidad	□	Comprensibilidad	□ Atracción
	□	Facilidad de Aprendizaje	□ Conformidad
	□	Operabilidad	
Eficiencia	□	Comportamiento Temporal	□ Conformidad
	□	Utilización de recursos	
Mantenibilidad	□	analizabilidad	Capacidad de prueba
	□	□ Cambiabilidad	Conformidad
	□	□ Estabilidad	
Portabilidad	□	Adaptabilidad	□ Remplazabilidad
	□	Facilidad de Instalación	□ Conformidad
	□	Coexistencia	

Fuente: (Pressman, 2002)

2.4.1.1 Funcionalidad

Es la capacidad del producto de software de satisfacer los requisitos funcionales prescriptos y las necesidades implícitas de los usuarios y tiene las siguientes sub características:

- ✓ **Adecuación:** La capacidad del producto de software para proveer un adecuado conjunto de funciones para las tareas y objetivos especificados por el usuario.

- ✓ **Exactitud:** La capacidad del producto de software para proveer los resultados o efectos acordados con un grado necesario de precisión.
- ✓ **Interoperabilidad:** La capacidad del producto de software de interactuar con uno o más sistemas especificados.
- ✓ **Seguridad:** La capacidad del producto de software para proteger la información y los datos de modo que las personas o los sistemas no autorizados no puedan leerlos o modificarlos.
- ✓ **Conformidad:** La capacidad del producto software para adaptarse a los estándares, convecciones o regulaciones en leyes y prescripciones relativas a la funcionalidad.

2.4.1.2 Fiabilidad

La capacidad del producto de software para mantener un nivel de desempeño bajo condiciones establecidas, por un periodo tiempo y contempla las siguientes sub características.

- **Madurez:** La capacidad del producto de software para evitar fallas como resultado de errores en el software.
- **Tolerancia a Errores:** La capacidad del producto de software para mantener un nivel especificado de funcionamiento en caso de errores del software o de incumplimiento de su interfaz especificada.
- **Recuperabilidad:** La capacidad del producto de software para restablecer un nivel especificado de funcionamiento y recuperar los datos afectados directamente en el caso de una falla.
- **Conformidad:** La capacidad del producto software para adaptarse a estándares, convenciones y regulaciones referidas a la fiabilidad.

2.4.1.3 Usabilidad

Es la capacidad del producto de software de ser entendido, aprendido, usado y atractivo al usuario, cuando es utilizado bajo las condiciones especificadas y se divide en las siguientes subcarateristicas.

- **Entendibilidad:** La capacidad del producto de software para permitir al usuario entender si el software es adecuado, y cómo puede ser utilizado para las tareas y las condiciones particulares de la aplicación.
- **Capacidad de Aprendizaje:** La capacidad del producto de software para permitir al usuario aprender su aplicación. Un aspecto importante a considerar aquí es la documentación del software.
- **Operabilidad:** La capacidad del producto de software para permitir al usuario operarlo y controlarlo. Los aspectos de propiedad, de cambio, de adaptabilidad y de instalación pueden afectar la operabilidad.
- **Atracción:** Capacidad del producto para atraer al usuario.
- **Conformidad:** La capacidad del producto software para adaptarse a estándares, convenciones y regulaciones referidas a la usabilidad.

2.4.1.4 Portabilidad

La capacidad del software para ser trasladado de un entorno a otro. El entorno puede incluir entornos organizacionales, de hardware o de software.

- **Adaptabilidad:** La capacidad del producto de software para ser adaptado a diferentes entornos especificados sin aplicar acciones o medios diferentes de los previstos para el propósito del software considerado.
- **Facilidad de Instalación:** La capacidad del producto de software para ser instalado en un ambiente especificado.
- **Reemplazabilidad:** La capacidad del producto de software para ser utilizado en lugar de otro producto de software, para el mismo propósito y en el mismo entorno.
- **Coexistencia:** La capacidad del producto para coexistir con otro software independiente en un ambiente común compartiendo recursos.
- **Conformidad:** La capacidad del producto software para adaptarse a estándares referidas a la portabilidad.

2.4.1.5 Mantenibilidad

Capacidad del producto de software para ser modificado. Las modificaciones pueden incluir correcciones, mejoras o adaptación del software a cambios en el entorno, y especificaciones de requerimientos funcionales.

- **Analizabilidad:** La capacidad del producto de software para atenerse a diagnósticos de deficiencias o causas de fallas en el software o la identificación de las partes a ser modificadas.
- **Cambiabilidad:** La capacidad del software para permitir que una determinada modificación sea implementada.
- **Estabilidad:** La capacidad del producto de software para evitar efectos inesperados debido a modificaciones del software.
- **Capacidad de prueba:** La capacidad del software para permitir que las modificaciones sean validadas.
- **Conformidad:** La capacidad del producto software de cumplir los estándares o convenciones relativas a la mantenibilidad.

2.4.1.6 Eficiencia

La capacidad del producto de software para proveer un desempeño adecuado, de acuerdo a la cantidad de recursos utilizados y bajo las condiciones planteadas.

- **Comportamiento temporal:** La capacidad del producto de software para proveer tiempos adecuados de respuesta y procesamiento, y ratios de rendimiento cuando realiza su función bajo las condiciones establecidas.
- **Utilización de recursos:** La capacidad del producto de software para utilizar cantidades y tipos adecuados de recursos cuando este funciona bajo las condiciones establecidas.
- **Conformidad:** La capacidad del producto software de cumplir los estándares o convenciones relacionadas a la eficiencia.

Cada factor de calidad (FC) se puede obtener como combinación de una o varias métricas:

FC=C1 M1 +……+Cn Mn **Ecuación 2.1**

Dónde:

FC: Es el factor de calidad

Cn: Coeficiente de regresión.

Mn: Métricas que afectaran al factor de calidad.

La calidad del software se calculará en función de los factores considerados y para cada factor las métricas establecidas, promediando los porcentajes de los factores de calidad obtenidos. Siendo el factor de calidad:

$$QT = \sum \frac{FACCA}{N° \; de \; FC}$$

2.4.2 Métricas Basadas en la Función

La métrica punto función (PF) se usa de manera efectiva como medio para medir la funcionalidad que entrega un sistema. PF se deriva empleando una relación empírica basada en medidas contables del dominio de la información del software y las evaluaciones de la complejidad de este.

Sea los valores de dominio de la información según la Tabla 2.3

Tabla 2.3 Tabla de Factor de Ponderación

Parámetros de medición		Cuenta	Factores de Ponderación			Total
			Simple	Medio	Complejo	
1	Nro. de Entradas de Usuario	X	3	4	6	=

2	Nro. de Salidas de Usuario	X	4	5	7	=
3	Nro. de Peticiones de Usuario	X	3	4	6	=
4	Nro. de Archivos	X	7	10	15	=
5	Nro. de Interfaces Externas	X	5	7	10	=
Cuenta Total						

Fuente: (PRESSMAN, 2002)

- **Número de entradas de usuario:** Se cuenta cada entrada del usuario que proporcione al software diferentes datos aplicados a la aplicación. Las entradas deben ser distinguidas de las peticiones que se contabilizan por separado.
- **Número de Salidas del usuario:** Se encuentra cada salida que proporciona al usuario información orientada a la aplicación. En este contexto se refieren a informes, pantallas, mensajes de error.
- **Número de peticiones al usuario:** Una petición está definida como una entrada interactiva que resulta de la generación de algún tipo de respuesta en forma de salida interactiva se cuenta cada petición por separado.
- **Numero de archivos:** Se cuenta cada archivo maestro lógico es decir una agrupación lógica de datos que puede ser una parte en gran base de datos o un archivo independiente.
- **Numero de interfaces externas:** Se cuentan todas las interfaces legibles por la máquina, por ejemplo: archivo de datos, en cintas o discos que son utilizados para transmitir información a otro sistema.

A cada conteo se le asocia un valor de complejidad, no obstante, la determinación de la complejidad es un poco subjetiva.

Para Calcular los puntos función se usa la siguiente ecuación.

$$PF= Cuenta\ Total\ *(0.65+0.01*\sum Fi)$$

Cuenta Total es la suma de todas las entradas de PF obtenidas de la tabla anterior **Fi** donde **i** puede ser de uno hasta 14 los valores de ajuste de complejidad basados en las respuestas a las cuestiones señaladas de la tabla

Tabla 2.4 Factor de Escala

0	1	2	3	4	5
Sin Influencia	**Incidental**	**Moderado**	**Medio**	**Significativo**	**Esencial**

Fuente: (PRESSMAN, 2002)

Tabla 2.5 Valoración Métrica Punto Función

Nro.	Preguntas	Fi
1	¿Requiere el sistema copia de seguridad y recuperación?	
2	¿Requiere comunicación de datos?	
3	¿Existen funciones de procesos distribuidos?	
4	¿El rendimiento es crítico?	
5	¿Sera ejecutado el sistema en entorno existente y fuertemente utilizado?	
6	¿Entrada de datos EN LINEA?	
7	¿Requiere la entrada de datos interactiva que las transiciones de entrada se llevan a cabo sobre múltiples pantallas o variadas opciones?	
8	¿Se actualizan los archivos maestros de forma interactiva?	
9	¿Son complejas de las entradas de salidas de archivos?	
10	¿Lógica del proceso Interno Compleja?	

11	¿Se diseña el código para ser reutilizable?
12	¿Están incluidas en el diseño conversiones de instalación?
13	¿Instalaciones Múltiples?
14	¿Facilidad de Cambios?

Fuente: (PRESSMAN, 2002)

Los valores constantes de la ecuación anterior y los factores de peso aplicados en las encuestas de los ámbitos de información han sido determinados empíricamente.

2.5 INGENIERÍA WEB

La ingeniería Web se debe al crecimiento desenfrenado que está teniendo la Web está ocasionando un impacto en la sociedad y el nuevo manejo que se le está dando a la información en las diferentes áreas en que se presenta ha hecho que las personas tiendan a realizar todas sus actividades por esta vía.

Para garantizar el buen funcionamiento y mantenimiento de los sitios web, este debe contar con ciertos atributos y características que en conjunto forman un concepto muy importante, para alcanzar el éxito en cualquier organización, herramienta, y todo aquello que se pueda considerar como servicio. Dicho concepto es la calidad, que con atributos como, usabilidad, navegabilidad, seguridad, mantenibilidad, entre otros, hace posible por un lado la eficiencia del artefacto Web y por ende la satisfacción del usuario final. (Chavez, 2012)

Cabe destacar que la ingeniería de la Web hace una diferencia entre un sitio web y un aplicativo, ya que la ingeniería de la Web no se dedica a la construcción de sitios web si no a la construcción de aplicativos web, la principal característica que los distingue (aplicativos de sitios web) es que los sitios web son sitios en la web en donde se publica contenido generalmente estático o un muy bajo nivel de interactividad con el usuario, mientras que los aplicativos son lugares con alto contenido de interactividad y funcionalidades que bien podrían ser de un software convencional, el aplicativo web más sencillo seria uno que contenga formularios y

subiendo de nivel encontramos los que realizas conexión con bases de datos remotas, y administradores de contenidos entre otras.

Entonces la Ingeniería de la Web es la aplicación de metodologías sistemáticas, disciplinadas y cuantificables al desarrollo eficiente, operación y evolución de aplicaciones de alta calidad en la World Wide Web(www). En este sentido, la Ingeniería de la Web hace referencia a las metodologías, técnicas y herramientas que se utilizan en el desarrollo de aplicaciones Web complejas y de gran dimensión en las que se apoya la evaluación, diseño, desarrollo, implementación y evolución de dichas aplicaciones.

2.5.1 Características de la Ingeniería Web

El desarrollo de aplicaciones Web posee determinadas características que lo hacen diferente del desarrollo de aplicaciones o software tradicional y sistemas de información. La Ingeniería Web es multidisciplinar y aglutina contribuciones de diferentes áreas: arquitectura de la información, ingeniería de hipermedia/hipertexto, ingeniería de requisitos, diseño de interfaz de usuario, usabilidad, diseño gráfico y de presentación, diseño y análisis de sistemas, Ingeniería de Software, Ingeniería de Datos, indexado y recuperación de información, testeo, modelado y simulación, despliegue de aplicaciones, operación de sistemas y gestión de proyectos. (Hernández Salguera, 2012)

La ingeniería de la Web no es un clon o subconjunto de la ingeniería de software, aunque ambas incluyen desarrollo de software y programación, pues a pesar de que la ingeniería de la Web utiliza principios de Ingeniería de Software, incluye nuevos enfoques, metodologías, herramientas, técnicas, guías y patrones para cubrir los requisitos únicos de las aplicaciones web. Sin embargo, el término de ingeniería de la web ha sido un término muy controversial especialmente para profesionales en disciplinas tales como la ingeniería del software ya que no la consideran como un campo dentro de la ingeniería. (Hernández Salguera, 2012)

Los principales aspectos de la ingeniería de la Web incluyen, entre otros, los siguientes temas:

- Diseño de procesos de negocio para aplicaciones web.
- Herramientas CASE para aplicaciones web.
- Generación de código para aplicaciones web.
- Desarrollo web colaborativo.
- Modelado conceptual de aplicaciones web.
- Diseñó de Modelos de datos para sistemas de información web.
- Ingeniería web empírica.
- Entornos de desarrollo de aplicaciones web integrados.
- Herramientas de autor para contenido multimedia.
- Pruebas de rendimiento de aplicaciones basadas en web.
- Personalización y adaptación de aplicaciones web.
- Modelado de procesos para aplicaciones web.
- Herramientas y métodos de prototipado.
- Control de calidad y pruebas de sistemas.
- Ingeniería de requisitos para aplicaciones web.
- Aplicaciones para la Web Semántica.
- Factorías de software para la web.
- Métodos, herramientas y automatización de pruebas para aplicaciones web.
- Aplicaciones web móviles y ubicuas.
- Usabilidad de aplicaciones web.
- Accesibilidad para la web.
- Metodologías de diseño web.
- Formación en ingeniería de la web.

- Diseño de interfaces de usuario.
- Métricas para la web, estimación de costes y medición.
- Gestión de proyectos web y gestión de riesgos.
- Desarrollo y despliegue de servicios web.

2.6 INGENIERÍA WEB BASADA EN UML (UWE)

UWE surgió a finales de la década de los noventa con la idea de encontrar un estándar para construir modelos de análisis y diseño para sistemas basados en la Web. El objetivo fue usar un lenguaje común o por lo menos definir un modelo a partir de los existentes.

UWE es una metodología basado en el proceso Unificado y UML para el desarrollo de aplicaciones Web.

UWE cubre todo el ciclo de vida de las aplicaciones Web, su proceso de desarrollo se basa en tres fases principales que son:

- Captura de Requisitos
- Análisis y Diseño
- Implementación

2.6.1 Características

La metodología UWE define vistas especiales representadas gráficamente por diagramas UML, tales como el análisis de requerimientos, diseño conceptual modelo de navegación y el modelo de presentación. UWE no limita el número de diagramas posibles de una aplicación

- Es una metodología orientada a objetos, iterativa e incremental basada en UML
- Se basa también en el proceso de desarrollo del software unificado
- Proporciona un diseño sistemático y una generación semiautomática en las aplicaciones web atreves de un Framework de publicación XML

- UWE define su propio perfil UML en el cual se definen todos los elementos necesarios para modelar los diferentes aspectos de una aplicación web que son la presentación y la navegación entre otros.
- En esta metodología se proponen 2 tipos de diagramas para el modelado de la navegación que son: el modelo de espacio , el cual se definen todos los caminos navegacionales, es decir todas aquellas asociaciones de navegación directa entre los distintos objetos de la aplicación más bien conocida como clases de navegación, y el segundo modelo de estructura de navegación el cual se define la estructura de acceso que son utilizadas en la navegación, es decir todo aquello referente a menús, índices y demás.

2.6.2 Modelos de la Metodología UWE

UWE proporciona guías para la construcción de modelos de forma sistemática, enfocándose en la personalización y en estudio de casos de uso.

Las actividades principales son el análisis de requerimientos, diseño conceptual, diseño de navegación y el diseño de presentación los mismos generan los siguientes modelos principales:

- **Modelo de Casos de Uso:** modelo para capturar los requisitos del sistema.
- **Modelo Conceptual:** Es un modelo para el desarrollo del contenido.
- **Modelo de Navegación:** Se muestra la navegación y flujo del sistema.
- **Modelo Presentación:** Muestra la forma que se va presentar frente al usuario.

En cuanto a los requisitos, UWE los clasifica dependiendo del carácter de cada uno. Además, distingue entre las fases de captura, definición y validación de requisitos.

2.6.2.1 Modelo de Casos de Uso

Un diagrama de casos de uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa.

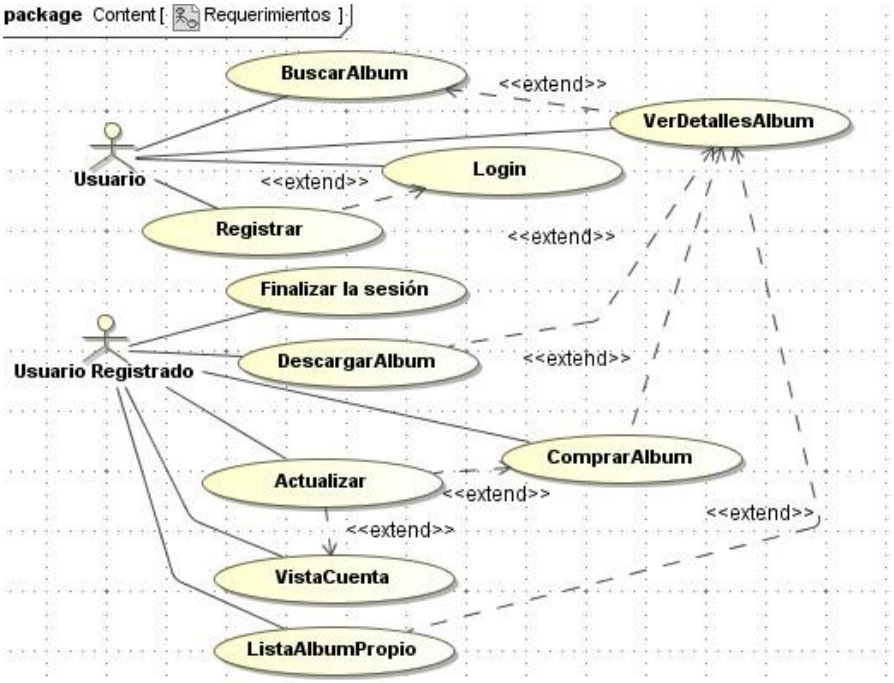


Figura 2.2: Modelo de Caso de Uso de la Aplicación Portal Musical

Fuente: (Engineering, 2012)

En la figura podemos ver que las funciones de un usuario no registrado son limitadas, ya que solo puede buscar información sobre los discos, registrarse y loguearse, para llegar a ser usuario Registrado.

En cuanto a las funciones del usuario Registrado, son las mismas que el usuario no registrado más las funcionalidades adicionales de comprar el disco, descargar el disco y las opciones de administración de su cuenta de crédito personal: recargar cuenta, ver cuenta y ver historial de discos comprados.

2.6.2.2 Modelo de Contenido

El modelo de contenido visualiza el dominio de información relevante para el sistema Web que incluye principalmente el contenido de la aplicación web.

En nuestro ejemplo, la información es proporcionada por la clase de álbum, artista y canción. Un diagrama de clases UML y clases UML simples se utilizan para modelar el contenido.

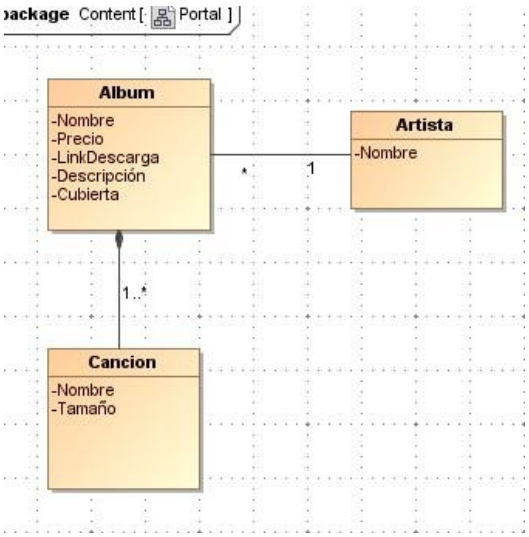


Figura 2.3 Modelo de Contenido de la Aplicación Portal Musical

Fuente: (Engineering, 2012)

Modelo de contenido define el contenido de los datos de la aplicación.

2.6.2.3 Modelo de Navegación

Este diseño especifica que objetos pueden ser visitados a través de la aplicación web.

En el proceso de construir el modelo de espacio de navegación el desarrollador toma decisiones cruciales de diseño, tales como qué vista del modelo conceptual es necesaria para la aplicación y cuáles serán los caminos de navegación requeridos para el aseguramiento de la funcionalidad. Los elementos utilizados para este modelo son las clases de navegación y las asociaciones de navegación, que expresan la navegación directa

La navegación está fuertemente simplificada y se destina principalmente para demostrar el uso de los elementos del modelo para lo cual se utiliza estereotipos e iconos las que se describen en la siguiente Tabla.

Tabla 2.6 Descripción de Estereotipos

Icono	Descripción	Icono	Descripción
	Clase de Navegación		Menú
	Índice		Pregunta
	Visita Guiada		Clase de Proceso
	Nodo Externo		

Fuente: Elaboración Propia Basado en (KOCH, 2000)

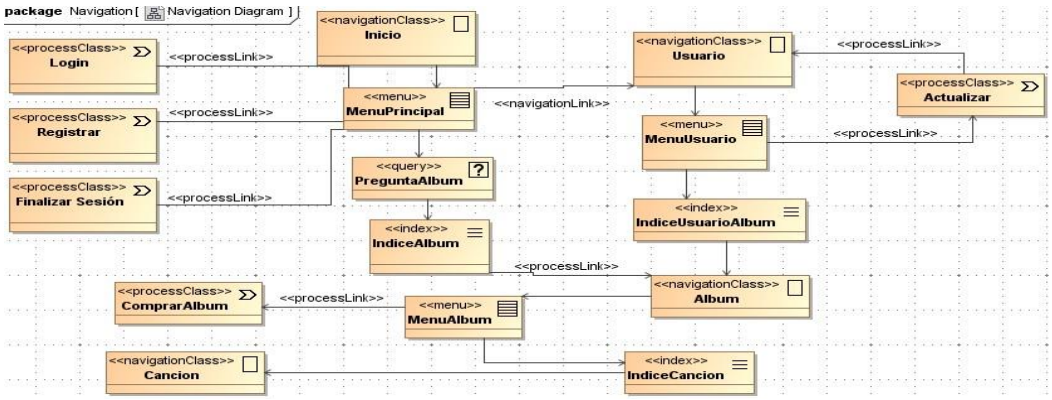


Figura 2.5: Modelo de Navegación de la Aplicación Portal Musical

Fuente: (Engineering, 2012)

En este diagrama podemos ver como se relacionan las clases entre ellas, llegando a un entendimiento superior que un simple diagrama UML con la extensión que UWE aporta a este estándar, esta extensión es la que explicamos en el modelo de navegación de la introducción de este documento con los estereotipos correspondientes.

