Universidad de las Ciencias Informáticas

Facultad 6

Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Título: "Sistema gestión de información para la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas"

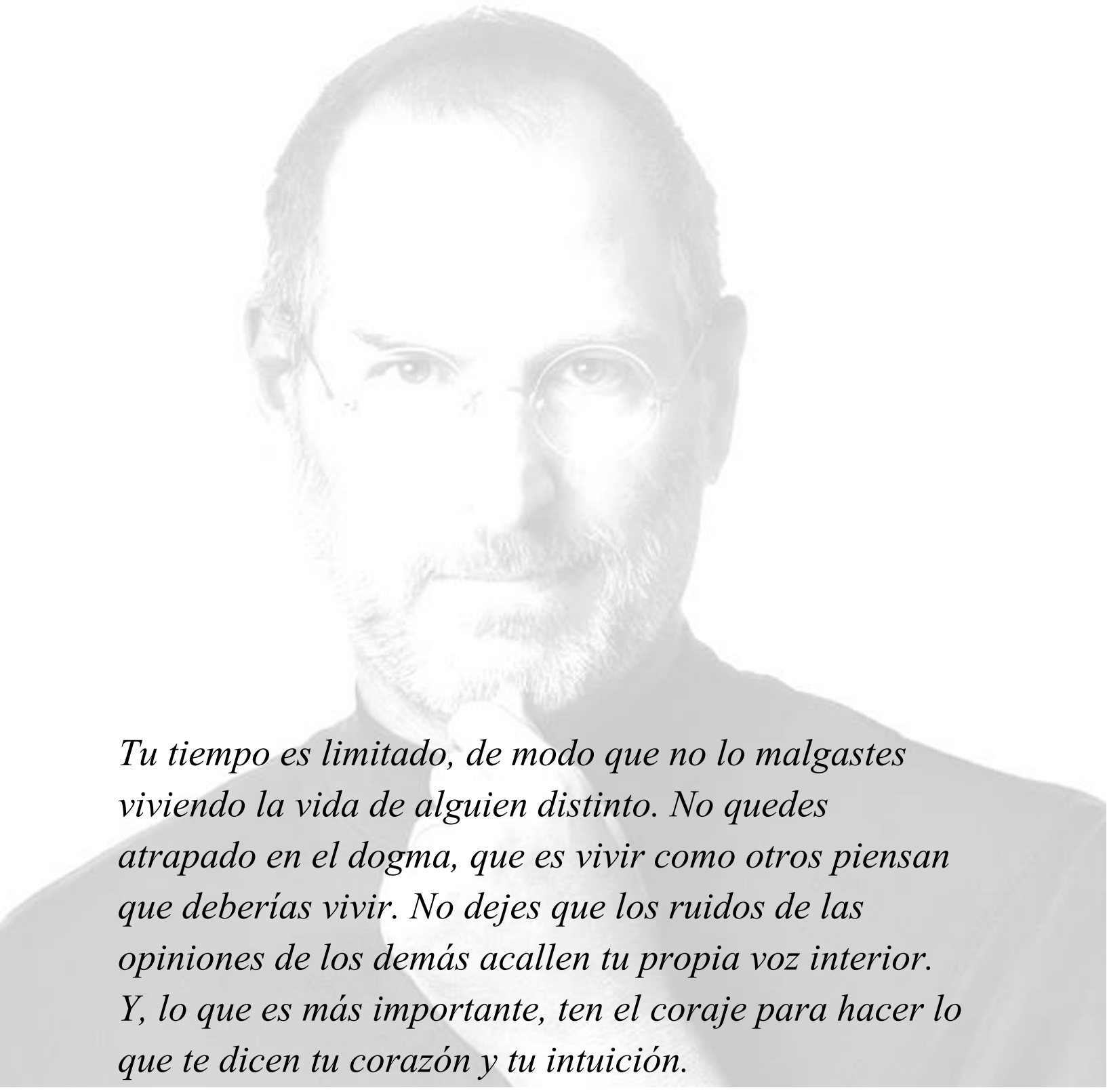
Autor:

Edel Eugenio Cañizares Miranda

Tutor:

Msc. Yuniel Eliades Proenza Arias

La Habana, julio del 2016
"Año 58 de la Revolución



Tu tiempo es limitado, de modo que no lo malgastes viviendo la vida de alguien distinto. No quedes atrapado en el dogma, que es vivir como otros piensan que deberías vivir. No dejes que los ruidos de las opiniones de los demás acallen tu propia voz interior. Y, lo que es más importante, ten el coraje para hacer lo que te dicen tu corazón y tu intuición.

Steve Jobs

Declaración de autoría

Declaro ser el único autor del trabajo titulado: “Sistema gestión de información para la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas” y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____

Edel Eugenio Cañizares Miranda

Autor

_____

MsC. Yuniel Eliades Proenza Arias

Tutor

Autor

Nombres: Edel Eugenio

Apellidos: Cañizares Miranda

Centro de estudios: Universidad de las Ciencias Informáticas

Tutor

Nombres: Yuniel Eliades

Apellidos: Proenza Arias

Centro de trabajo: Universidad de las Ciencias Informáticas

Correo electrónico: yproenza@uci.cu

Síntesis del Tutor: Graduado de Ingeniería Informática en el año 2006 de la Universidad de Holguín y CUJAE. Máster en Ingeniería de Software e Inteligencia Artificial por la Universidad de Málaga en el 2011. Se ha desempeñado como analista y desarrollador de aplicaciones Web y Desktop. Tiene experiencia en el trabajo con Sistemas de Información Geográfica y el desarrollo de Ontologías.

Dedico este Trabajo de Diploma a mis padres y a mis hermanos, por acompañarme en todo momento y brindarme todo el amor y dedicación que he necesitado.

Mis más sinceros agradecimientos a La Revolución Cubana y a la Universidad de las Ciencias Informáticas por darme la oportunidad de formarme como joven profesional.

Agradezco a mis padres por apoyarme en todo momento y porque además hoy se hacen realidad sus sueños también.

A mis hermanos, Enso y Osmani por ayudarme a llegar hasta aquí y servir siempre de ejemplo como hermanos mayores.

A toda mi familia en general por brindarme lo mejor de ellos.

A mi tutor Yuniel, quien me supo guiar de la mejor manera en la realización de este trabajo.

Al tribunal y mi oponente por sus consejos y recomendaciones en aras de lograr un trabajo con la calidad requerida.

A mis amistades de la Universidad, principalmente: Cesar, Alejandro, Raúl, Rafa. Carlos y el Cuso por ser mis hermanos incondicionales. En fin a todas aquellas personas que de una forma u otra, hicieron posible la realización de este trabajo: Muchas Gracias.

Resumen

Los Sistemas de información constituyen una poderosa herramienta en el funcionamiento de las entidades y organismos a nivel mundial, se han convertido en uno de los principales pilares en la gestión de una empresa contribuyendo a lograr un eficiente funcionamiento. Se desarrollan con el objetivo de ayudar al proceso de ayuda a la toma de decisiones y proporcionan altos niveles de confianza. El presente trabajo contiene la investigación y desarrollo de un sistema gestión de información para la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas (UCI), dirigido al control y procesamiento de información referente a los diversos planteamientos realizados por la comunidad universitaria. La aplicación desarrollada sobre el framework symfony versión 2.8 permite llevar a cabo la gestión de los planteamientos, la cual comprende su recepción, almacenamiento, procesamiento y respuesta a la persona que realiza el planteamiento, así como también la creación de los distintos reportes semestrales. Para el desarrollo de la investigación se empleó la metodología de desarrollo AUP, posibilitando así la documentación de cada una de las etapas de vida del desarrollo del software. Además se definieron las tecnologías y herramientas adecuadas para concretar la implementación de la aplicación propuesta. Una vez concluido el desarrollo del sistema se aplicaron pruebas necesarias para verificar su correcto funcionamiento. Teniendo como resultado final una aplicación que es capaz de gestionar de forma eficiente los planteamientos en la Oficina de Atención a la Población de la UCI.