2.6.2.4 Modelo de Proceso

El modelo de proceso del ejemplo se compone de las clases de procesos que se integran en el modelo de navegación, las clases de proceso adicionales para manejar la entrada del usuario y las actividades que definen el comportamiento de los procesos. El modelo de proceso se muestra la relación de la clase de proceso principal y clases de procesos, que se han asociado el flujo del proceso.

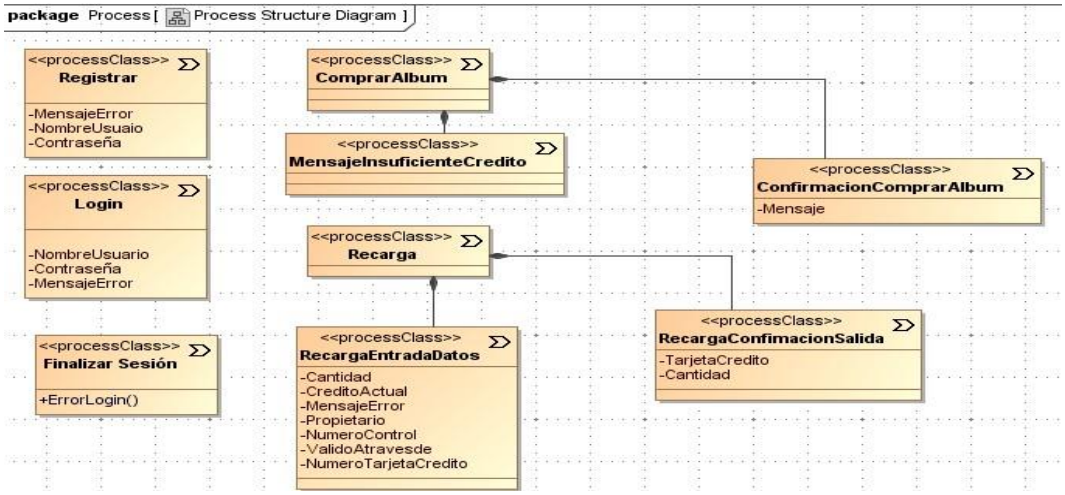


Figura 2.6: Modelo de Proceso de la Aplicación Portal Musical

Fuente: (Engineering, 2012)

2.6.2.5 Modelo de Flujo de Procesos

En este diagrama podemos ver el flujo de interno de la operación de loguearse del sitio web.

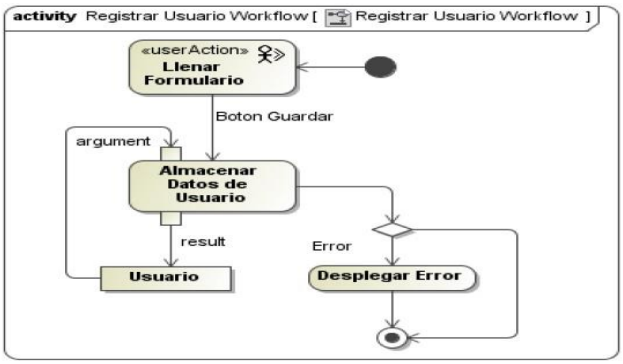


Figura 2.7 Flujo de Proceso para Registro de Usuario

Fuente: (Engineering, 2012)

2.6.2.6 Modelo de Presentación

Para el modelo de presentación se utiliza los siguientes Estereotipos e iconos como se detalla en el siguiente cuadro:

Tabla 2.7 Descripción de Estereotipos

Icono	Descripción	Icono	Descripción
	Grupo de Presentación		Página de Presentación
	Texto		Entrada de Texto
	Ancla		FileUpload
	Botón		Imagen
	Formulario		Componente de Cliente
	Alternativas de Presentación		Selección

Fuente: Elaboración Propia Basado en (KOCH, 2000)

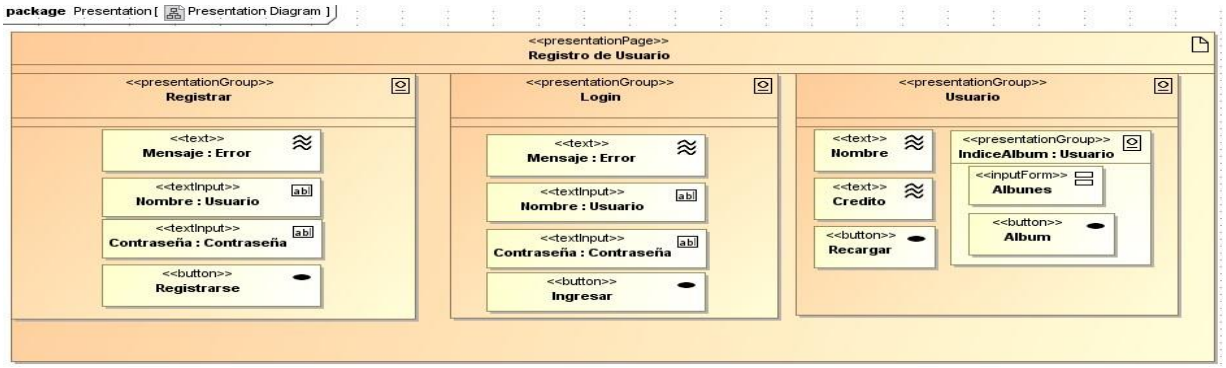


Figura 2.8: Modelo de Presentación de Registro Usuario

Fuente: (Engineering, 2012)

El modelo de presentación del ejemplo se muestra como un diagrama de estructura compuesta UML. En este tipo de diagrama, las propiedades que están contenidos por la composición se muestran como rectángulos que están contenidos en la figura de la clase que contiene.

2.7 ARQUITECTURA CLIENTE SERVIDOR

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

La red cliente-servidor es aquella red de comunicaciones en la que todos los clientes están conectados a un servidor, en el que se centralizan los diversos recursos y aplicaciones con que se cuenta; y que los pone a disposición de los clientes cada vez que estos son solicitados. Esto significa que todas las gestiones que se realizan se concentran en el servidor, de manera que en él se disponen los requerimientos provenientes de los clientes que tienen prioridad, los archivos que son de uso público y los que son de uso restringido, los archivos que son de sólo lectura y los que, por el contrario, pueden ser modificados, etc. Este tipo de red puede utilizarse conjuntamente en caso de que se esté utilizando en una red mixta.

2.7.1 Características

Un sistema Cliente/Servidor es un Sistema de Información distribuido basado en las siguientes características:

- Servicio: unidad básica de diseño. El servidor los proporciona y el cliente los utiliza.
- Recursos compartidos: Muchos clientes utilizan los mismos servidores y, a través de ellos, comparten tanto recursos lógicos como físicos.
- Protocolos asimétricos: Los clientes inician “conversaciones”. Los servidores esperan su establecimiento pasivamente.
- Transparencia de localización física de los servidores y clientes: El cliente no tiene por qué saber dónde se encuentra situado el recurso que desea utilizar.
- Independencia de la plataforma HW y SW que se emplee.

- Sistemas débilmente acoplados. Interacción basada en envío de mensajes.
- Encapsulamiento de servicios. Los detalles de la implementación de un servicio son transparentes al cliente.
- Escalabilidad horizontal (añadir clientes) y vertical (ampliar potencia de los servidores).

Integridad: Datos y programas centralizados en servidores facilitan su integridad y mantenimiento

2.7.2 Ventajas

- **Centralización del control:** Los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
- **Escalabilidad:** Se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).
- **Fácil mantenimiento:** Al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar, o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio. Esta independencia de los cambios también se conoce como encapsulación.
- **Tecnología:** Existen suficientemente desarrolladas, diseñadas para el paradigma de C/S que aseguran la seguridad en las transacciones, la amigabilidad de la interfaz, y la facilidad de empleo.
- **Manejo de Capas:** Consiste en una capa de la Presentación, otra capa de la lógica de la aplicación y otra capa de la base de datos.

2.7.3 Desventajas

- Dificultad de la programación en comparación con modelo de dos capas ya que se comunican más dispositivos para realizar las peticiones de los

clientes; además el incremento del tráfico en la red debido al flujo de los datos

2.7.4 Seguridad de Modelo Cliente /Servidor

- Es de gran importancia por el valor intrínseco para los usuarios. Tiene tres componentes:
- Confidencialidad. - Protección contra individuos no autorizados.
- Integridad. - Protección contra la alteración o corrupción.
- Disponibilidad. - Protección contra la interferencia con los procedimientos de acceso a los recursos.

2.7.5 Arquitectura de Dos y Tres Niveles Cliente /Servidor

2.7.5.1 Arquitectura Dos Niveles

La arquitectura en 2 niveles se utiliza para describir los sistemas cliente/servidor en donde el cliente solicita recursos y el servidor responde directamente a la solicitud, con sus propios recursos. Esto significa que el servidor no requiere otra aplicación para proporcionar parte del servicio.

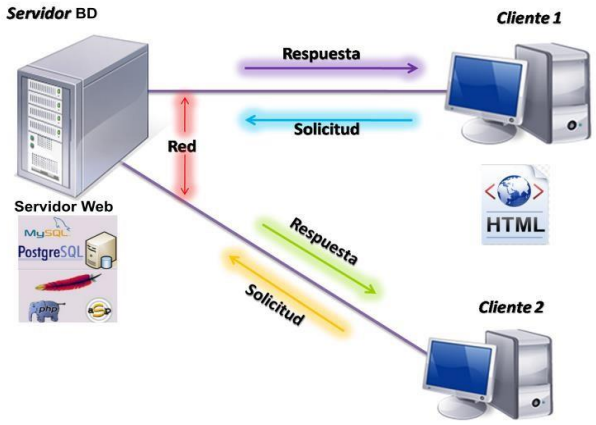


Figura 2.9: Arquitectura en dos niveles

Fuente: (Leal Castellano, Leal Molina, & Medina Castiblanco, 2011)

2.7.5.2 Arquitectura Tres Niveles

En la arquitectura en tres niveles, existe un nivel intermediario. Esto significa que la arquitectura generalmente está compartida por:

- Un cliente, es decir, el equipo que solicita los recursos, equipado con una interfaz de usuario (generalmente un navegador Web) para la presentación
- El servidor de aplicaciones (también denominado software intermedio), cuya tarea es proporcionar los recursos solicitados, pero que requiere de otro servidor para hacerlo
- El servidor de datos, que proporciona al servidor de aplicaciones los datos que requiere

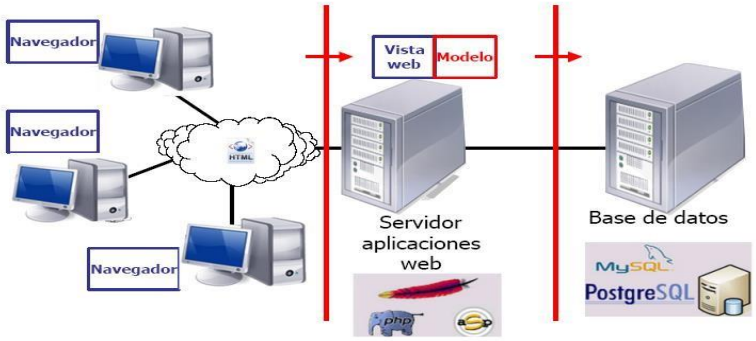


Figura 2.10: Arquitectura en tres niveles

Fuente: (Leal Castellano, Leal Molina, & Medina Castiblanco, 2011)

El uso masivo del término arquitectura en tres niveles también denota las siguientes arquitecturas:

- Aplicación compartida entre un cliente, un software intermedio y un servidor empresarial
- Aplicación compartida entre un cliente, un servidor de aplicaciones y un servidor de base de datos empresarial.

2.7.6 Comparación entre Ambos Tipos de Arquitecturas

La arquitectura en dos niveles es, por lo tanto, una arquitectura cliente/servidor en la que el servidor es polivalente, es decir, puede responder directamente a todas las solicitudes de recursos del cliente.

Sin embargo, en la arquitectura en 3 niveles, las aplicaciones al nivel del servidor son descentralizadas de uno a otro, es decir, cada servidor se especializa en una determinada tarea, (por ejemplo: servidor web/servidor de bases de datos). La arquitectura en 3 niveles permite:

- Un mayor grado de flexibilidad
- Mayor seguridad, ya que la seguridad se puede definir independientemente para cada servicio y en cada nivel
- Mejor rendimiento, ya que las tareas se comparten entre servidores

Tabla 2.8 Comparación de Arquitecturas

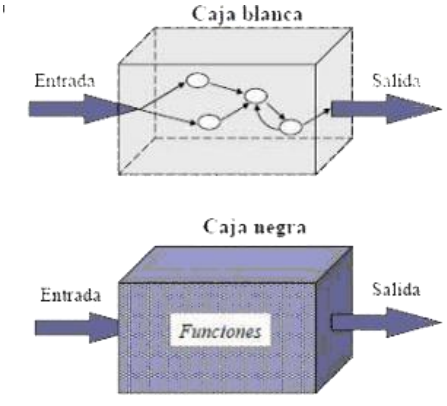
Dos Niveles	Tres Niveles
• **Consiste en una capa de presentación y lógica de la aplicación; y la otra de la base de datos. Cuando se requiera poco procesamiento de datos en la organización.** • **Cuando se tiene una base de datos centralizada en un solo servidor.**	• Consiste en una capa de la Presentación, otra capa de la lógica de la aplicación y la capa de la base de datos. • Cuando se requiera mucho procesamiento de datos en la aplicación. • En aplicaciones donde la funcionalidad este en constante cambio. • Cuando los procesos no están relativamente muy relacionados con los datos. • Cuando se requiera aislar la tecnología de la base de datos para que sea fácil de cambiar.

• **Cuando la base de datos es relativamente estática.** • **Cuando se requiere un mantenimiento mínimo.**	• Cuando se requiera separar el código del cliente para que se facilite el mantenimiento. • Está muy adecuada para utilizarla con la tecnología orientada a objetos.

Fuente: (Leal Castellano, Leal Molina, & Medina Castiblanco, 2011)

2.8 CAJA NEGRA Y CAJA BLANCA

Los términos caja negra y caja blanca son muy utilizados dentro de la teoría en sistemas con respecto al tipo de perspectiva con la cual es estudiado un sistema. estos dos tipos de estudios dentro de un sistema son usados dependiendo de lo que exactamente deseemos estudiar, si queremos saber cómo funciona internamente un elemento de un sistema se utiliza el termino caja blanca. si lo que queremos es estudiar la interacción de dicho modulo con los demás módulos del sistema se utiliza el termino caja negra.



2.9 SEGURIDAD DE APLICACIONES WEB

La seguridad de las aplicaciones Web es un tema crítico y complejo para los desarrolladores Web. Ya que requiere realizar estudios y tener entendimiento sobre los puntos de vulnerabilidad en cada aplicación Web.

Se puede clasificar los puntos más importantes:

Seguridad en el servidor

- Servidor Web
- Servidor de Base de Datos
- Lenguaje de programación

Seguridad en la Aplicación

- Control de Acceso
- Validación de datos de entrada
- Pruebas de código

Seguridad en la Comunicación

- Protocolos de seguridad
- Certificados de Comunicación

Copias de Seguridad

- Backup de la Base de Datos
- Backup del control de Acceso

Las pruebas de seguridad están diseñadas para probar la vulnerabilidad en el ambiente del lado del cliente, las comunicaciones de red que ocurren mientras los datos pasan del cliente al servidor y de vuelta y el ambiente del lado del servidor. Cada uno de estos dominios puede recibir ataques y es labor de quien prueba la seguridad descubrir las debilidades que pueden explotar quienes tengan la intención de hacerlo (Pressman R., 2005)

A continuación, se muestra las vulnerabilidades al lado del cliente

- Vulnerabilidad en los navegadores
- Vulnerabilidad en los correos electrónicos
- Acceso no autorizado a cookies
- Simulación en la comunicación con el servidor

Por lo tanto, se puede aplicar las siguientes medidas

Cortafuegos: Mecanismo de filtrado combinación de hardware y software para examinando cada paquete de información entrante para garantizar que no sea un dato sospechoso

Uso de Autenticación: Una contraseña o clave es una forma de autentificación que utiliza información secreta para controlar el acceso hacia algún recurso. A aquellos que desean acceder a la información se les solicita una clave; si conocen o no conocen la contraseña, se concede o se niega el acceso a la información según sea el caso.

Modelo de Criptografía: La encriptación está asociada con la transformación de un mensaje inteligente a una forma no inteligente con la ayuda de una clave secreta antes de que sea colocada en un medio inseguro.

Algoritmo básico del hash: Una función criptográfica **hash**- usualmente conocida como "**hash**"- es un algoritmo matemático que transforma cualquier bloque arbitrario de datos en una nueva serie de caracteres con una longitud fija.

Independientemente de la longitud de los datos de entrada, el valor **hash** de salida tendrá siempre la misma longitud.

2.10 MODELO VISTA CONTROLADOR (MVC)

El modelo MVC es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista.

2.10.1 Modelo

Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos; por ejemplo, no permitiendo comparar un número de unidades negativo, calculando si hoy es el cumpleaños del usuario o los totales, impuestos o importes en un carrito de la compra.

2.10.2 Vista

Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

2.10.3 Controlador

Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

2.10.4 Características del Modelo Vista Controlador

Aunque se pueden encontrar diferentes implementaciones de MVC, la característica del flujo que sigue el control generalmente es el siguiente:

- El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón, enlace, etc.)
- El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.
- El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el. Los controladores complejos están a menudo estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.
- El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo. El modelo no debe tener conocimiento directo sobre la vista. Sin embargo, el patrón de observador puede ser utilizado para proveer cierta dirección entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio.
- La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente.

2.11 METODO DE ESTIMACIÓN DE COSTES DEL SOFTWARE

2.11.1 Modelo Constructivo de Costos (COCOMO)

El Modelo Constructivo de Costos COCOMO, por su acrónimo del inglés (COnstructive COst MOdel) es un modelo matemático de base empírica utilizado para estimación de costos de software. Incluye tres submodelos, cada uno ofrece un nivel de detalle y aproximación, cada vez mayor, a medida que avanza el proceso de desarrollo del software: básico, intermedio y detallado (COCOMO, 2013).

Este modelo fue desarrollado por Barry W. Boehm a finales de los años 70 y comienzos de los 80, exponiéndolo detalladamente en su libro "Software

Engineering Economics" (Hall, 1981)

2.11.2 Características

Pertenece a la categoría de modelos de subestimaciones basados en estimaciones matemáticas. Está orientado a la magnitud del producto final, midiendo el "tamaño" del proyecto, en líneas de código principalmente.

2.11.3 Inconvenientes

- ✓ Los resultados no son proporcionales a las tareas de gestión ya que no tiene en cuenta los recursos necesarios para realizarlas.
- Se puede desviar de la realidad si se indica mal el porcentaje de líneas de comentarios en el código fuente.
- Es un tanto subjetivo, puesto que está basado en estimaciones y parámetros que pueden ser "vistos" de distinta manera por distintos analistas que usen el método.
- Se miden los costes del producto, de acuerdo a su tamaño y otras características, pero no la productividad.
- La medición por líneas de código no es válida para orientación.
- Utilizar este modelo puede resultar un poco complicado, en comparación con otros métodos (que también sólo estiman)

2.11.4 Ecuaciones del Modelo COCOMO

Las ecuaciones que se utilizan en los tres modelos son:

E = a(Kl)^b*m(X), en persona-mes

Tdev = c(E)^d, en meses

P = E/Tdev, en personas

Dónde:

E es el esfuerzo requerido por el proyecto, en persona-mes

Tdev es el tiempo requerido por el proyecto, en meses.

P es el número de personas requerido por el proyecto

a, b, c y d son constantes con valores definidos en una tabla, según cada Submodelo

KI es la cantidad de líneas de código, en miles.

m(X) Es un multiplicador que depende de 15 atributos.

A la vez, cada SubModelo también se divide en modos que representan el tipo de proyecto, y puede ser:

Modo Orgánico: Un pequeño grupo de programadores experimentados desarrollan software en un entorno familiar. El tamaño del software varía desde unos pocos miles de líneas (tamaño pequeño) a unas decenas de miles (medio).

Modo Semilibre o Semiencajado: Corresponde a un esquema intermedio entre el orgánico y el rígido; el grupo de desarrollo puede incluir una mezcla de personas experimentadas y no experimentadas.

Modo Rígido o Empotrado: El proyecto tiene fuertes restricciones, que pueden estar relacionadas con la funcionalidad y/o pueden ser técnicas. El problema a resolver es único y es difícil basarse en la experiencia, puesto que puede no haberla.

2.11.5 Modelos de COCOMO

2.11.6 Modelo Básico:

Se utiliza para obtener una primera aproximación rápida del esfuerzo, y hace uso de la siguiente tabla de constantes para calcular distintos aspectos de costes.

Tabla 2.9 Constantes de Costes

. Modo	a	b	c	d
Orgánico	2.40			0.38
Semilibre	3.00	1.05 2.50 1.12	2.50	0.35
Rígido	3.60	1.20	2.50	0.32

Fuente: (COCOMO, 2013) Estos valores son para las fórmulas:

- Personas necesarias por mes para llevar adelante el proyecto (**MM**) = a*(KI^b)
- Tiempo de desarrollo del proyecto (**TDEV**) = c*(MM^d)
- Personas necesarias para realizar el proyecto (**CosteH**) = MM/TDEV
- Costo total del proyecto (**CosteM**) = CosteH * Salario medio entre los programadores y analistas.

Se puede observar que a medida que aumenta la complejidad del proyecto (modo), las constantes aumentan de 2.4 a 3.6, que corresponde a un incremento del esfuerzo del personal. Hay que utilizar con mucho cuidado el modelo básico puesto que se obvian muchas características del entorno

2.11.7 Modelo Intermedio

En este modelo se introducen 15 atributos de coste para tener en cuenta en el entorno de trabajo, incrementando así la precisión de la estimación.

Para este ajuste, al resultado de la fórmula general se lo multiplica por el coeficiente surgido de aplicar los atributos que se decidan utilizar.

Los valores de las constantes a reemplazar en la fórmula se muestran en la siguiente

tabla 2.5.

Tabla 2.10 Coeficientes de COCOMO

PROYECTO SOFTWARE	A	b
	3,2	
Orgánico		1,05
Semi-Libre	3,0	1,12
Rígido	2,8	1,20

Fuente: (COCOMO, 2013)

Se puede observar que los exponentes son los mismos que los del modelo básico, confirmando el papel que representa el tamaño; mientras que los coeficientes de los modos orgánico y rígido han cambiado, para mantener el equilibrio al rededor del SemiLibre con respecto al efecto multiplicador de los atributos de coste.