Palabras claves: gestión, herramientas, información, sistemas, tecnologías.

Abstract

Information systems are a powerful tool in the functioning of institutions and organizations worldwide, they have become one of the main pillars in the management of a company helping to achieve efficient operation. They are developed with the aim to help the decision making help process and provide high levels of confidence. This paper contains research and development of a information management system for the Office of Attention to the Population of the University of Informatics Sciences (UCI), directed to the control and processing of information on the various proposals made by the university community. The application developed on Symfony framework version 2.8 allows you to perform management approach, which includes receipt, storage, processing and responding to the person making the approach, as well as the creation of various semiannual reports. For the development of research AUP development methodology was used, thus enabling the documentation for each stage of software development life. In addition appropriate technologies and tools to realize the implementation of the proposed implementation were defined. Once the completed system development tests necessary to verify proper operation were applied. With the final result an application that is able to efficiently manage the proposals in the Office of Attention to the Population of the UCI.

Key words: management, tools, information, systems, technologies.

Tabla de contenidos

Índice de tablas

Índice de figuras

Introducción

En el mundo de hoy la información ha llegado a convertirse en un recurso estratégico imprescindible para determinar la competitividad y sustento a los procesos de toma de decisiones, siendo esta el único elemento capaz de crear conocimiento y satisfacer las necesidades de las personas y los organismos institucionales, constituyendo así un pilar indispensable para el desarrollo y control de las entidades.

Por su parte las Tecnologías de la Información y la Comunicación (TIC) han permitido evolucionar el mundo empresarial a un nivel superior, al darle un nuevo enfoque al uso estratégico de la información. Las TIC no solo han influido positivamente en el desarrollo empresarial sino también en todas esferas de la sociedad actual, la cual demanda productos y servicios informáticos de gran calidad, que den respuesta a sus necesidades en un mercado cada día más globalizado y competitivo.

A su vez el proceso de toma de decisiones en las empresas debe sustentarse en la correcta utilización de las TIC, proporcionando un marco de soluciones amplio, que incluye el almacenamiento, la recuperación, el procesamiento y la transmisión de datos de un sitio a otro y la información necesaria para generar resultados y elaborar informes de alta calidad. La información y el tiempo son elementos importantes para plantear directrices organizacionales, las cuales son necesarias para lograr una mayor eficiencia y disminuir el consumo de tiempo de procesamiento de la información y, de esta forma, tomar decisiones oportunas y coherentes.

En Cuba el desarrollo de la informática y las comunicaciones constituye un proceso de mucha importancia y para el cual se destinan cuantiosos esfuerzos, con el objetivo de propiciar un mayor desarrollo en los sectores del país. La Universidad de las Ciencias informáticas (UCI) constituye un motor impulsor en este ámbito situándose a la vanguardia en el proceso de informatización. Como es de esperar en la UCI se pone de manifiesto dichos adelantos presentando un porciento elevado de áreas informatizadas.

El rectorado de la UCI está compuesto por múltiples departamentos en los cuales se desarrollan numerosas tareas de gestión y control de información. Entre ellos está la Oficina de Atención a la Población constituye un departamento del rectorado el cual se encarga de atender los diferentes planteamientos de la población y emitir la respuesta correspondiente a cada uno de ellos. Esta tarea está regida por la INSTRUCCIÓN No. 03/13 del Ministerio de la Educación Superior (MES) la cual, en lo referente a la gestión de los planteamientos, establece que:

"Es política del Gobierno y del Ministerio de Educación Superior atender con prioridad aquellas cuestiones que se relacionan con la población. Especial interés se ha de prestar a verificar y dar una respuesta oportuna y adecuada a los planteamientos provenientes de la población, lo que constituye una responsabilidad indelegable de los principales jefes de las entidades, para que esta actividad se atienda adecuadamente y se someta a control y evaluación periódica de los órganos colectivos de dirección".