2.11.7.1 Ecuaciones Nominales De Coste

Para cada modo de desarrollo, los 15 atributos del coste intervienen como multiplicadores en el coste nominal, Kn, para producir el coste ajustado.

Las ecuaciones nominales de coste para el modelo intermedio son

Modo orgánico Kn = 3.2 Sk1.05

Modo semiencajado Kn = 3.0 Sk1.12

Modo empotrado Kn = 2.8 Sk1.20

2.11.7.2 Atributos de Coste

Cada atributo se cuantifica para un entorno de proyecto. La escala es muy baja bajo - nominal - alto - muy alto - extremadamente alto. Dependiendo de la calificación de cada atributo, se asigna un valor para usar de multiplicador en la fórmula el significado de los atributos es el siguiente, según su tipo de:

2.11.7 software

- **RELY**: garantía de funcionamiento requerida al software. Indica las posibles consecuencias para el usuario en el caso que existan defectos en el producto. Va desde la sola inconveniencia de corregir un fallo (muy bajo) hasta la posible pérdida de vidas humanas (extremadamente alto, software de alta criticidad).
- **DATA**: tamaño de la base de datos en relación con el tamaño del programa. El valor del modificador se define por la relación: **D/K**, donde D corresponde al tamaño de la base de datos en bytes y K es el tamaño del programa en cantidad de líneas de código.
- **CPLX**: representa la complejidad del producto.

HARDWARE

- **TIME**: limitaciones en el porcentaje del uso de la CPU.
- **STOR**: limitaciones en el porcentaje del uso de la memoria.
- **VIRT**: volatilidad de la máquina virtual.
- **TURN**: tiempo de respuesta requerido.

PERSONAL

- **ACAP**: calificación de los analistas.
- **AEXP**: experiencia del personal en aplicaciones similares.
- **PCAP**: calificación de los programadores.
- **VEXP**: experiencia del personal en la máquina virtual.
- **LEXP**: experiencia en el lenguaje de programación a usar.

PROYECTO

- **MODP**: uso de prácticas modernas de programación.
- **TOOL**: uso de herramientas de desarrollo de software.

- **SCED**: limitaciones en el cumplimiento de la planificación

El valor de cada atributo, de acuerdo a su calificación, se muestra en la siguiente tabla

Tabla 2.11 Atributos de Coste

			Valor			
Atributos	**Muy bajo**	**Bajo**	**Nominal**	**Alto**	**Muy alto**	**Extra alto**
	Atributos de software					
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de Base de datos		0,94	1,00	1,08	1,16	
Complejidad	0,70	0,85	1,00	1,15	1,30	1,65
	Atributos de hardware					
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
	Atributos de personal					
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje	1,14	1,07	1,00	0,95		
	Atributos del proyecto					
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Utilización de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,22	1,08	1,00	1,04	1,10	

Fuente: (COCOMO, 2013)

2.11.8 Modelo Detallado

Incluye todo lo del modelo intermedio además del impacto de cada conductor de coste en las distintas fases de desarrollo. Presenta principalmente dos mejoras respecto al modelo básico e Intermedio.

Los factores correspondientes a los atributos son sensibles o dependientes de la fase sobre la que se realizan las estimaciones.

Aspectos tales como la experiencia en la aplicación, utilización de herramientas de software, etc., tienen mayor influencia en unas fases que en otras, y además van variando de una etapa a otra.

Establece una jerarquía de tres niveles de productos, de forma que los aspectos que representan gran variación a bajo nivel, se consideran a nivel módulo, los que representan pocas variaciones, a nivel de subsistema; y los restantes son considerados a nivel sistema.

Para nuestro caso el modelo intermedio será el que usaremos, dado que realiza las estimaciones con bastante precisión.

CAPITULO III

MARCO APLICATIVO

En esta etapa de desarrollo del Sistema de Registro al seguro social de la Universidad Pública de El Alto, se aplica la metodología, las normas y técnicas mencionadas en el marco teórico del capítulo II.

3.1 DESCRIPCION GENERAL DE LA UNIDAD DE SEGURO SOCIAL UPEA

La Universidad Pública y Autónoma de El Alto, ubicado en la Avenida Sucre (Villa Esperanza) de la Ciudad de El Alto.

La unidad de seguro social se encuentra ubicada en el edificio emblemático de la UPEA, Actualmente busca dar el mejor servicio al estamento estudiantil.

3.2 OBJETIVO

La unidad del seguro social está sujeta a cumplir los principios, normas y condiciones que regulan los procesos de Administración de la unidad y las obligaciones y derechos que derivan de éstos.

3.3 ORGANIGRAMA DE LA UNIDAD DE SEGURO SOCIAL

El organigrama muestra como la unidad de seguro social está organizada, como se ve en la Figura 3.1

Figura 3.1 Organigrama Seguro Social

Fuente: (Elaboración Propia)

3.4 PROCESO ACTUAL

La dirección de Seguro Social, es una unidad de nuestra universidad (UPEA), que diversifica diferentes especialidades como ser; clínica médica, exámenes médicas y rayos x.

Un servicio de excelencia en atención al estamento estudiantil la cual funciona las 24 horas y los 7 días de la semana.

Misión

Brindar al estamento estudiantil un servicio de excelencia en atención médica a través de un servicio de alto nivel basado en cualificación, experiencia y tecnología; de óptima calidad y al alcance para toda la población universitaria, con un equipo profesional altamente capacitado y con una gran performance profesional.

Visión

Llegar a ser una unidad médica de referencia en nuestra universidad, siendo reconocido por su atención de alta calidad, en sus diagnósticos y tratamientos realizados.

3.5 INGENIERÍA DE REQUERIMIENTOS

Ingeniería de requerimientos es el conjunto de actividades en las cuales, utilizando técnicas y herramientas, se analiza un problema y se concluye con la especificación de una solución (a veces más de una) Entonces:

Las Funciones que un sistema debe realizar se clasifican en tres categorías como se detallan en la tabla 3.1.

Tabla 3.1 Categoría de la Funciones

Categoría de la función	Significado
Evidente	Debe Realizarse y el usuario debería saber que se ha realizado
Oculto	Debe Realizarse, aunque no es visible para los usuarios.

	Esto se aplica a muchos servicios técnicos subyacentes, por ejemplo, guardar información en un mecanismo persistente de almacenamiento
Superflua	Opcionales; su inclusión no repercute de forma significativa en costo ni en otras funciones.

Fuente: (Larman 1999)

3.5.1 Requerimientos Funcionales

Los requerimientos funcionales para el modelado del sistema de Seguro Social se detallan a continuación en la tabla 3.2.

Tabla 3.2 Requerimientos Funcionales

Ref.	Función	Categoría
R.1.1	Acceder al Sistema por tipos de Usuario (Administrador, Limitado e invitado)	Evidente
R.1.2	Registro de Usuario al Sistema	Evidente
R.1.3	Registro de familiares o parentescos	Evidente
R.1.4	Registro de accidentes o incidentes.	Evidente
R.1.11	Reporte de registros por carrera.	Evidente
R.1.12	Registro de Doctores	Evidente

Fuente: (Elaboración Propia)

3.5.2 Requerimientos no Funcionales

Tabla 3.3 Requerimientos No Funcionales

Ref.	Función	Categoría
R.1.1	El sistema debe visualizarse y funcionar correctamente en cualquier Navegador, Edge, Firebird, Mozilla, Chrome,	Evidente
R.1.2	El sistema debe Funcionar en base a la ISO 9126 Métricas para la calidad de tu web	Evidente

R.1.3	El sistema no debe tardar más de diez segundos en mostrar los resultados de una búsqueda. Si se supera este plazo, el sistema detiene la búsqueda y muestra los resultados encontrados	Superflua
R 1.4	El sistema debe tener seguridad en el acceso a la información del sistema	Evidente

Fuente: (Elaboración Propia)

3.6 APLICACION DEL MODELO UWE

3.6.1 Modelo de Casos de Uso

En el presente numeral se plasmará el análisis de requerimientos del sistema mediante el diseño de casos de uso mismos expresados en el comportamiento del sistema frente a las acciones de los actores del mismo, funcionalidades del sistema y además elementos que permiten la abstracción del problema

3.1.1.1 Diagrama de Caso de Uso: General

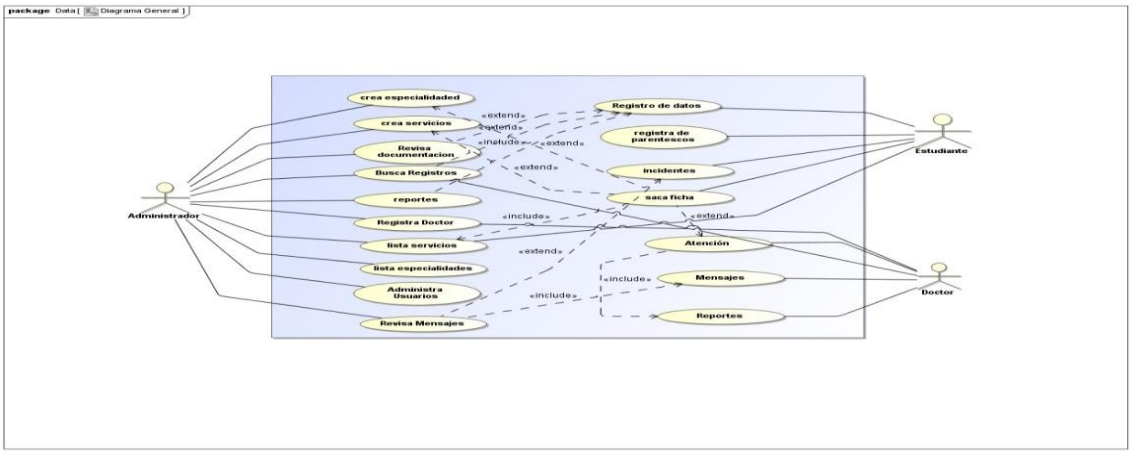


Figura 3.3 Diagrama de Caso de Uso General

Fuente: Basado en (KOCH, 2000)

3.1.1.2 Caso de Uso: Administración.

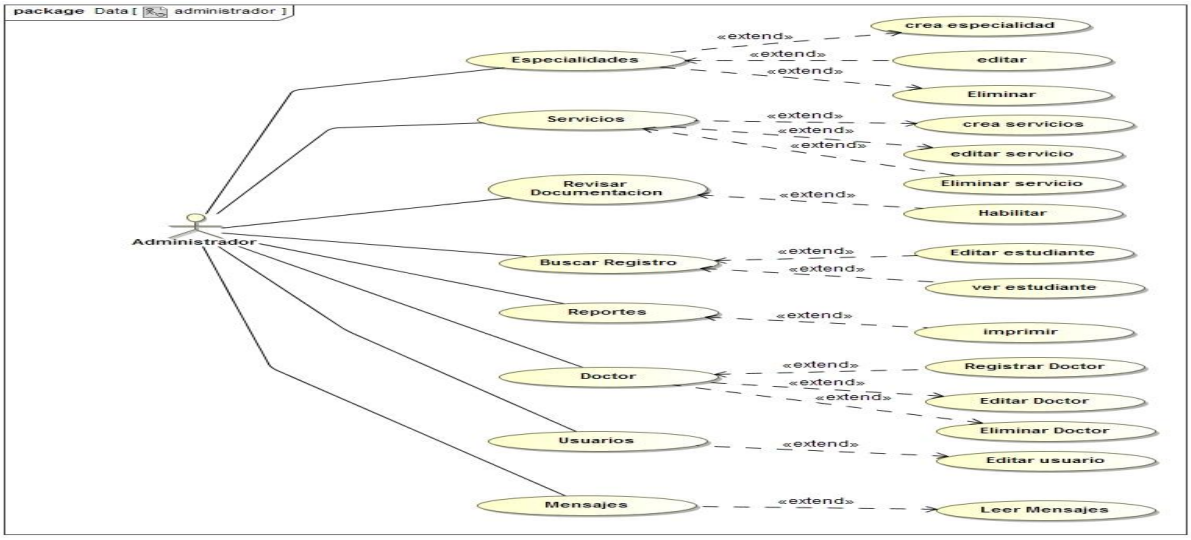


Figura 3.4 Caso de Uso de Administración

Fuente: Basado en (KOCH, 2000)

Tabla 3.4 Especificación del caso de uso de Registrar, Modificar y Eliminar Usuario

Descripción de Caso de Uso 1		
Nombre:	Administración	
Actor:	Administrador	
Descripción:	El Administrador puede registrar, actualizar y eliminar en diferentes módulos.	
Flujo Principal	**Evento Actor**	**Evento Sistema**
	1. Ingresa al sistema	1. Solicita y valida Nombre de Usuario y contraseña.
	2. Ingresa usuario y contraseña	2. Inicia sesión y despliega menú de información, numero de registros al sistema, número de visitas por día y total al sistema, estadista de registros por carrera, doctores registrados y

Flujo Principal		mensajes de valoración por parte de los estudiantes.
	3. Escoge Menú correspondiente	3. Despliega la lista del correspondiente modulo y muestra las acciones de Registrar, Modificar y Eliminar.
	4. Escoge una de las acciones Registrar, Modificar o Eliminar	4. Dependiendo de la opción realiza una acción despliegue de formulario de Registrar o formulario para Modificar o Eliminar Usuario.
	5. Introduce o modifica o elimina un Usuario	5. Valida los formularios y registrar los datos con la opción realizada y retorna al menú principal mostrando un mensaje de confirmación con la acción realizada.
	6. Escoge el módulo de especialidades	6. Se despliega el formulario de crear, editar y realizar el borrado lógico.
	7. Escoge el módulo de servicios	7. Se despliega el formulario de crear, editar y realizar el borrado lógico del Módulo,
	8. Escoge el módulo de Reportes	8. Se despliega el formulario de sacar reportes de registro y atención de doctores.
	9. Escoge el módulo de Usuarios	9. Se despliega el formulario de crear, editar y realizar el borrado lógico de usuarios donde también puede dar privilegios.
	10. Escoge el módulo de Doctores	10. Se despliega el formulario de crear, editar y realizar el borrado lógico de

		doctores,
	11. Escoge el módulo de Buscar	11. Se despliega el formulario de buscar al estudiante registro por su numero de carnet y muestra la información simple del estudiante.
	12. Escoge el módulo de Revisión de documentos.	12. Se despliega el formulario de registro del estudiante donde se habilita al estudiante.
	13. Escoge el módulo de Mensajes	13. Se despliega el formulario de leer los mensajes que provienen del módulo de reportar incidente.

Fuente: (Elaboración propio)

3.1.1.3 Diagrama de Caso de Uso: Estudiante

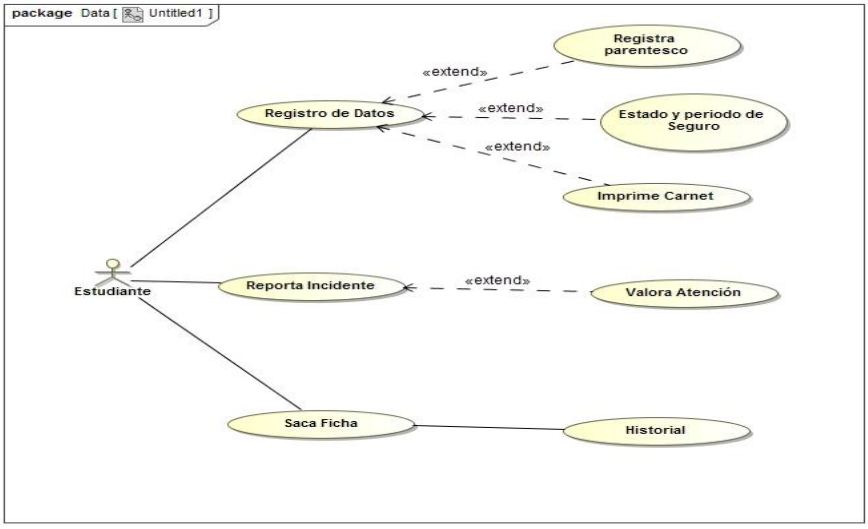


Figura 3.5 Caso de Uso del Estudiante

Fuente: Basado en (KOCH, 2000)

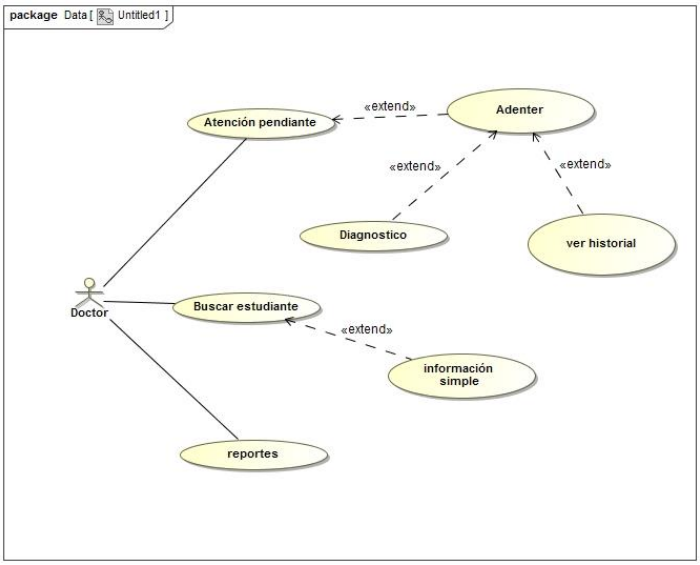
Tabla 3.8 Especificación del caso de uso registro

Descripción de Caso de Uso 1		
Nombre:	Estudiante	
Actor:	Registro del Estudiante	
Descripción:	El Estudiante se registra al sistema, Registra parentesco, Imprime su carnet de registro, saca ficha o servicio disponible.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Ingresa al sistema	1. Solicita y valida Nombre de Usuario y contraseña.
	2. Ingresa usuario y contraseña	2. Inicia sesión y despliega menú de registro de datos, registro de parentesco e imprime su carnet.
	3. Escoge Menú de Reportar incidente.	3. Despliega el Formulario de enviar mensaje y dar la valoración de atención.
	4. Escoge el menú de ficha	4. Se muestra los servicios disponibles, escoge un servicio y se despliega los horarios disponibles.

Fuente: (Elaboración propia)

3.1.1.4 Diagrama de Caso de Uso: Doctor

Figura 3.6 Caso de Uso de



Doctor

Fuente: Basado en (KOCH, 2000)

Tabla 3.14 Especificación del caso de uso Doctor

Descripción de Caso de Uso 7		
Nombre:	Módulo Doctor	
Actor:	Doctor	
Descripción:	Los Doctores atienden las fichas que sacaron los estudiantes diagnostican, dan receta. Ven historial y sacan reportes de atenciones realizados.	
Flujo	Evento Actor	Evento Sistema
	1. Ingresa al sistema	1. Solicita y valida Nombre de Usuario y contraseña.
	2. Ingresa usuario y contraseña	2. Inicia sesión y despliega menú de atenciones

Principal		pendientes.
	3. Escoge Menú de Buscar.	3. Despliega el Formulario de información del estudiante.
	4. Escoge Menú de Reportes.	4. Despliega el Formulario de Reportes general o por fecha.

Fuente: (Elaboración propia)

3.6.2 Modelo Contenido

El diagrama de contenido tiene por propósito mostrar las relaciones entre las entidades y la estructura de los datos que se encuentran alojados en el sistema el modelo de contenido contiene la información relevante almacenada en el sistema como se estructura como se relaciona. Esto se representa mediante un diagrama de clases UML como se muestra en la Figura 3.9

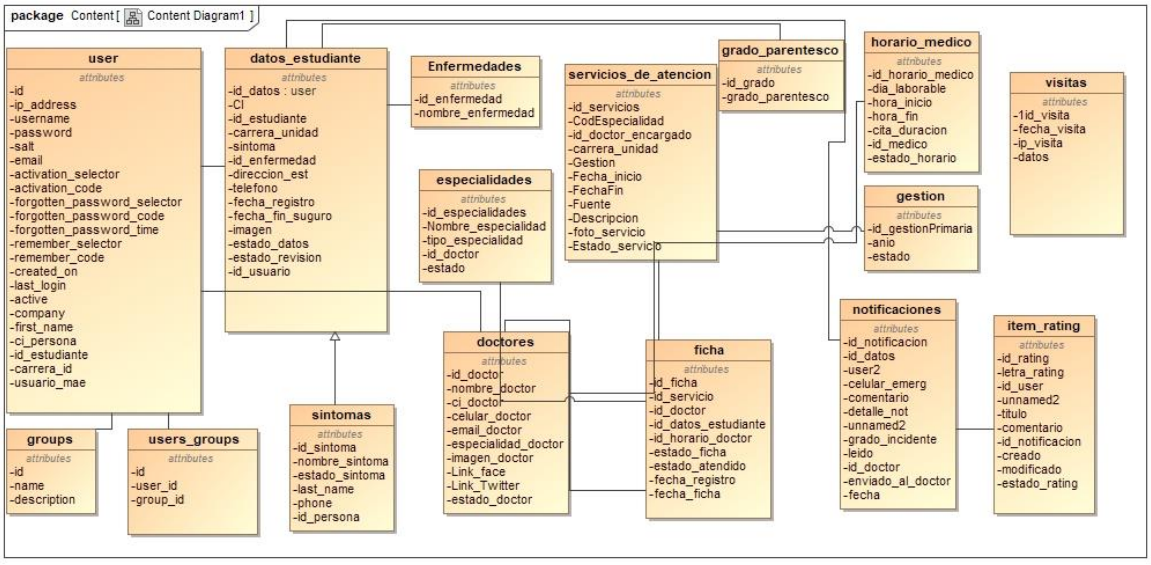


Figura 3.9 Modelo de Contenido de Sistema

Fuente: Basado en (KOCH, 2000)

El modelo muestra la relación entre las entidades del sistema, pudiendo destacar la importancia de la entidad de usuario, solicitud y Ítem solicitud, que son el eje principal del proceso.

3.6.3 Modelo de Navegación

Estos tienen por cometido ilustrar los vínculos lógicos y de navegación entre clases. El modelo de clase navegación y las asociaciones de navegación general del proyecto, que expresa la navegación directa se muestra en la Figura 3.10.

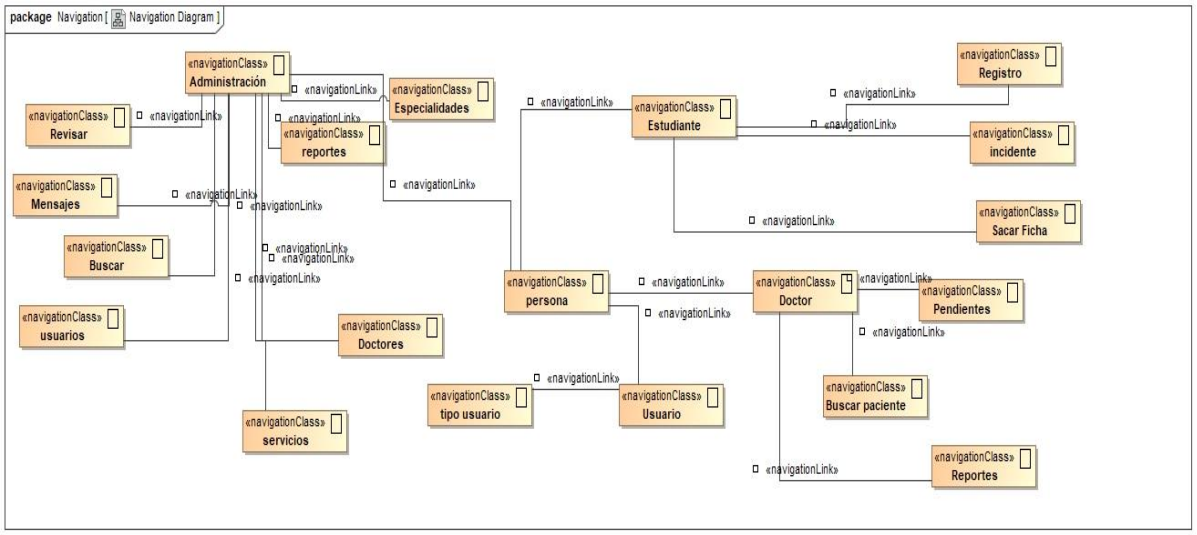


Figura 3.10 Modelo de Navegación del Sistema

Fuente: Basado en (KOCH, 2000)

3.6.4 Modelo de Estructura

Este modelo de estructura de navegación permite ilustrar los vínculos lógicos y de navegación entre clases, menús, índices y clases de proceso para tener una mejor comprensión del sistema como se muestra en las siguientes figuras.

Los usuarios tienen acceso de acuerdo a sus privilegios acceso a la información de los usuarios, administrativos, doctor y estudiantes para realizar los servicios y acceso a los reportes según el tipo de acceso como se muestra en las siguientes Figuras.

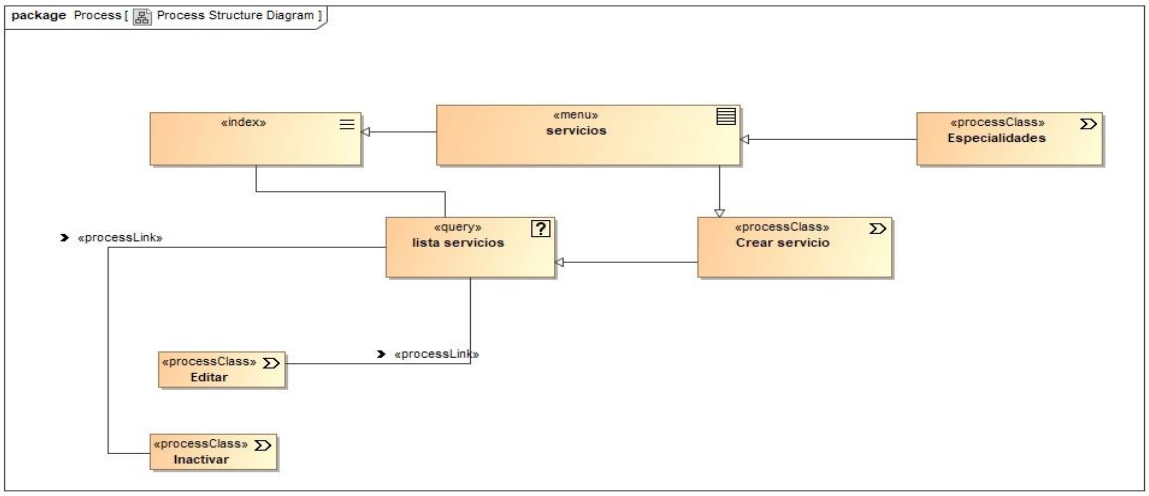


Figura 3.11 Modelo de Estructura de Navegación de Administración de Servicios

Fuente: Basado en (KOCH, 2000)

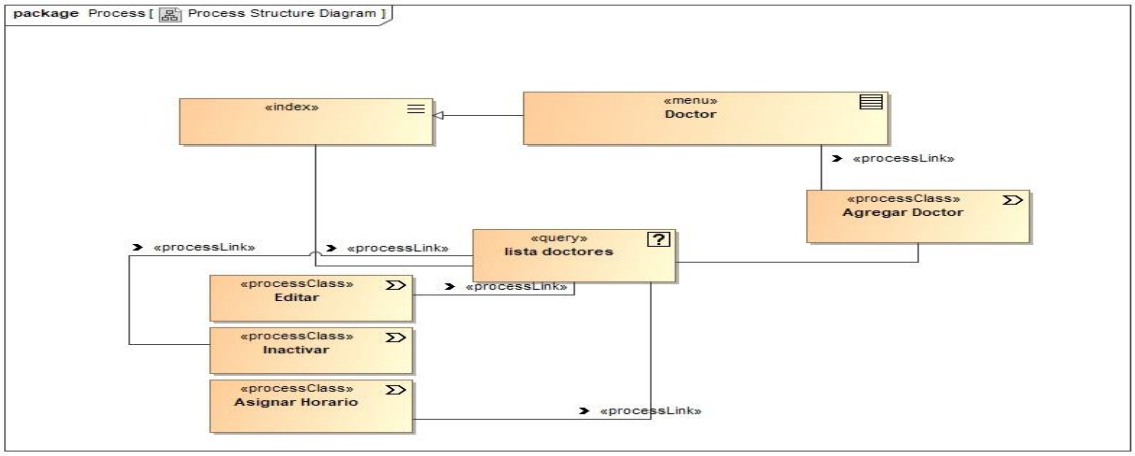


Figura 3.12 Modelo de Estructura de Navegación de Administración de agregar doctor

Fuente: Basado en (KOCH, 2000)

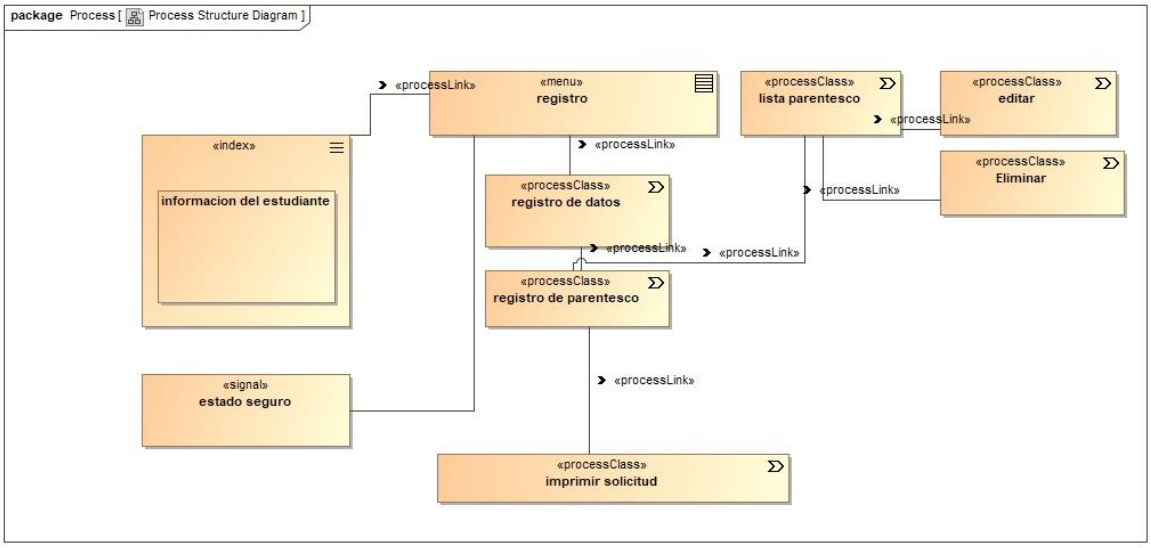


Figura 3.13 Caso de Uso de Registro del Estudiante

Fuente: Basado en (KOCH, 2000)

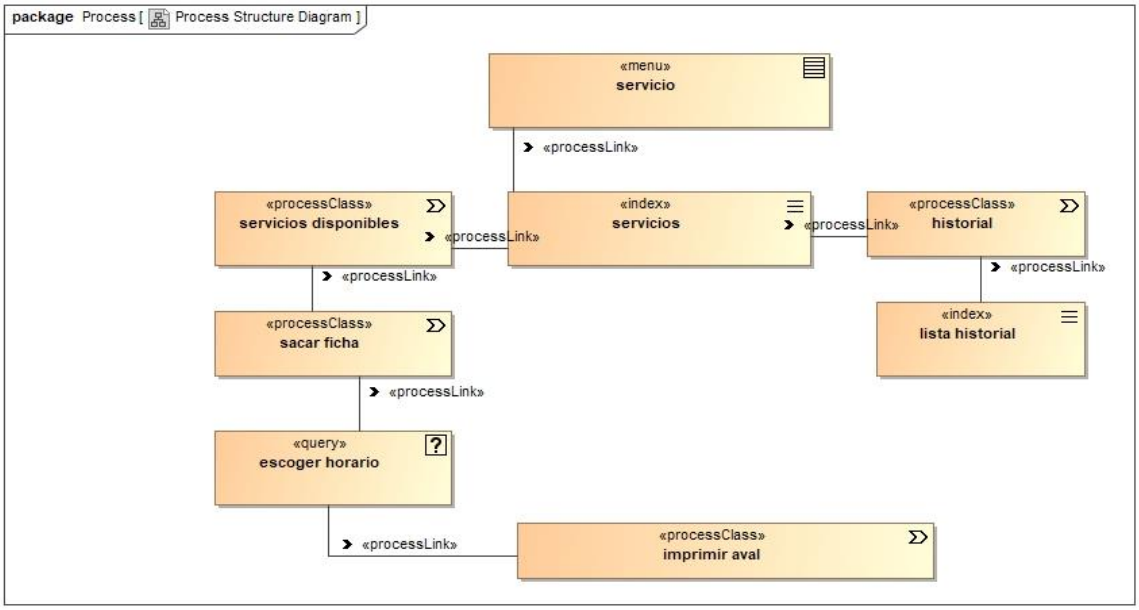


Figura 3.14 Caso de Uso de Sacar Ficha

Fuente: Basado en (KOCH, 2000)

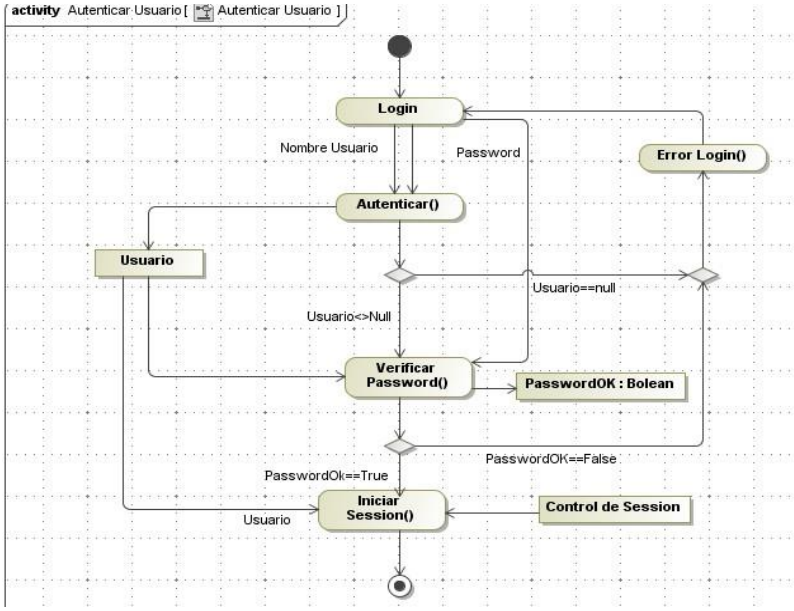


Figura 3.15 Modelo de Flujo de Proceso de Autenticar Usuarios al Sistema

Fuente: Basado en (KOCH, 2000)

3.6.5 Modelo de Presentación

A continuación, se muestran los modelos de presentación para el sistema de la unidad de bienes y servicios, según UWE propone para la construcción de las páginas en forma de bosquejos derivadas desde la Figura 3.14 hasta la Figura 3.19 donde se muestra como los usuarios podrán acceder al sistema mostrado los menús correspondientes según el tipo de usuario.

.

Figura 3.16 Modelo de Presentación de Autenticación del Sistema

Fuente: Basado en (KOCH, 2000)

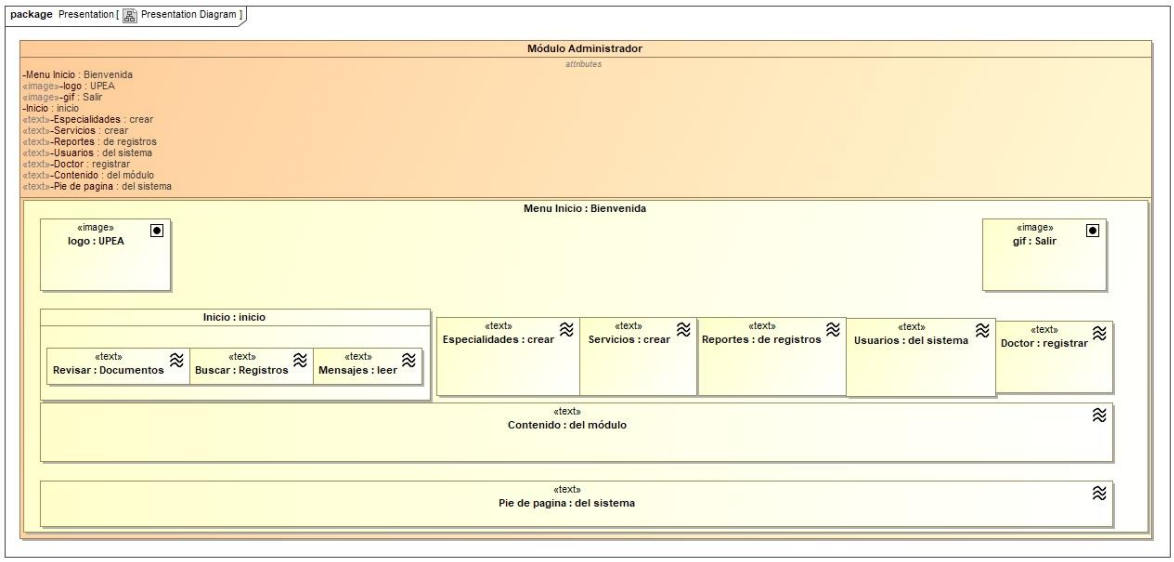


Figura 3.17 Modelo de Presentación del Módulo de Administración del Sistema

Fuente: Basado en (KOCH, 2000)

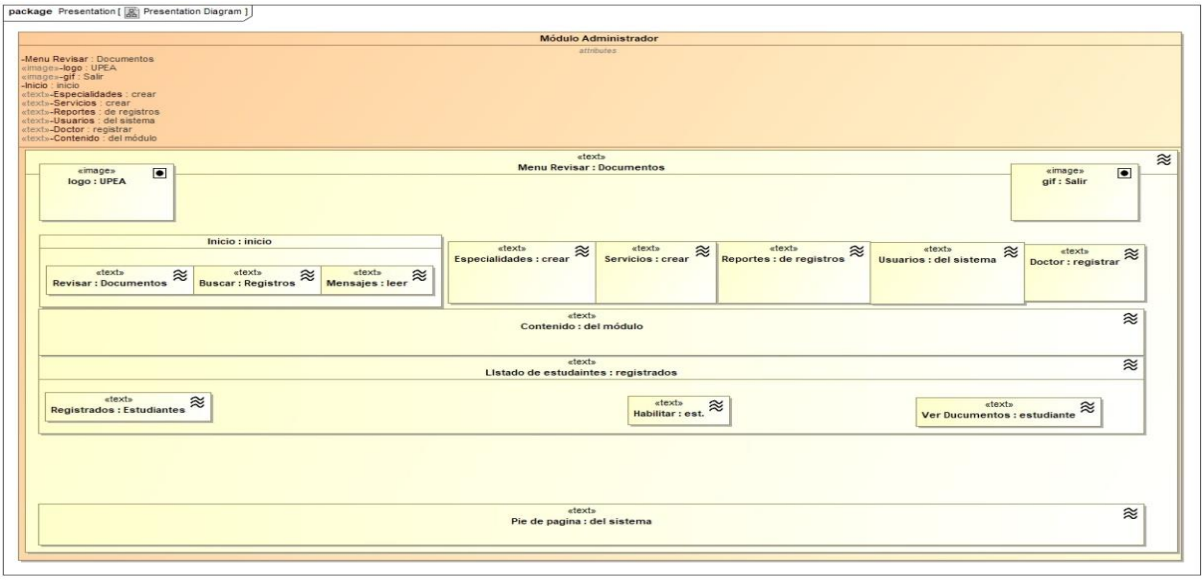


Figura 3.18 Modelo de Presentación del Menú de Administración, Revisar Documentos

Fuente: Basado en (KOCH, 2000)

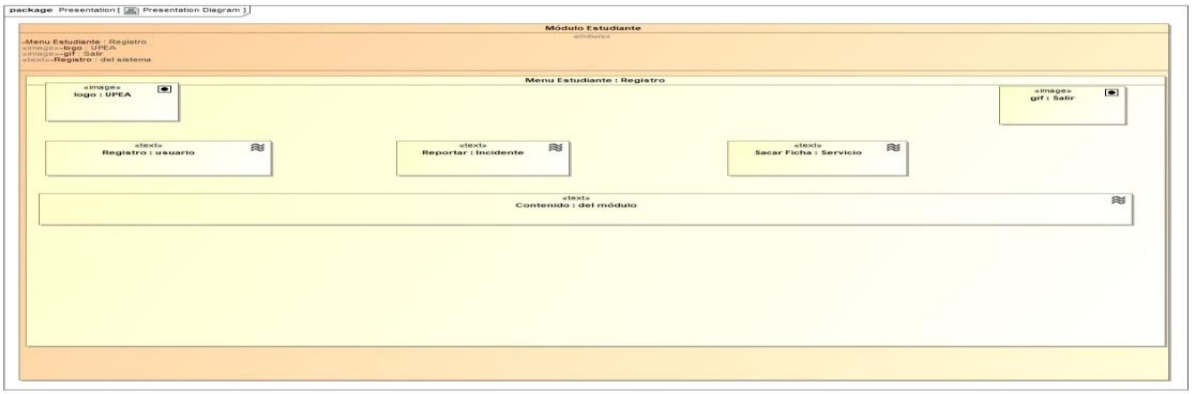


Figura 3.19 Modelo de Presentación de Ingreso del Estudiante

Fuente: Basado en (KOCH, 2000)

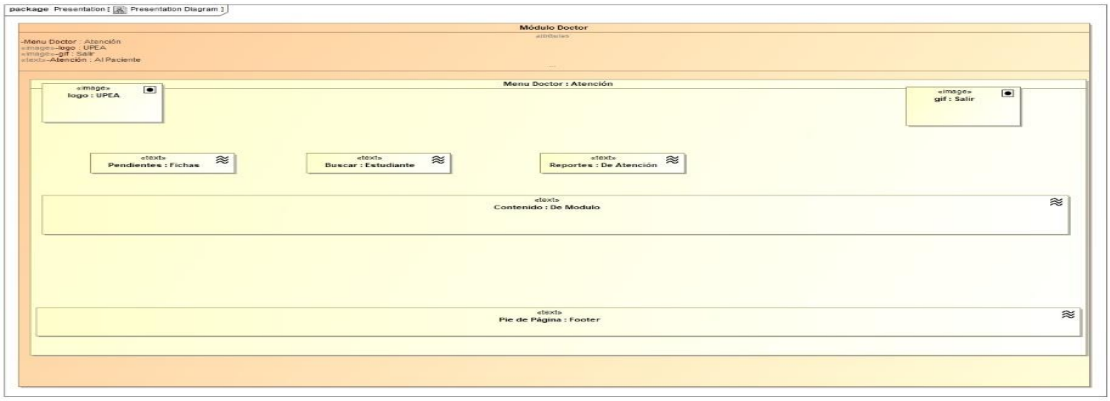


Figura 3.20 Modelo de Presentación de Doctor Para la Atención

Fuente: Basado en (KOCH, 2000)

3.6.6 Modelo Implementación

En esta fase de implementación consiste en mostrar la el desarrollo de la presentación de las interfaces del sistema y sus elementos construidos a partir del diseño del Modelo de Presentación UWE. Como se muestra en las siguientes Figuras.