En la actualidad, este control se efectúa de forma manual y solo se cuenta con el apoyo de herramientas como procesadores de textos y hojas de cálculo para llevar a cabo toda la gestión de los planteamientos, la cual comprende su recepción, almacenamiento, procesamiento y respuesta a la persona que lo realiza, así como también la creación de los distintos reportes semestrales. Esto trae consigo que el trabajo no sea realizado de forma óptima y se reduzcan notablemente las posibilidades que brinda el proceso de ayuda a la toma de decisiones. Es por ello que el personal encargado de realizar esta tarea experimenta algunas limitaciones al realizar el proceso de almacenamiento, transformación y control de la información recibida, lo que incide negativamente en el tiempo de respuesta a los afectados y propicia que el margen de error al emitir los reportes periódicos sea mayor, por esta razón la calidad del trabajo se ve seriamente afectada.

Teniendo en cuenta lo expuesto anteriormente se identifica como **problema de la investigación:** ¿Cómo facilitar la gestión de la información de los planteamientos en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas para disminuir el tiempo en su procesamiento y emisión de los reportes asociados?

Para llevar a cabo la investigación se ha definido como **objeto de estudio** los sistemas de gestión de información, enmarcado en el **campo de acción** la informatización de los procesos de gestión de información en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas.

Para darle solución al problema se plantea el siguiente **objetivo general:** Desarrollar un sistema para la gestión de información de los planteamientos en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas.

Para el desarrollo de la investigación se tienen en cuenta las siguientes **preguntas científicas:**

1. ¿Qué factores influyen en la demora, ejecución o falta de precisión de los reportes que emite la Oficina de Atención a la Población de la UCI, asociados a los planteamientos recibidos?

2. ¿Qué tareas o subprocesos asociados a la gestión de la información de dichos planteamientos pudieran automatizarse, de manera que se mejore la toma de decisiones?
3. ¿Cómo influiría el empleo de un sistema informático en la disminución del tiempo empleado en el procesamiento de la información y elaboración de los reportes?

Para lograr el cumplimiento del objetivo general se plantearon las siguientes **tareas de la investigación:**

1. Definición de los principales conceptos, características y tendencias actuales relacionadas con la gestión de la información y los sistemas de información.
2. Análisis del estado actual del negocio en la Oficina de Atención a la Población, sus características y necesidades.
3. Caracterización de las soluciones similares existentes, a nivel nacional e internacional.
4. Selección de la metodología de desarrollo, las herramientas y tecnologías a utilizar durante el desarrollo del sistema para la gestión de planteamientos.
5. Definición de las características y el diseño del sistema a implementar.
6. Implementación del sistema para la gestión de información de los planteamientos en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas.
7. Aplicación de las pruebas de funcionamiento (Caja Negra) y prueba de validación del sistema desarrollado.

Para el desarrollo de la presente investigación se tuvieron en cuenta los siguientes métodos de investigación científica:

Métodos Teóricos:

- **Analítico-sintético**: permite la extracción de los elementos más importantes del estudio y análisis de diversas fuentes bibliográficas que se relacionan con el desarrollo de sistemas de gestión de información, de forma tal que permita aplicarlos en el posterior desarrollo del mismo. Su uso permitió haber un análisis claro de la problemática asociada a la gestión de los planteamientos y quejas en la Oficina de Atención a la Población de la UCI, además de poder identificar los procesos a informatizar.
- **Histórico-lógico**: permite valorar en el transcurso de la historia, los sistemas y tecnologías que han sido empleados en la solución del objeto de estudio de la investigación y así definir sus ventajas y desventajas para apoyar a la selección de los

elementos necesarios que pueden servir de base para el desarrollo del sistema gestión de información a desarrollar. La aplicación de este método permitió realizar un estudio de las características, tendencias en tecnologías y tipos de aplicaciones que automatizan procesos de gestión de información, tanto a nivel nacional como internacional.