usuario

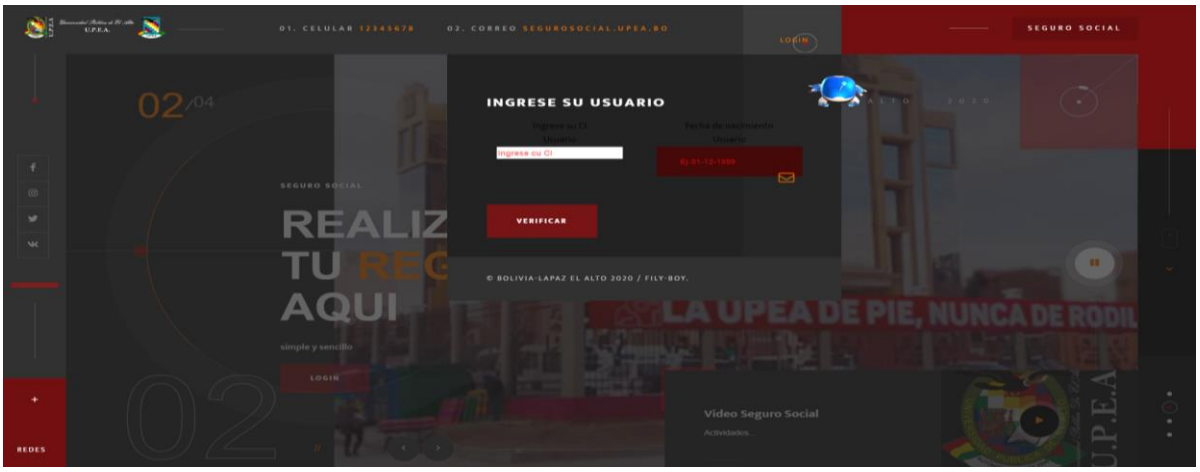


Figura 3.22 Autenticación de Usuarios

Fuente: (Elaboración Propia**)**

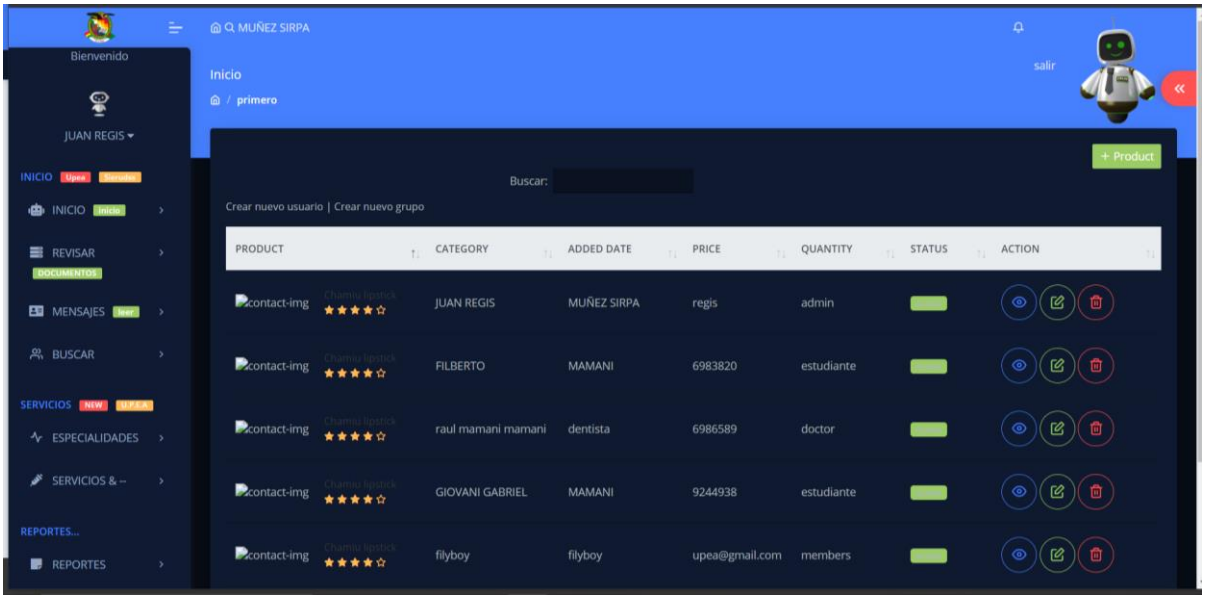


Figura 3.23 Administración de Usuarios

Fuente: (Elaboración Propia**)**

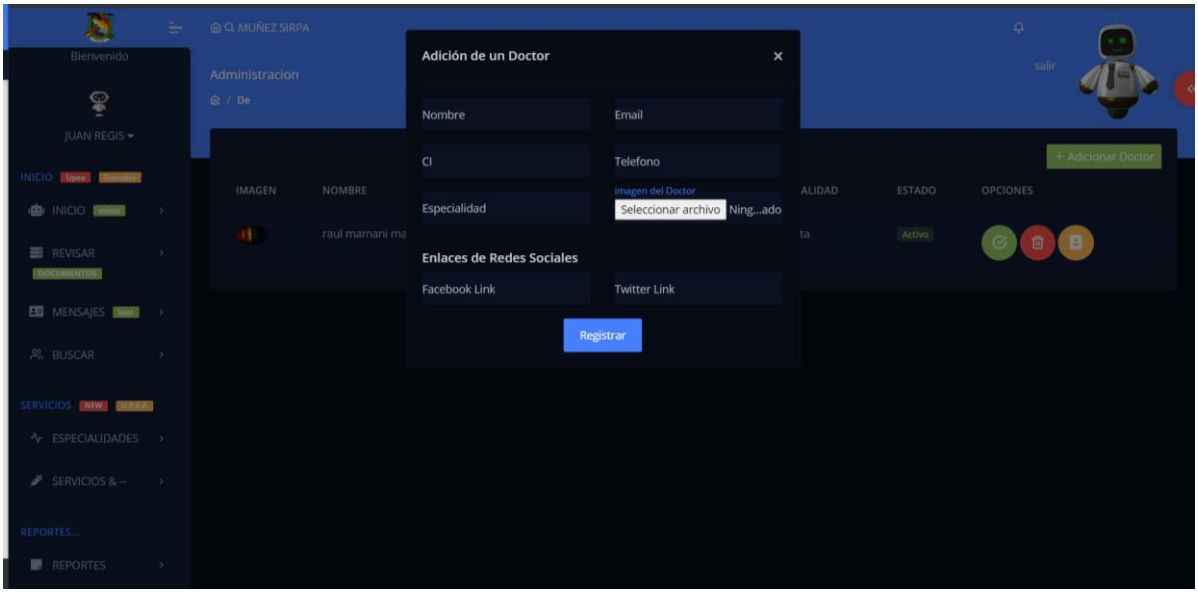


Figura 3.24 Registro de Doctores

Fuente: (Elaboración Propia**)**

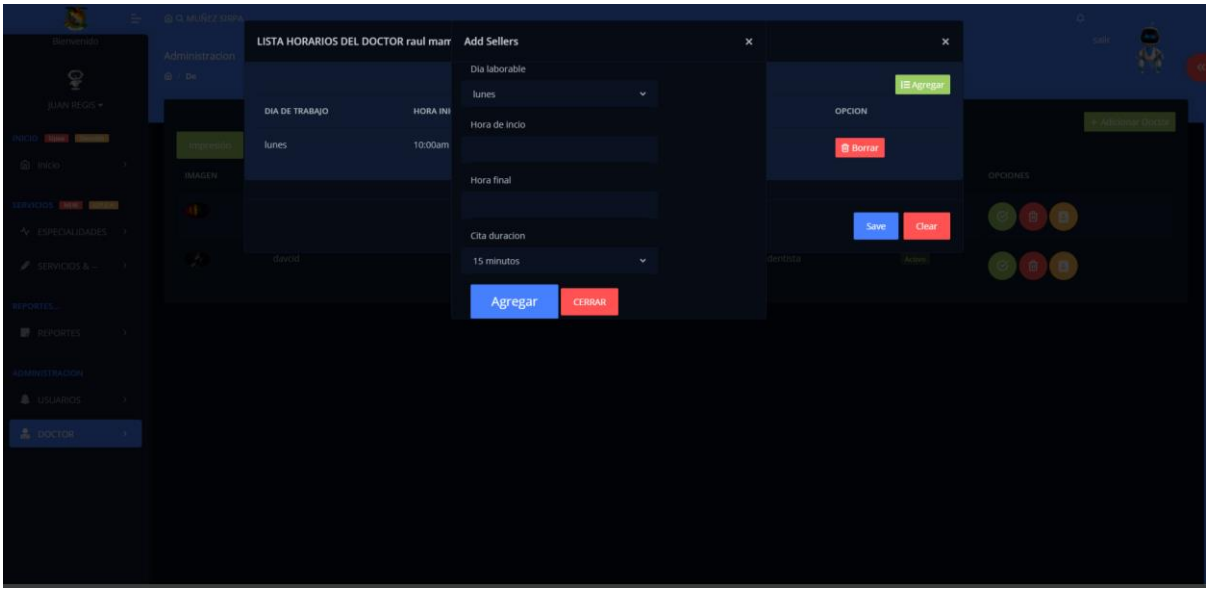


Figura 3.25 Asignar Horarios Al Doctor

Fuente: (Elaboración Propia**)**

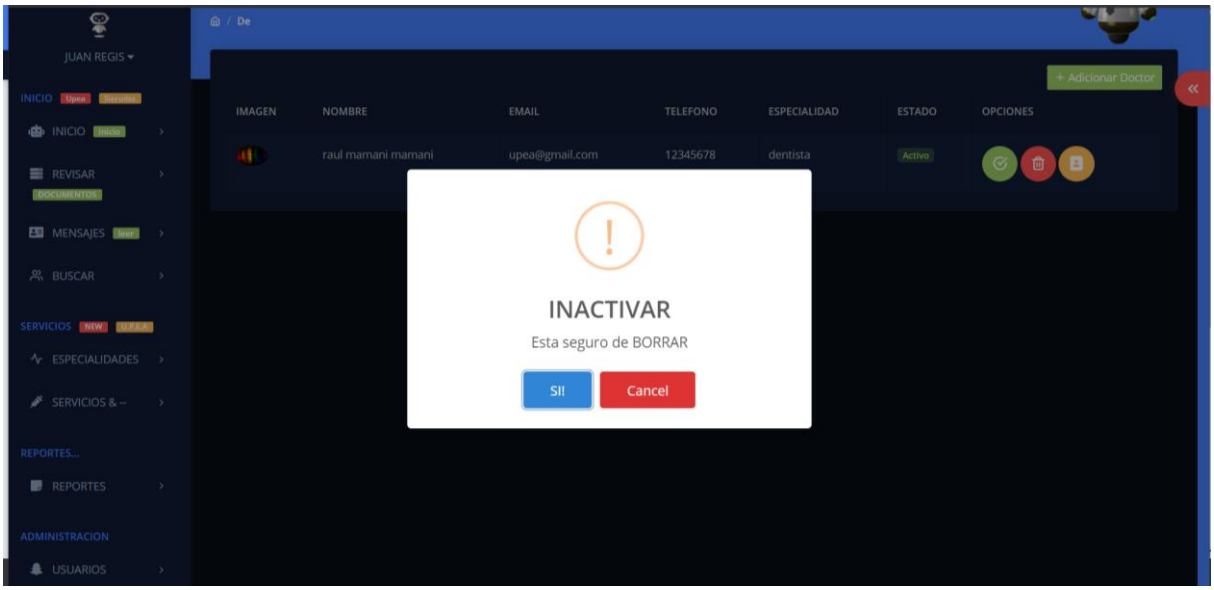


Figura 3.24 Inactivar Doctor

Fuente: (Elaboración Propia**)**

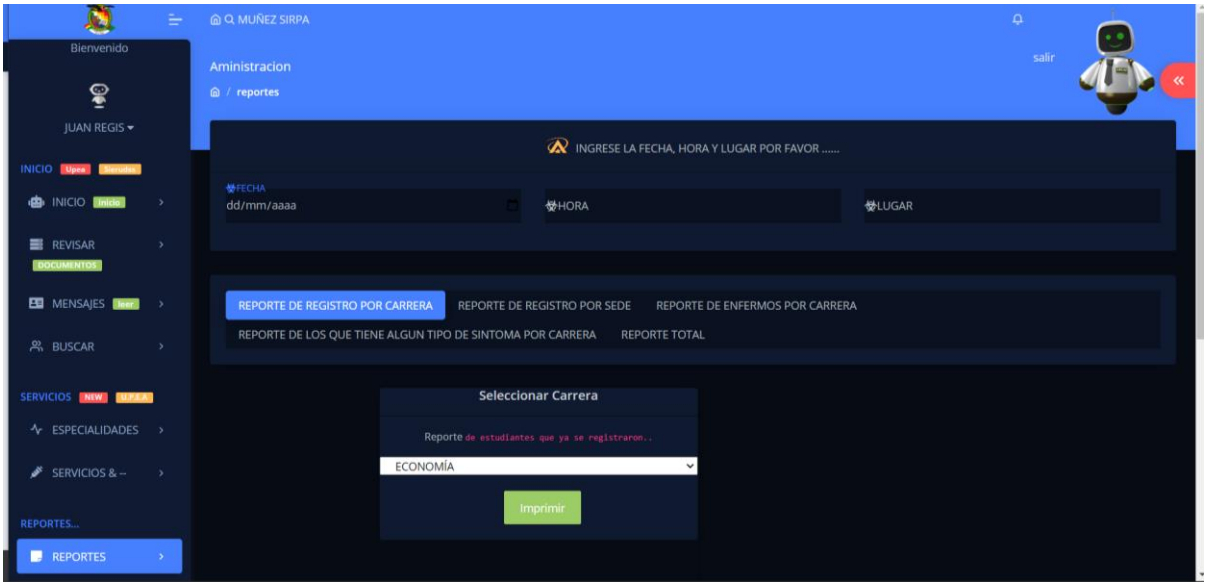


Figura 3.26 Vista Reportes de Registro

Fuente: (Elaboración Propia**)**

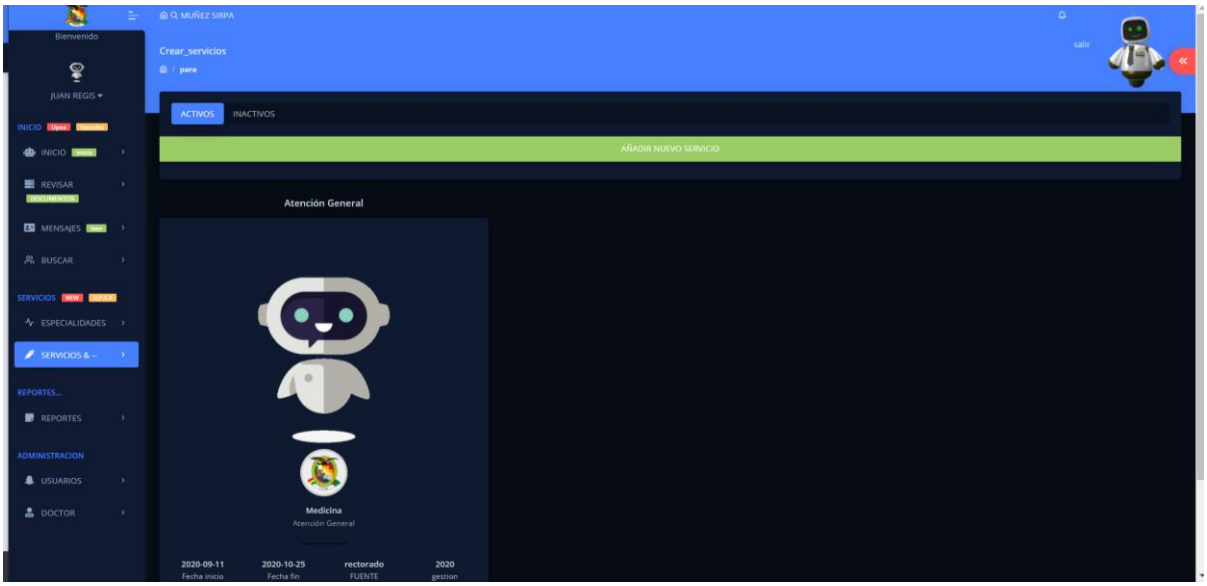


Figura 3.27 Lista Servicios

Fuente: (Elaboración Propia**)**

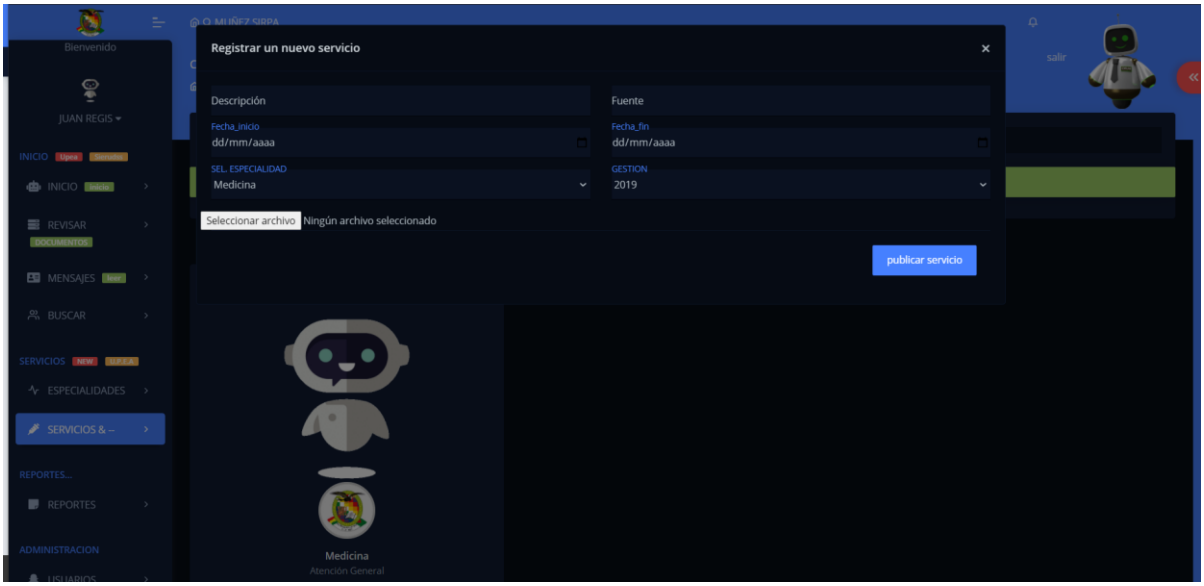


Figura 3.28 Crea un Nuevo Servicio

Fuente: (Elaboración Propia**)**

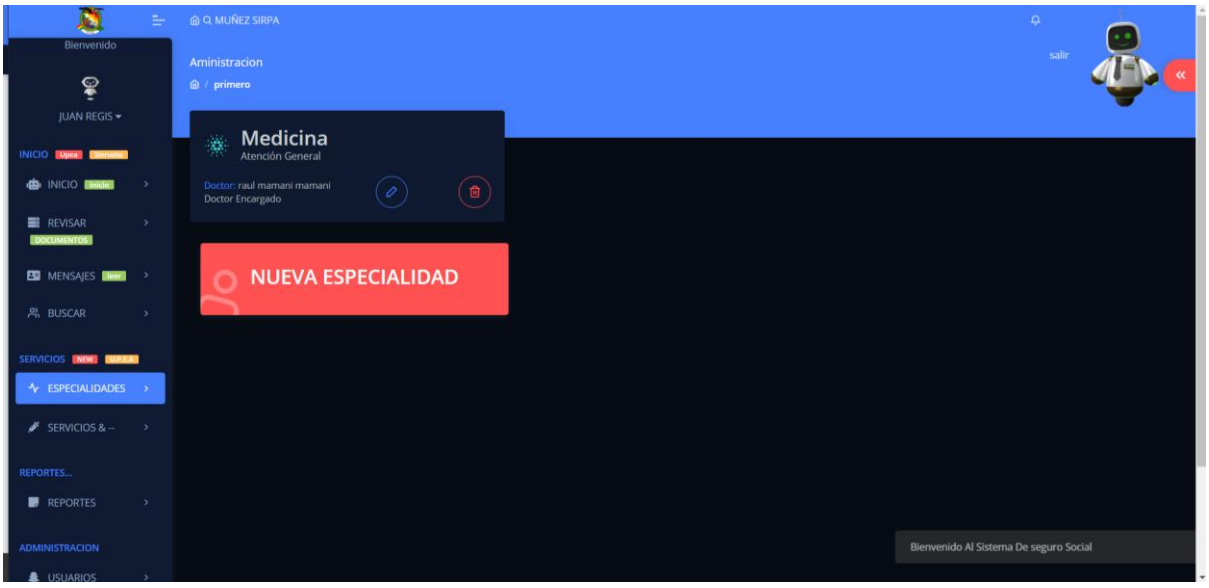


Figura 3.29 Lista Especialidades

Fuente: (Elaboración Propia**)**

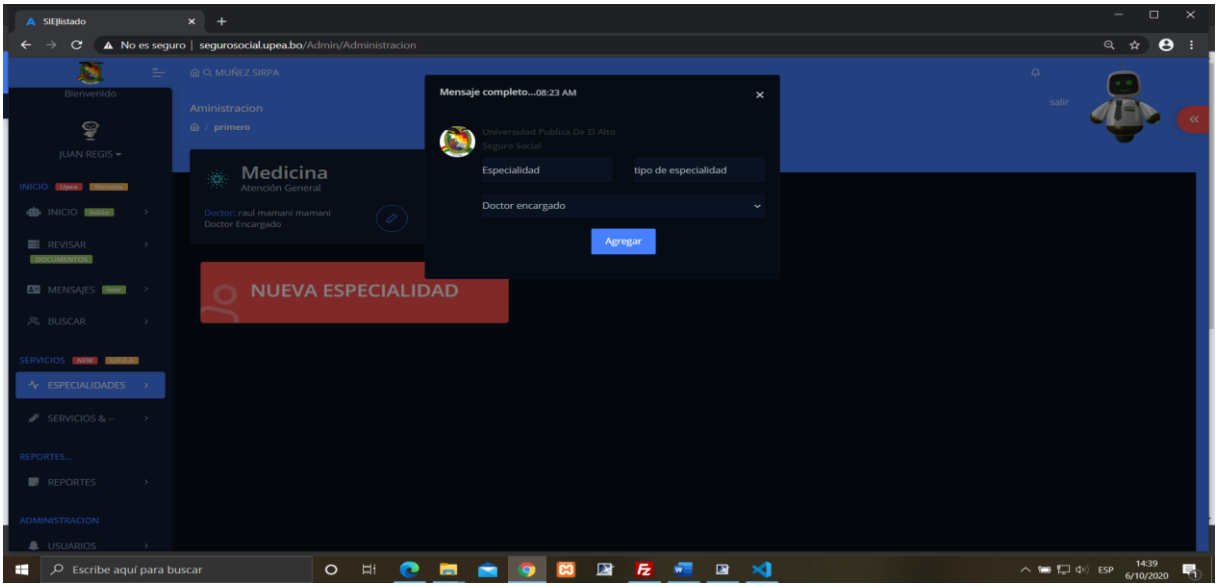


Figura 3.30 Crea Especialidades

Fuente: (Elaboración Propia**)**

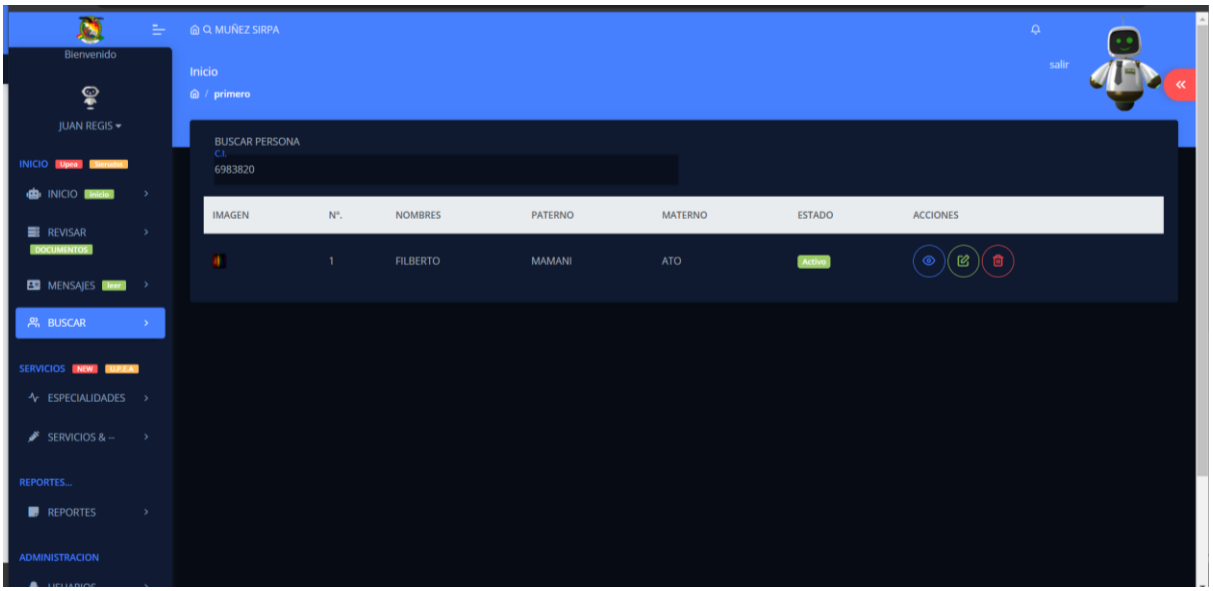


Figura 3.31 Busca Estudiante Registrado

Fuente: (Elaboración Propia**)**

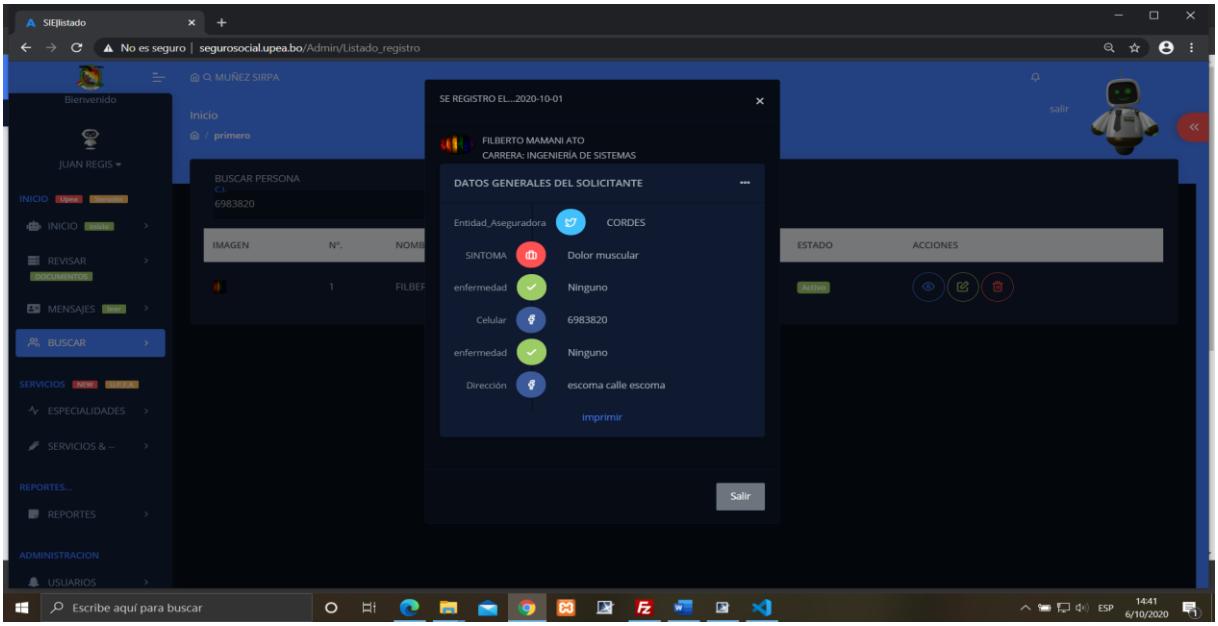


Figura 3.32 Información del estudiante: (Elaboración Propia**)**

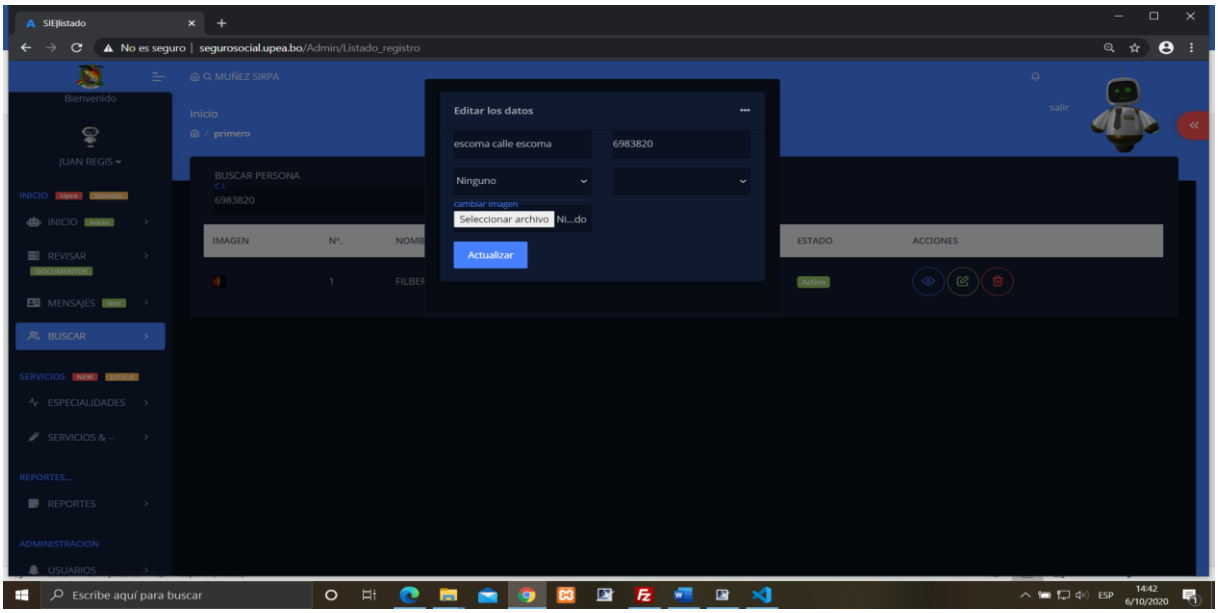


Figura 3.33 Edita Registro

Fuente: (Elaboración Propia**)**

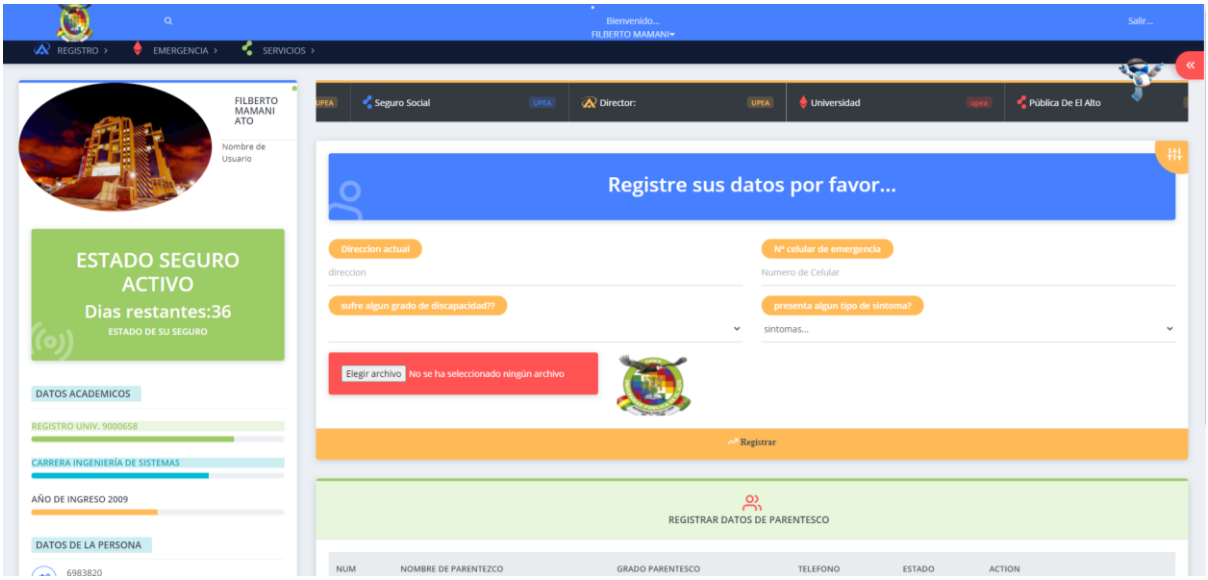


Figura 3.34 Registro de Estudiante

Fuente: (Elaboración Propia**)**

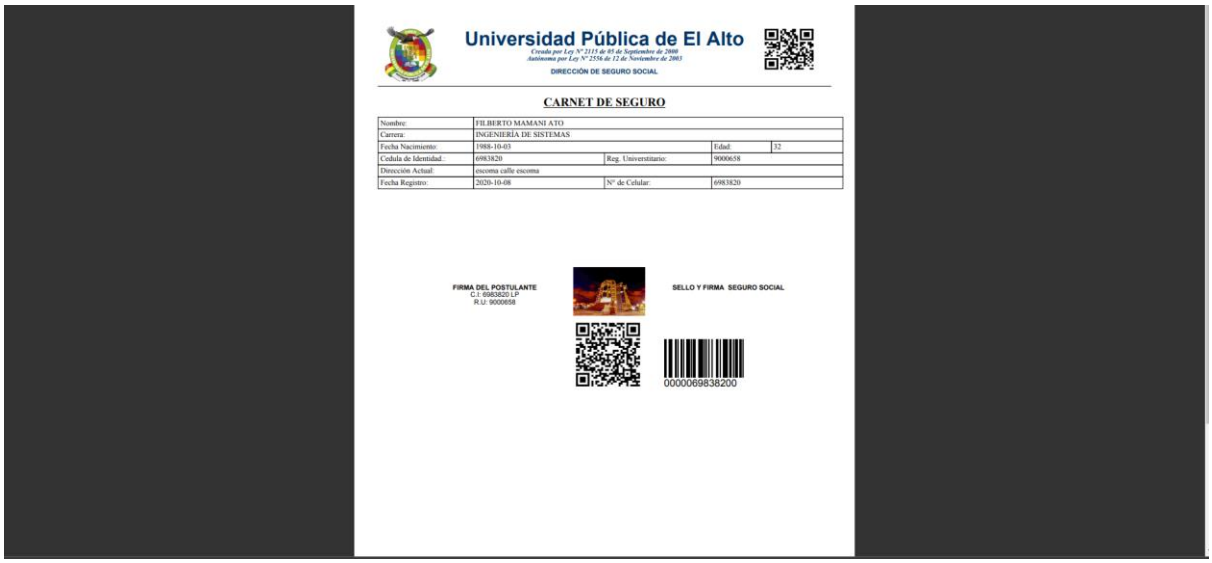


Figura 3.35 Carnet de seguro

Fuente: (Elaboración Propia**)**

Figura 3.36 Reporte de Incidente

Fuente: (Elaboración Propia**)**

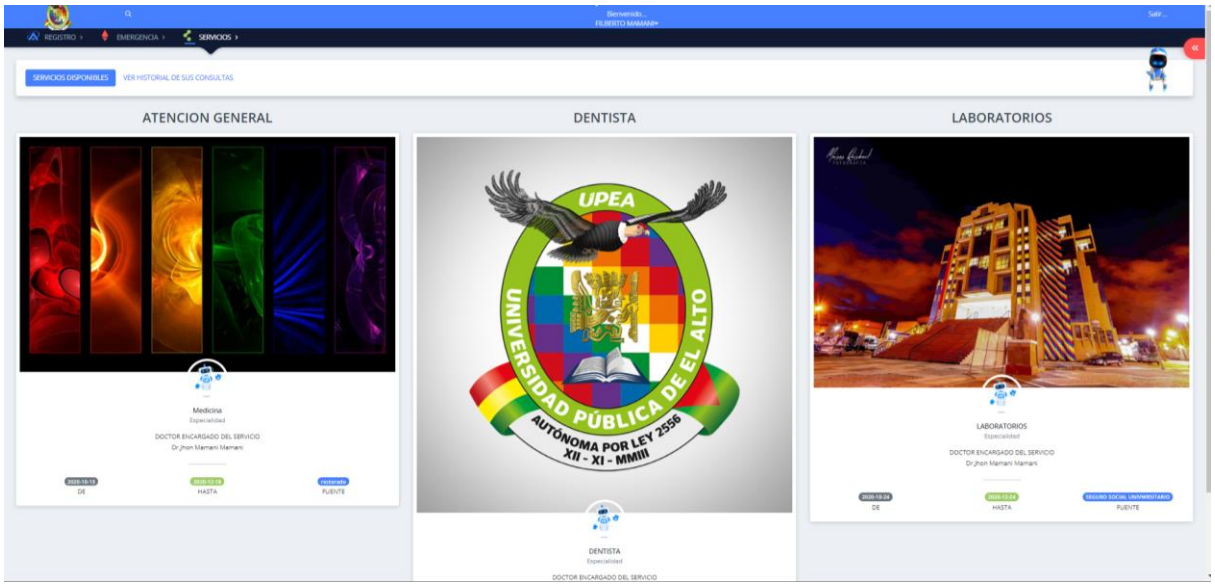


Figura 3.37 Módulo de servicios

Fuente: (Elaboración Propia**)**

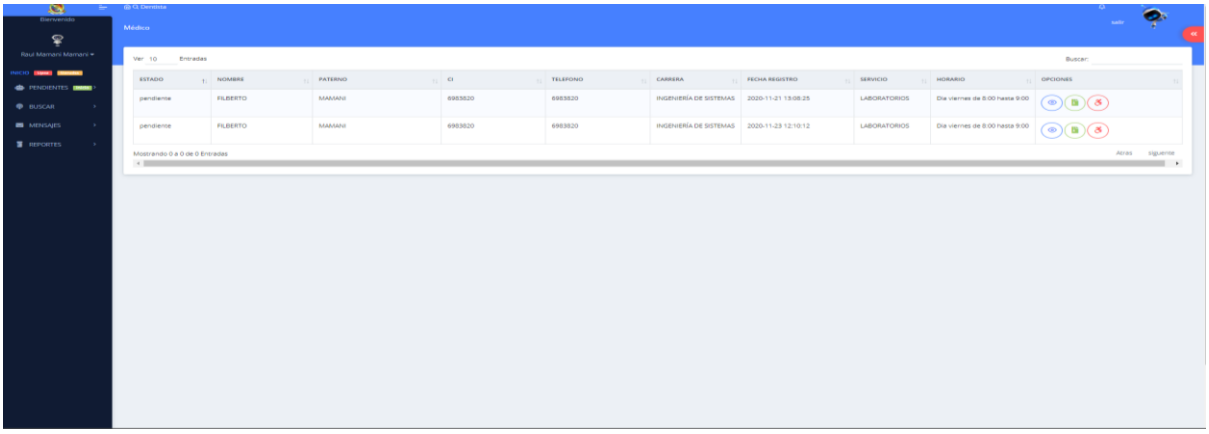


Figura 3.38 Módulo doctor de atenciones pendientes

Fuente: (Elaboración Propia**)**

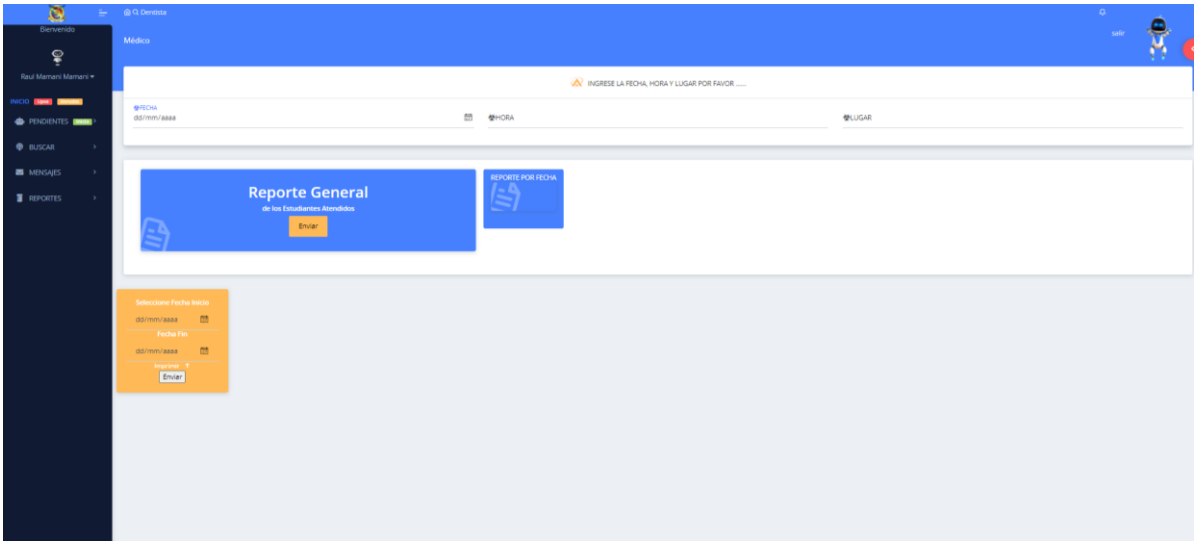
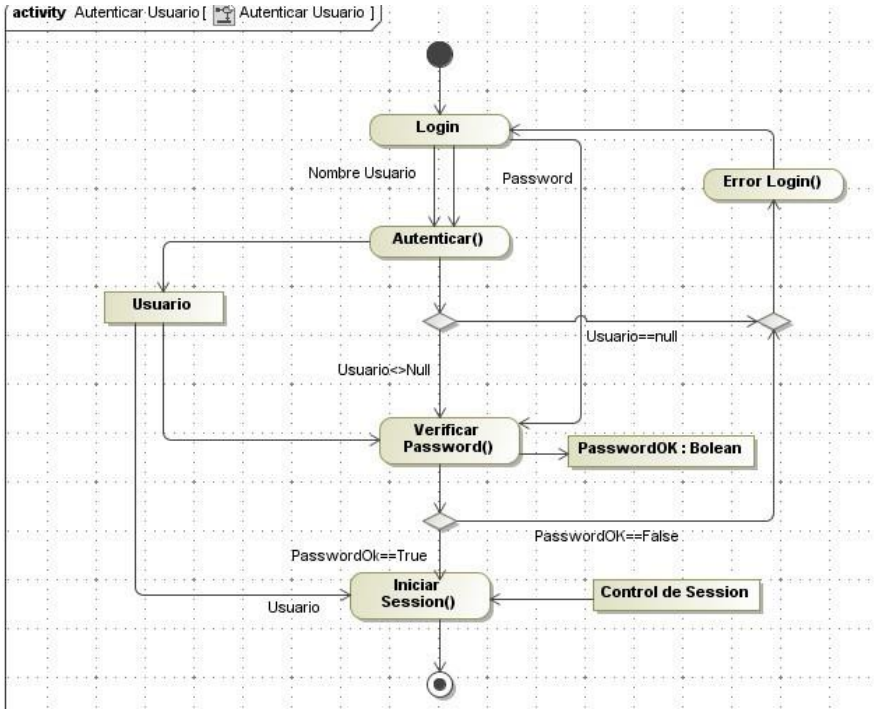


Figura 3.39 Módulo doctor Reportes

Fuente: (Elaboración Propia**)**

CAJA BLANCA



CAJA NEGRA

CAPITULO IV CALIDAD Y SEGURIDAD

3.7 Seguridad del sistema

Para la seguridad del sistema se consideran las siguientes precauciones

- Autenticación de usuarios: los Usuarios deben autentificarse con un usuario y contraseña de 8 caracteres o más el cual es encriptado en 160 bits
- Manejo de tipos de usuario: EL ingreso al sistema es por niveles de usuario.
- Se aplica el paradigma de Modelo Vista Controlador para tener mayor seguridad en el código el cual es un patrón de arquitectura de software.
- Manejo de sesiones
- Encriptación hash en las contraseñas
- Autenticación de Captcha
- Manejo de roles y tipos usuario en el gestor de Base de Datos
- Manejo de Vistas

3.8 CALIDAD DEL SISTEMA

En este numeral se verá el desarrollo de la medición de la calidad del software mediante la métrica iso-9126 define un modelo general de calidad que ayudara a demostrar que el sistema es confiable.

3.8.1 funcionalidad

La funcionalidad no se puede medir directamente por esta razón corresponde derivar medidas directas como es el punto función que cuantifica el tamaño y la complejidad del sistema en términos de las funciones del usuario. Determina las cinco características del dominio de información, teniendo en cuenta su cantidad y se definen de la siguiente forma:

- Número de Entradas de Usuario

- Número de Salidas de Usuario
- Numero de petición de Usuario
- Numero de archivos
- Numero de Interfaces externas

Número de Entradas de usuario, se cuenta cada entrada de usuario que proporciona diferentes datos orientados a la aplicación. Las entradas se deberían diferenciar de las peticiones, las cuales se cuentan de forma separada.

Tabla 3.25 Número de Entradas de Usuario

Entradas de Usuario	
1 Administración de usuarios	2
2 Administración de Doctores	2
3 Administración de estudiantes	20
Total	24

Fuente: (Elaboración Propia**)**

Número de Salidas Usuario, se cuenta cada salida que proporciona al usuario información orientada a la aplicación. En este contexto la salida se refiere a informes, pantallas, mensajes de error, etc. Los elementos de datos particulares dentro de un informe no se cuentan de forma separada.

Tabla 3.26 Número de Salidas Usuario

Salidas de Usuario	
1 Administración de usuarios	3
2 Administración de estudiantes	20
3 Administración de Doctores	2
Total	25

Fuente: (Elaboración Propia**)**

Número de Peticiones de usuario, una petición se define como una entrada interactiva que produce la generación de alguna respuesta del software inmediata en forma de salida interactiva. Se cuenta cada petición por separada.

Tabla 3.27 Número de Peticiones del Usuario

Archivo	
1 administración de Usuarios	2
2 administración de estudiantes	6
3 administración de doctores	2
Total	10

Fuente: (Elaboración Propia**)**

Número de Archivos, se Cuenta archivo maestro Lógico

Tabla 3.28 Numero de Archivos

Archivo		
1 administración de usuarios		4
2 administración de doctores		3
3 administración de estudiantes		14
Total		21

Fuente: (Elaboración Propia**)**

Numero de interfaces Externas, se cuenta todas las interfaces legibles por la máquina.

Tabla 3.29 Numero de Interfaces Externas

Archivo			
1	Internet		2
2	Intranet		2
		Total	4

Fuente: (Elaboración Propia**)**

Para realizar el cálculo de la cuenta Total con factores de ponderación se debe realizar con la siguiente Tabla 3.30

Tabla 3.30 Factores de Ponderación

Parámetros de medida		Cuenta	Factores de Ponderación			Total
			Simple	Medio	Complejo	
1	Nro. de Entradas de Usuario	24	3	4	6	144
2	Nro. de Salidas de Usuario	25	4	5	7	175
3	Nro. de Peticiones de Usuario	21	3	4	6	126
4	Nro. de Archivos	17	7	10	15	255
5	Nro. de Interfaces Externas	4	5	7	10	40
					Total	740

Fuente: (Elaboración Propia**)**

Valores de ajuste de complejidad según las respuestas a las siguientes preguntas que se muestra en la siguiente tabla 3.31

Tabla 3.31 Valores de Ajuste de Complejidad

Nro.	Factor de Complejidad	Sin Influencia Incidental	Moderada	Medio Significativa	Esencial	Fi
		0 1	2	3	4	5	
1	¿Requiere el sistema copia de seguridad y recuperación?					X	5
2	¿Requiere comunicación de datos?					X	5
3	¿Existen funciones de procesos distribuidos?			X			3
4	¿El rendimiento es critico				X		4
5	¿Sera ejecutado el sistema en entorno existente y fuertemente utilizado?					X	5
6	¿Entrada de datos EN LINEA?					X	5
7	¿Requiere la entrada de datos interactiva que las transiciones de entrada se llevan a cabo sobre múltiples pantallas o variadas opciones?					X	5
8	¿Se actualizan los archivos maestros de forma interactiva?					X	5
9	¿Son complejas de las entradas de salidas de archivos?			X			3
10	¿Lógica del proceso Interno Compleja?			X			3
11	¿Se diseña el código para ser reutilizable?					X	5
12	¿Están incluidas en el diseño conversiones de instalación?		X				2

13	¿instalaciones Múltiples?	X				2
14	¿Facilidad de Cambios?				X	5
	Factor de Complejidad Total (FCT)					57

Fuente: (Elaboración Propia**)**

Para calcular los puntos función (PF), utilizaremos la relación siguiente

$$\mathbf{PF= Cuenta\ Total\ *(0.65+0.01*\sum Fi)}$$

Dónde:

Cuenta Total: Nivel de complejidad del sistema con respecto al usuario

$\mathbf{(0.65+0.01*\sum Fi)}$: Ajuste de complejidad según el dominio de la información.

0.01: Factor de conversión, es decir un error de 1%

0.65: Valor mínimo de ajuste

Calculando el punto función de según la ecuación:

PF = 740*[0.65+0.01*57]

PF=902.8

Si calculamos al 100% el nivel de confianza consideramos la sumatoria de Fi=70 como el máximo valor de ajuste de complejidad se tiene:

$$\mathbf{PF\ max=Cuenta\ Total\ *[1+0.01*\sum Fi]}$$

PF = 740*[0.65+0.01*70]

PF=999

Con los máximos valores de ajuste de complejidad se tiene que la funcionalidad real es:

Funcionalidad=$\frac{902.8}{999}$ =0.90

$$\mathbf{Funcionalidad = 0{,}90 * 100 = 90\%}$$

Entonces la funcionalidad del sistema es un 90% esto quiere decir que el sistema tiene un 90% que funcione sin riesgos de fallo y operatividad constante y 10% de colapso de sistema.

3.8.2 confiabilidad

La confiabilidad del sistema se define como la probabilidad de operación libre de fallos de un programa de computadora en un entorno determinado y durante un tiempo específico.

Para calcular confiabilidad del sistema se toma en cuenta el periodo de tiempo en el cual se ejecuta y se obtiene muestras

$F(t)=\boldsymbol{f}*\boldsymbol{e}(-\mu*\boldsymbol{t})$

En el inicio de ejecución t_0=0 lo que significa el tiempo inicial en el cual dará inicio el funcionamiento del sistema.

$F(0)=\boldsymbol{f}*\boldsymbol{e}(-\mu*\boldsymbol{t0})$

Se observa el trabajo del sistema hasta que produce una falla en el instante T, el cual se aproxima a una variable aleatoria continua.

Como se aproxima a variables aleatorias continuas, la confidencialidad a ser obtenida en términos probabilísticos.

Entonces el término en el cual el sistema trabaja sin falla está dado por la ecuación (2) y tiempo en el cual no falla el sistema está dado por ecuación (3).

P(T<=t) =F(t) (2) probabilidad de fallas

P(T<=t) =1-F(t) (3) Probabilidad de trabajo sin falla

En un periodo de 20 días como tiempo de prueba se define de cada 10 ejecuciones 1 falla

Conociendo la funcionalidad del 90% del sistema calculamos para el periodo establecido.

$$P(T<=t)=1-F(t)$$

$$F(t) = 1 - 90 + e^{(-\frac{1}{10}+20)}$$

$$F(t) = 1 - 0,122$$

$$F(t)=0,88$$

La confiabilidad del sistema es del 88% en un periodo de 20 días como tiempo de prueba.

3.8.3 Mantenibilidad

El mantenimiento se desarrolla para mejorar el sistema en respuesta a los nuevos requerimientos que la unidad tenga y los reglamentos que está regida por la misma.

El estándar IEE94 sugiere un índice de madurez del software (IMS) que proporciona un indicador en la estabilidad de un producto, se lo determina con la siguiente formula.

$$IMS=\frac{[Mt-(Fa+Fc+Fd)]}{Mt}$$

Mt= Numero de módulos la Versión Actual

Fa= Numero de Módulos en la versión actual que se han añadido

Fc: Numero de Módulos en la versión actual que se han Cambiado.

Fd: Numero de Módulos en la versión anterior que se han borrado en la versión actual.

Calculado el IMS

IMS= [6-(1+0+0)]/6

IMS=0,83

Con lo que podemos decir que el nuevo sistema tiene una estabilidad de 83% que es la facilidad de mantenimiento el 17% restante es el margen de error correspondiente a los cambios y modificaciones efectuados desde el prototipo de la versión actual

Puesto que es un sistema diseñado con los requerimientos actuales con el tiempo surgirán nuevos requerimientos los cuales cambiara el valor índice de madurez del software.

Mantenimiento Correctivo, el sistema presenta diseño modular y es por eso que tolera variaciones en su corrección.

Mantenimiento Adaptativo. Se realiza cuando en la organización se produce algún cambio haciendo que el sistema sufra modificaciones.

El sistema por su programación modular permitirá fácilmente hacer modificaciones en sus módulos o integrar nuevos al sistema.

3.8.4 USABILIDAD

La usabilidad es lo mismo decir facilidad de uso, esta métrica nos muestra el costo de aprender a manejar el producto, lo cual se calcula con la siguiente formula:

$$FU=[(Sum(xi)/n) *100]$$

Tabla 3.32 Ajuste de Preguntas

PREGUNTAS	Respuestas		Ponderación %
	SI	NO	
¿Puede Utilizar con facilidad el sistema?	5	1	83%
¿Puede Controlar operaciones que el sistema solicita?	5	1	83%
¿Las Respuestas del sistema son complicadas?	1	5	83%
¿El Sistema permitió la retroalimentación de información?	6	0	100%
¿El sistema cuenta con interface agradable a la vista?	6	0	100%
¿La respuesta del sistema es satisfactoria?	5	1	83%
¿Le parece complicada las funciones del sistema?	1	5	83%
¿Se hace difícil o dificultoso aprender a manejar el sistema?	1	5	83%
¿Los resultados que proporciona el sistema facilitan el trabajo?	6	0	100%
¿Durante el uso del sistema se produjo errores?	1	5	83%
USABILIDAD			88%

Fuente: (Elaboración Propia**)**

Existe un 88% de comprensión o entendimiento de los usuarios con respecto a la capacidad del sistema.

3.8.5 PORTABILIDAD

Para la portabilidad del sistema se tomará en cuenta dos aspectos como ser a nivel aplicación, nivel hardware la portabilidad la dividimos en dos secciones portabilidad del lado del servidor y portabilidad del lado del cliente.

Portabilidad lado servidor

A nivel sistema de software, El sistema de información Web, para el seguro social es portable bajo los siguientes sistemas operativos de la familia Microsoft

Windows 2000, Windows milenio, Windows XP, Windows Vista, Windows 7, Windows 8, Windows 8.1, Windows 10 y todo el sistema operativo libre de Linux.

A nivel de base de datos se utiliza base de datos creada en MySQL.

A nivel Hardware el sistema de información automatizado para el control de la gestión de registro es portable bajo las siguientes características mínimas de hardware microprocesador Pentium IV RAM de 256 Mb Como mínimo espacio en el disco duro 1Gb como mínimo, monitor SVGA.

Resultados:

De acuerdo a los resultados obtenidos se puede establecer la calidad total del sistema en base a los parámetros medidos anteriormente. La calidad está directamente relacionada con el grado de satisfacción con el usuario que ingresa al sistema.

CARACTERISTICAS	RESULTADOS
Usabilidad	88%
Funcionalidad	90%
Confiabilidad	88%
Mantenibilidad	83%
Evaluación Total	90.4%

[Fuente: Elaboración Propia]

Evaluación de calidad total = 87%

CAPITULO V

COSTO BENEFICIO

3.9 ANALISIS DE COSTOS

Para calcular el esfuerzo, necesitaremos hallar la variable KLDC (kilo-líneas de código)

Este proyecto se implementa 5500 Líneas de Código en el lenguaje PHP. Aplicando Conversiones se tiene

KLCD= (LCD)/1000

KLCD= 5500/1000

KLDC= 5.500 KLDC

Es un modelo Intermedio y sistema orgánico

Tabla 3.33 Aplicación del Modelo Intermedio

PROYECTO SOFTWARE	a	b	c	d
Orgánico	3,2	1,05	2,50	0,38

Fuente: (COCOMO, 2013)

Para hallar los valores de FAE, se utilizará la tabla de atributos multiplicadores

Tabla 3.34 Calculo de Atributos FAE

Atributos	Valor Muy bajo	Bajo	Nominal	Alto	Muy alto	Extra alto
Atributos de software						
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de Base de datos		0,94	1,00	1,08	1,16	
Complejidad	0,70	0,85	1,00	1,15	1,30	1,65
Atributos de hardware						
Restricciones de tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
Atributos de personal						
Capacidad de análisis	1,46	1,19	1,00	0,86	0,71	

Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje	1,14	1,07	1,00	0,95		
Atributos del proyecto						
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Utilización de herramientas de software	1,24	1,10	1,00	0,91	0,83	
Restricciones de tiempo de desarrollo	1,22	1,08	1,00	1,04	1,10	
		Total	**0,72**			

Fuente: (Elaboración Propia)

Por tanto, nuestro Factor de ajuste será

FAE=0,72

Aplicando y remplazando valores a la fórmula de esfuerzo, se tiene:

$E = a * KLCD^b * FAE$ (Personas/Mes)

$E = 3{,}2 * 5500^{1{,}05} * 0{,}72$

$E = 19{,}5$(Personas Mes)

Cálculo del Tiempo

$T = c * Esfuerzo^{d}$ (Meses)

T = 2,5 $*$ 19,5^{0,38}

T = 7,72(Meses)

Calculo de la Productividad

$PR = \frac{LCD}{Esfuezo}$(Meses)

$PR = \frac{5500}{19,5}$(Meses)

PR = 282(LCD/personas Mes)

Cálculo del personal requerido

$P = \frac{E}{T}$(Personas)

$P = \frac{19,5}{7,72}$(Personas)

P = 2.52(Personas) equivalente a 3 personas

Costo Total del Proyecto

(Coste Mes) = P* Salario medio entre los programadores y analistas

Costo Persona Mes 300$

Coste Mes= 3*350=900$us

Costo Total= 900*7=6.300$us

En resumen, se requiere 3 personas estimando un trabajo de 7 meses y con costo total de 6.300$us Equivalente en bolivianos a 44.100 Bs.

CAPITULO VI

CONCLUSIONES Y RECOMENDACIONES

En este capítulo se dará las conclusiones y recomendaciones pertinentes del Proyecto de Grado.

4.1 CONCLUSIONES

Se concluyó con los objetivos planteados en el presente proyecto, desarrollando un Sistema de Información Automatizado con Arquitectura Cliente/Servidor que coadyuve a la unidad de seguro social; en el manejo de datos de manera centralizada del estamento estudiantil;

- Puesto que los estudiantes de la UPEA necesitan tener una información al instante.
- Aplicando con éxito la las normas de calidad, metodología UWE, paradigma de Modelo Vista Controlador con PHP7 y las herramientas de programación para que tenga alta usabilidad y funcionalidad.

4.2 RECOMENDACIONES

En base a las políticas de seguridad propuesta y las observaciones realizadas durante las pruebas se elabora las siguientes recomendaciones.

Mantener un control estricto acerca de la seguridad física en el ambiente

La seguridad lógica debe estar a cargo del administrador del servidor web debe incluir el uso de contraseñas, la administración de puertos y servicios del servidor, el control de archivos y contenidos, la ejecución Firewall y la administración de la base de datos

Se recomienda a la unidad seguro social se desarrollen en base al sistema actual propuesto para tener un sistema integral de toda la información correspondiente de los estudiantes de la universidad.

BIBLIOGRAFÍA

- STAIR, Ralph M. Principios de Sistemas de Información: Enfoque Administrativo, Editorial Thomson Editores, 2000.
- Programador PHP de Eugenia Bahit se distribuye bajo una Licencia Creative Commons Atribución-NoComercial-SinDerivadas 3.0 Unported, ®2010-2012 Eugenia Bahit Buenos Aires, Argentina.
- CodeIgniter was created by EllisLab and is now a project of the British Columbia Institute of Technology © Copyright 2014 - 2018, British Columbia Institute of Technology. Last updated on Jun 12, 2018.
- Administración y Gestión de un Servidor Web Apache, GNU General Public License: http://www.gnu.org/copyleft/gpl.html GPL Versión 2, June 1991 Copyright © 1989, 1991 Free Software Foundation, Inc.
- Pérez Valdés, D. (2 de noviembre de 2007). *Los diferentes lenguajes de programaciónparalaweb*.Obtenidodemaestrosdelweb:http://www.maestros delweb.com/los-diferentes-lenguajes-de-programacion-para-la-web/. Recuperado 28 de marzo de 2015.
- *COCOMO.* (octubre de 2013). Recuperado el 12 de diciembre de 2013, de http://es.wikipedia.org/wiki/COCOMO
- Alvarez, M., López, D., & Gutierrez, M. (17 de 01 de 2013). *Taller de PHP.* Obtenido de http://www.desarrolloweb.com/manuales/6/
- Barrera Juarez, C. A. (26 de septiembre de 2011). *METODOLOGIAS PARA EL DESARROLLO DE APLICACIONES WEB*. Recuperado el 08 de noviembre de 2013, de http://www.slideshare.net/ARCANGEL2032/metodologias-para-el-desarrollo-deaplicaciones-web-9419415#btnNext

Chavez, K. (29 de noviembre de 2012). *Ingeniería Web*. Recuperado el 08 de diciembre de 2013, de http://kevinchavez93.blogspot.com/2012/11/ingenieria-web.html

Contraloría, G. d. (2013). Sistema de Administración de Bienes y Servicios. La Paz, Murillo, Bolivia.

Cooper, R. (2004). *Apache Práctico* (1ra Edicion ed.). Madrid: Anaya Multimedia.

CUATRORIOS. (2011). *Norma ISO-9126 para análisis de software*. Recuperado el 10 de Diciembre de 2013, de http://www.cuatrorios.org/index.php?option=com_content&view=article&id=163:normaiso-9126-para-an%C3%A1lisis-de-software&catid=39:blogsfeeds

EcuRed. (2007). *Características de MagicDraw*. Recuperado el 12 de Septiembre de 2013, de http://www.ecured.cu/index.php/MagicDraw

EGUILUZ, J. (2013). *INTRODUCCIÓN A CSS*. Recuperado el 10 de 01 de 2013, de http://librosweb.es/css/

EGUILUZ, J. (2013). *INTRODUCCIÓN A JAVA SCRIPT.* Recuperado el 10 de 01 de 2013, de http://librosweb.es/javascript/

EGUILUZ, J. (2013). *INTRODUCCION A XHTML.* Recuperado el 10 de 01 de 2013, de http://librosweb.es/xhtml/

EGUILUZ, J. (2013). *INTRODUCIÓN A AJAX.* Recuperado el 10 de 01 de 2013, de http://librosweb.es/ajax/

Engineering, I. f. (07 de 09 de 2012). *UWE - UML - BASED WEB ENGINEERING*. Recuperado el 18 de 07 de 2013, de http://uwe.pst.ifi.lmu.de/

Hall, P. (1981). *Software Engineering Economics.* USA.

HENNICKER, R., & KOCH, N. (01 de 10 de 2001). *A UML-based Methodology for Hypermedia Design*. Recuperado el 07 de 08 de 2013, de http://www.pst.informatik.unimuenchen.de/~kochn/Uml2000.pdf

Hernández Salguera , Á. I. (4 de Diciembre de 2012). *Ingenieria del Software*. Recuperado el 10 de
Octubre de 2013, de http://ingsoftaihs.blogspot.com/

INSTITUTE FOR INFORMATICS REASEARCH UNIT PROGRAMING AND SOFTWARE ENGINEERING. (07 de 09 de 2012). *MAGICUWE - UWE PLUGINPARA MAGICDRAW*. Recuperado el 18 de 07 de
2013, de http://uwe.pst.ifi.lmu.de/toolMagicUWE.html

KOCH, N. (2000). *Software Engineering for Adaptive Hypermedia Systems. Referece Modeling*
Techniques and Development Process. Germany: Ludeing.Maximilians-Universität München.

Leal Castellano, M. Y., Leal Molina, Y. C., & Medina Castiblanco, L. C. (2011). *Taller Cliente Servidor.* Recuperado el 14 de Diciembre de 2011, de http://basesii.wikispaces.com/file/view/Caracter%C3%ADsticas+de+la+arquitectura+Client e.pdf

LEXIVOX. (20 de Febrero de 2013). *Portal Jurídico*. Recuperado el 01 de Diciembre de 2013, de http://www.lexivox.org/norms/BO-DS-N1497.xhtml#

Miguel Ángel, A. (19 de Septiembre de 2012). *Manual de JQuery*. Recuperado el 05 de Octubre de
2013, de http://www.desarrolloweb.com/manuales/manual-jquery.html

MÜNCH, L., & ANGELES, E. (2010). *Métodos y Tánicas de Investigación* (2da. ed.). México: Trillas.

NOMAGIC. (2013). *MAGICDRAW*. Recuperado el 17 de 07 de 2013, de SysML Programas:
http://www.nomagic.com/products/magicdraw-addons/sysml-plugin.html

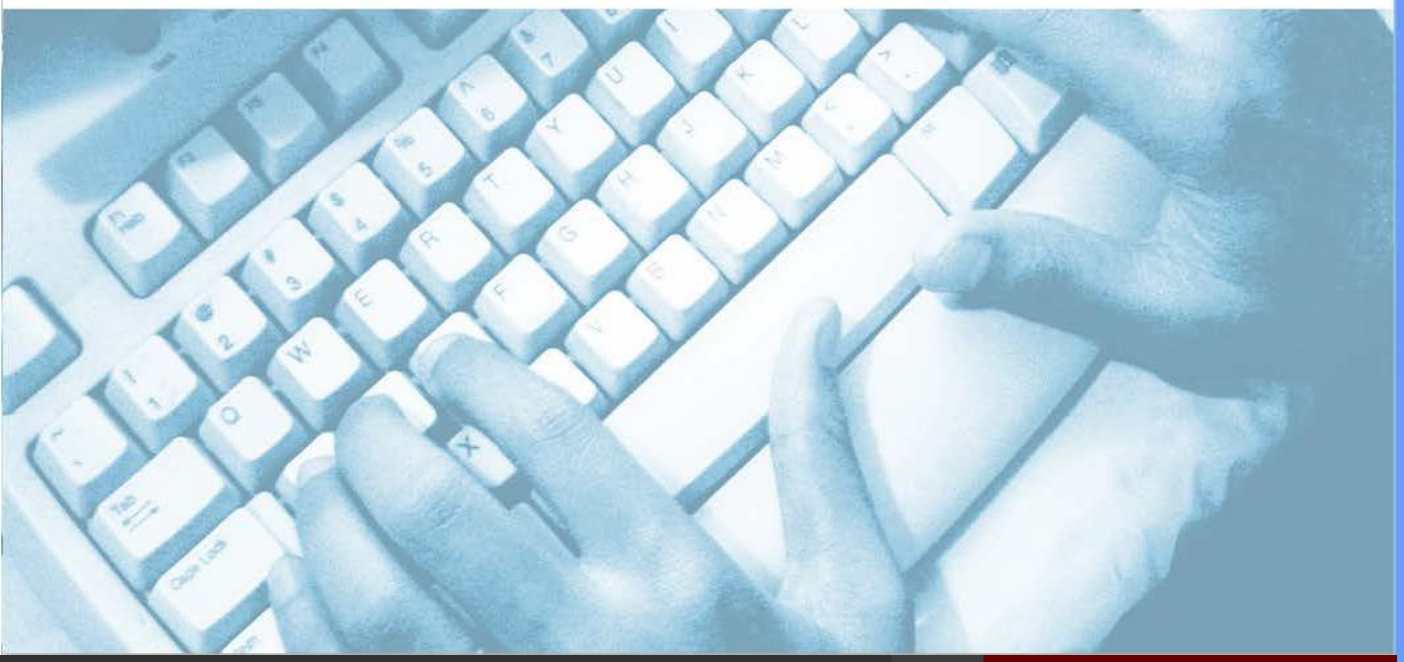
- PRESSMAN, R. (2002). *Ingeniería de Software. Un Enfoque Práctico. Quinta Edición* (QUITA EDICIÓN ed.). MEXICO: MC GRAW HILL.

- Pressman, R. (2005). *Ingeniería del Software.*
- Pressman, R. S. (2010). *Ingeniería del software un Enfoque Practico* (7ma. ed.). Mexico: Mc Graw Hill.
- UPEA. (19 de octubre de 2012). *Universidad Pública de El Alto*. Recuperado el 3 de diciembre de 2012, de http://www.upea.bo/ http://www.inegi.org.mx/inegi/contenidos/espanol/prensa/Contenidos/Articulos/tecnologia/punt osxfuncion.pdf

SISTEMA DE INFORMACION WEB, PARA EL SEGURO SOCIAL UNIVERSITARIO.
Manual de usuario

Versión 1.00

Unidad de seguro social (UPEA)



ÍNDICE

1. INTRODUCCIÓN

EL SISTEMA DE INFORMACION WEB, PARA EL SEGURO SOCIAL UNIVERSITARIO, está pensada y diseñada para facilitar a los estudiantes de las diferentes carreras de nuestra Universidad, para el registro al seguro social y gozar de los beneficios, que realiza la unidad de seguro social.

2. ENTRADA AL SISTEMA

Para acceder al sistema, el usuario debe de hacer uso de sus credenciales de acceso (usuario y contraseña).

- Usuario: Número de carnet.
- Contraseña: fecha de nacimiento (01-10-2000) día-mes-año

IMPORTANTE:

Solo podrán acceder los estudiantes matriculados de la gestión actual en el sistema mae.

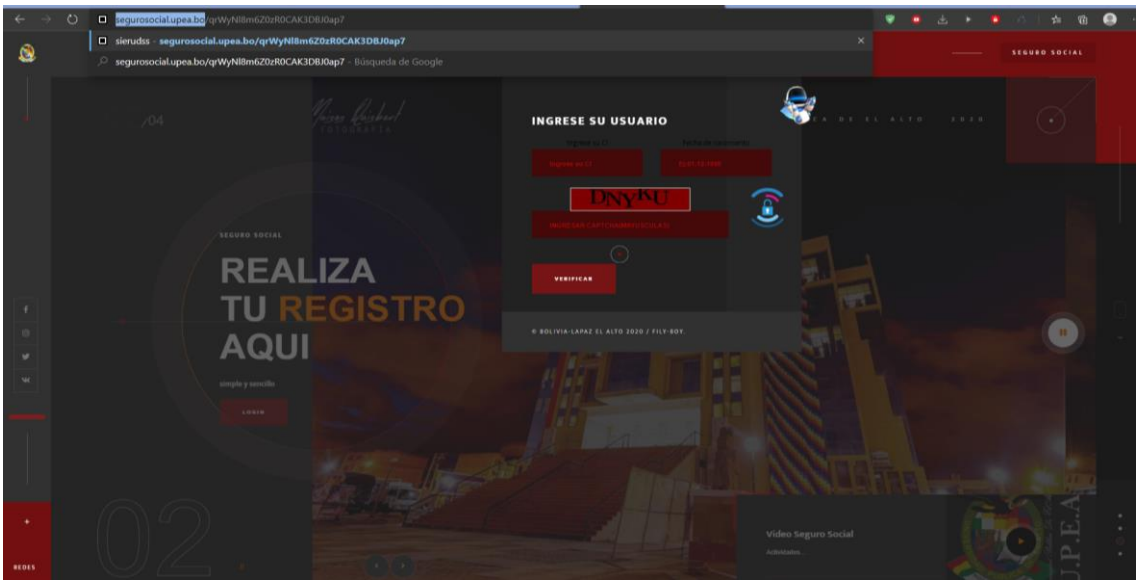
En caso el usuario y la entidad, no figuren como dados de alta en nuestra base de datos, podrán solicitar dichas altas desde las oficinas de sie.

2.1. ventana de entrada

Para acceder al sistema se utilizará el siguiente enlace:

http://segurosocial.upea.bo

Al dar clic en la url podrá ver el inicio de entrada al sistema (login), desde donde puede acceder al Registro, con sus credenciales.

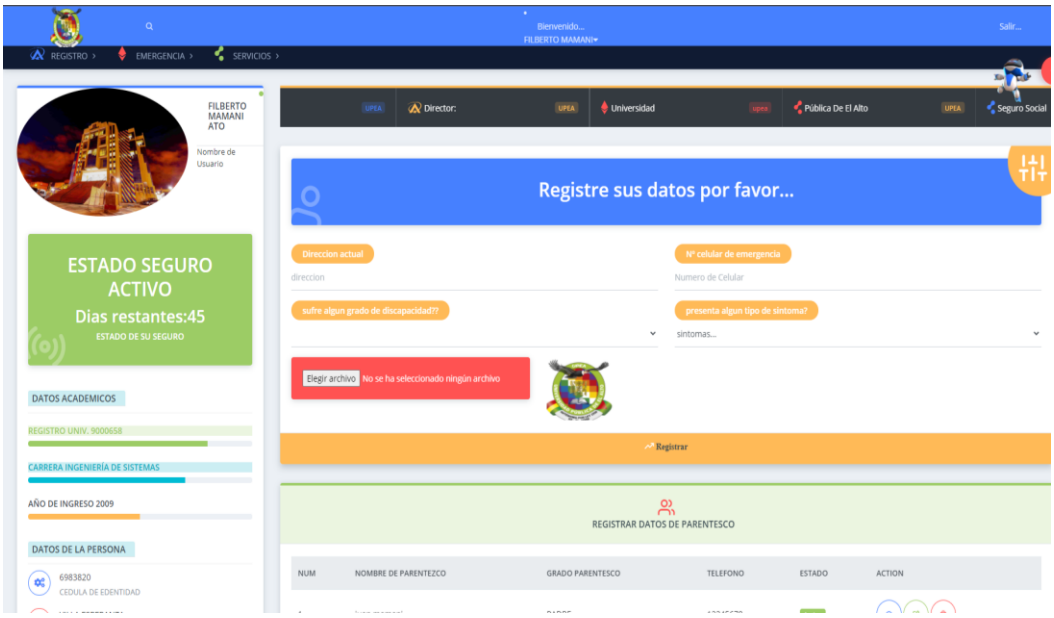


Botones disponibles:

ACCEDER: Una vez introducido el usuario y la contraseña, pulsar este botón para verificar.

2.1.1. Registro de datos

Una vez realizado el logeo el podrá registrar sus datos actuales.

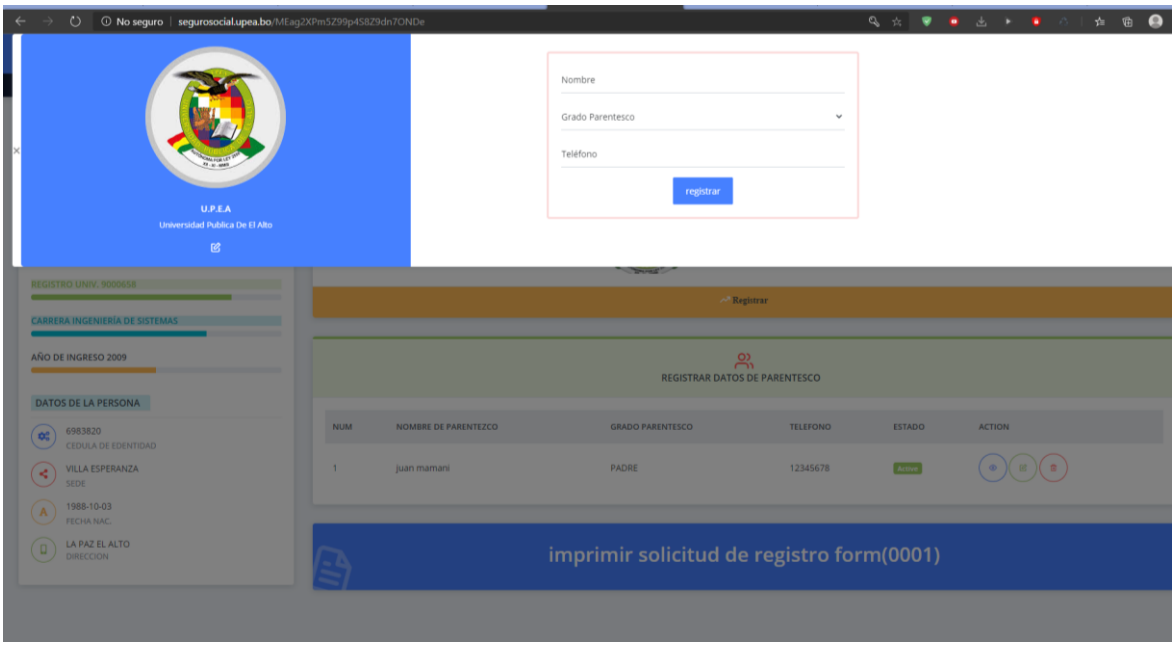


Apartados

Datos usuarios: Cumplimentar los datos relativos al usuario indicando al menos los campos indicados**.**

IMPORTANTE: El usuario no pude acceder a ningún servicio mientras no este registrado en sistema de seguro.

Registro de parentesco: En este apartado deberá llenar los datos de sus familiares.



Botones disponibles

Imprimir solicitud de registro: imprime el carnet de seguro social.

2.2 Reporte de incidente

En este menú de Emergencia puede realizar un reporte de incidente.

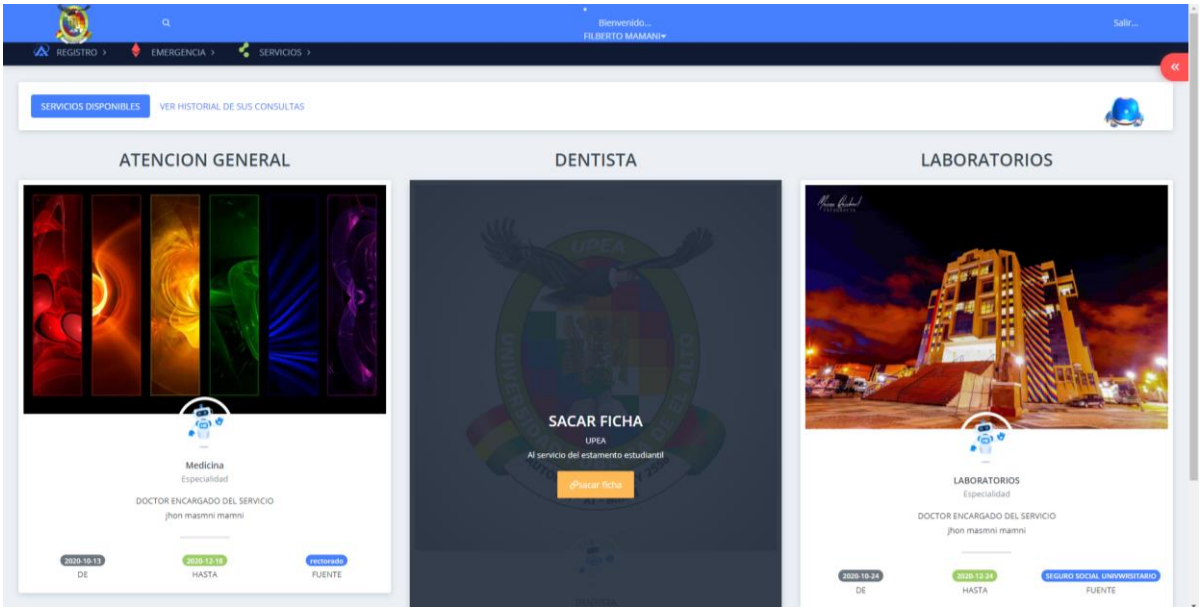
IMPORTANTE: llenar todos campos de manera responsable.

2.2.1 valoración de los mensajes

Una vez enviado el mensaje el usuario podrá a ver la valoración de la atención.

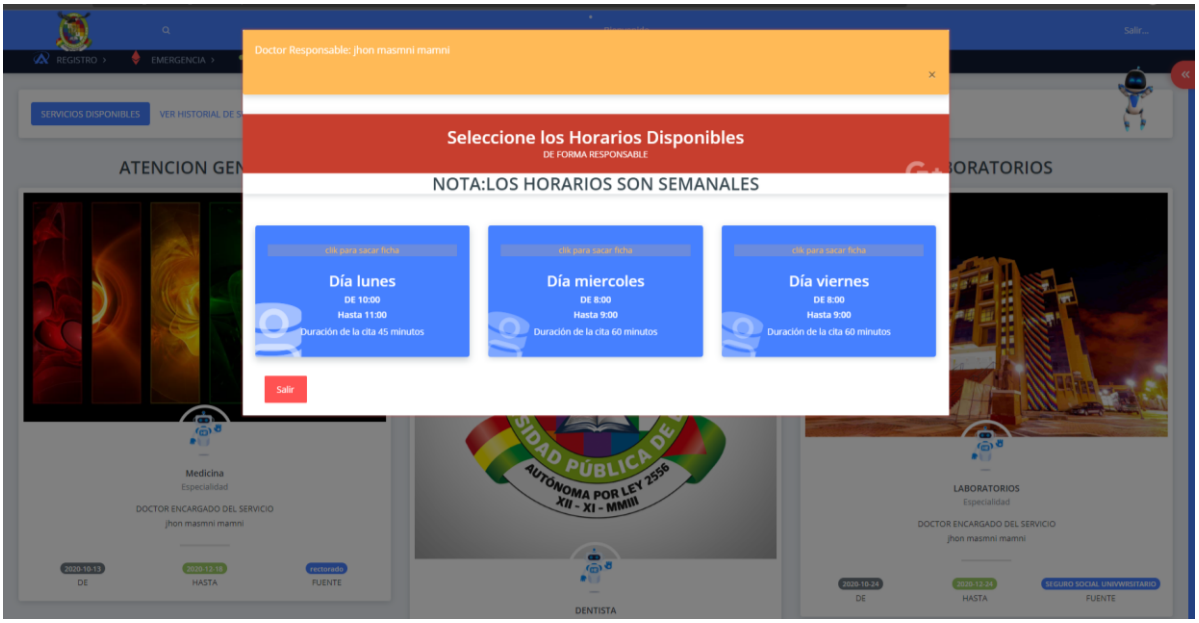
2.3 servicios

En menú SERVICIOS, se podrán ver los servicios disponibles lanzados por la unidad de seguro, donde se muestra la fuente, doctor encargado y tiempo de vigencia del servicio.



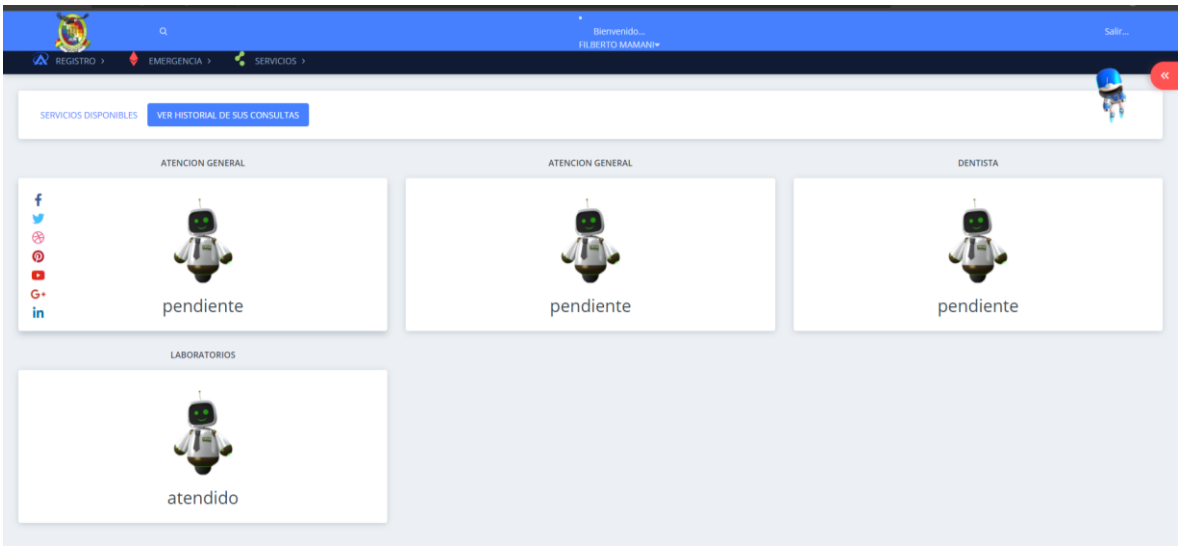
2.3.1 horarios disponibles de los servicios

en cada uno de los servicios el usuario podrá sacar **un horario disponible.**

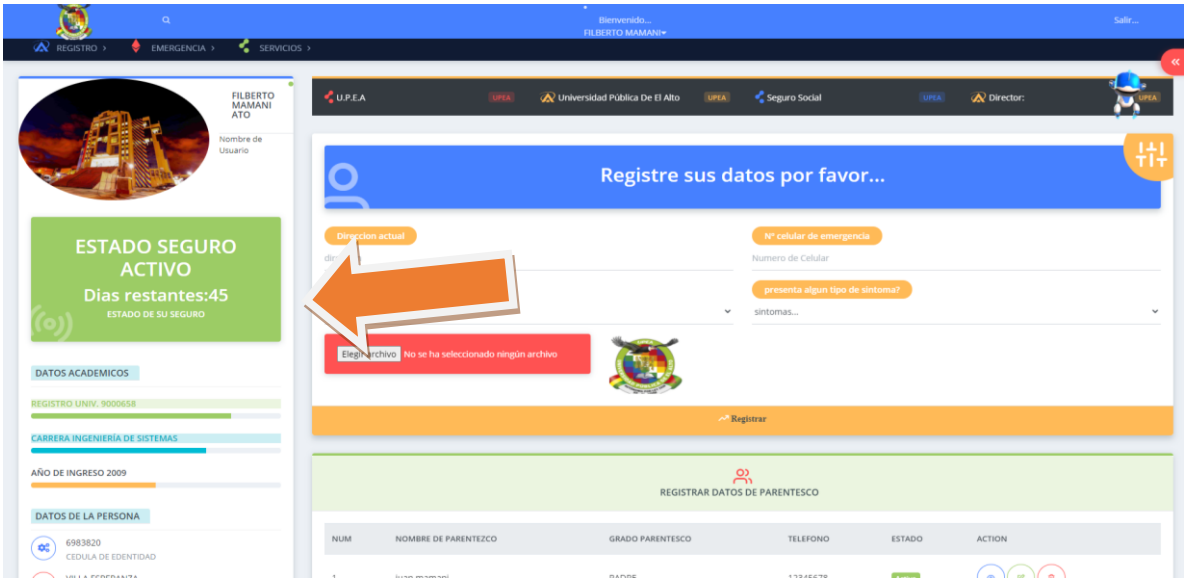


2.3.2 historiales

En el botón VER HISTORIAL DE SUS CONSULTAS.

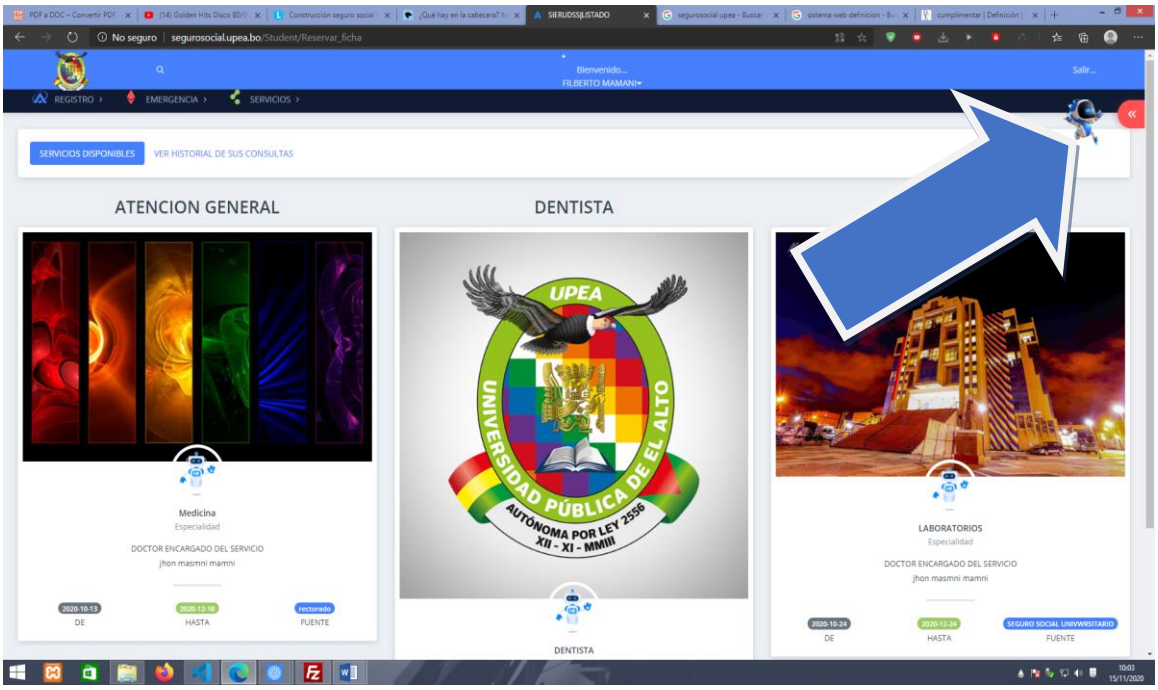


2.4 estado del seguro



En el estado seguro se muestra los días disponibles del seguro, una vez que se termine los días se inactiva el seguro de forma automática y el usuario ya no podrá acceder a ningún servicio

2.5 Cerrar Sesión



UNIVERSIDAD PÚBLICA DE EL ALTO

INGENIERIA DE SISTEMAS

MANUAL TÉCNICO

SISTEMA DE INFORMACION WEB, PARA EL SEGURO SOCIAL UNIVERSITARIO.
CASO: UPEA

Autor: Filberto Mamani Ato

2020

Contenido

Objetivos General

- Detallar la información necesaria para levantar el sistema.

Objetivos Específicos

- Definir claramente el proceso de instalación del sistema.
- Detallar los requerimientos mínimos de hardware y software para el funcionamiento del sistema.
- Describir las herramientas utilizadas en el desarrollo del sistema.

Introducción

En este manual se describa los pasos necesarios para poner en funcionamiento el sistema, se requiere que la persona encargada de la instalación, tenga conocimientos básicos de sistemas.

Requerimientos Técnicos

Requerimientos mínimos de hardware

- **Procesador:** Core
- **Memoria RAM:** 1 Gb
- **Disco Duro:** 250 Gb

Requerimientos mínimos de software

- Privilegios de Administrador
- **Sistema Operativo:** Windows 7 / Windows 8 / Windows 10 /Linux
- **Conexión a internet.**

Herramientas utilizadas para el desarrollo

APACHE 2.4

Servidor web HTTP[1] de código abierto para la creación de páginas y servicios web. Es un servidor multiplataforma, gratuito, muy robusto.

PHP 7

Es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. PHP[2] está enfocado principalmente a la programación de scripts del lado del servidor, por lo que se puede hacer cualquier cosa que pueda hacer otro programa CGI, como recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir cookies.

PHP puede emplearse en todos los sistemas operativos principales, incluyendo

Linux, Microsoft Windows, Mac OS X, RISC OS y probablemente otros más. PHP admite la mayoría de servidores web de hoy en día, incluyendo Apache, IIS, y muchos otros.

MYSQL 10.1

Es un sistema de administración de bases de datos relacionales rápido, sólido y flexible. Es ideal para crear bases de datos con acceso desde páginas web dinámicas, para la creación de sistemas de transacciones on-line, o para cualquier otra aplicación que implique almacenar datos, teniendo la posibilidad de realizar múltiples y rápidas consulta

Hosting:

El sistema se encuentra alojado en los servidores de nuestra universidad.

Dominio:

la dirección necesaria para que funcionen el sistema web es **upea.bo**

Enlace del sistema

En navegador Chrome, Edge, Firefox y otros ir al siguiente enlace
http://segurosocial.upea.bo/

Roles para ingreso al sistema

El aplicativo está definido por tres roles que son:

Administrador, Doctor, Estudiante

Administrador: tiene acceso a todos los módulos del sistema.

Buscar registro, Revisar, Mensajes, Especialidades, Servicios, Reportes, Control de usuarios, Registro Doctor

Doctor: tiene acceso a los siguientes módulos:

Atención de pacientes pendientes, Buscar Estudiante, Reportes

Estudiante: tiene acceso a los siguientes módulos:

Registro de datos, Reportar incidente, Servicios.