Para la recopilación de información se tuvo en cuenta las técnicas Entrevista y Tormenta de ideas las cuales se exponen a continuación:

Entrevista: Esta técnica consiste en realizar una conversación entre varias personas, al menos dos, en la cual una realiza la función de entrevistador y otra u otras fungen como entrevistados; estas personas dialogan con respecto a ciertos esquemas o pautas acerca de un problema o cuestión determinada de importancia para el entrevistador, teniendo un propósito profesional (Pérez, 2005). El uso de esta técnica permitió entender primeramente como se realiza la gestión de los planteamientos en la Oficina de Atención a la Población y en un aspecto más específico permitió conocer cuáles son los procedimientos que se llevan a cabo, así como la información que se maneja de cada planteamiento realizado.

Tormenta de ideas: Es una de las más importantes del repertorio de técnicas conocidas para el grupo de trabajo y es la técnica básica del pensamiento creativo. La sencillez con la que se puede aplicar, la facilidad con que modifica la forma habitual de pensar en grupo (debate clásico). Su principal objetivo es la generación de ideas en un ambiente libre de críticas principalmente al comienzo del proceso de captura de requerimientos (Cembranos, y otros, 2003). El debate se estableció con la persona encargada de llevar el control de los planteamientos en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas; donde se permitió definir el problema de investigación, obtener los requerimientos del sistema a desarrollar y finalmente determinar y analizar las consecuencias generadas a partir de la situación problemática existente.

Estructura del Documento

El presente Trabajo de Diploma está estructurado de la siguiente manera: introducción, tres capítulos, conclusiones, recomendaciones y referencias bibliográficas.

Capítulo 1: Fundamentos teóricos.

Este capítulo aborda definiciones y conceptos importantes respecto a los sistemas de gestión de información. Se realiza un estudio de sistemas existentes en el mundo y en Cuba orientados a la gestión de la información. Se fundamenta además mediante una detallada descripción la metodología, herramientas y tecnologías utilizadas para dar solución al problema planteado.

Capítulo 2: Características y diseño del sistema.

En este capítulo se profundiza sobre las funcionalidades que debe tener el sistema. Se presentan los requisitos funcionales y no funcionales. Además se muestran los casos de usos del sistema y su descripción, así como los actores que interactúan con el sistema. Además describe el diseño de la solución propuesta, se brinda una descripción de los estilos y patrones arquitectónicos más utilizados y la aplicación de los mismos para la confección de los diagramas de clases y el modelo del diseño del sistema.

Capítulo 3: Implementación y pruebas del sistema.

Este capítulo relaciona los contenidos de los capítulos anteriores y da paso a la realización del modelo de implementación para desarrollar el sistema propuesto. Además recoge los modelos de prueba que verifican el correcto funcionamiento de los requisitos en la solución propuesta.

Capítulo 1: Fundamentos teóricos.

En el presente capítulo se presentan los conceptos importantes respecto a los sistemas de gestión de información. Se realiza un estudio de sistemas existentes en el mundo y en Cuba orientados a la gestión de la información. Se fundamenta además, mediante una detallada descripción, la metodología, herramientas y tecnologías utilizadas en el desarrollo de la propuesta de solución al problema planteado.

1.1 Principales Conceptos asociados al tema de investigación

Gestión de información (GI)

Se define gestión de información como el conjunto de actividades realizadas con la finalidad de controlar, almacenar y, posteriormente, recuperar de forma eficiente la información producida, recibida o captada por cualquier entidad en el perfeccionamiento de sus tareas (Ponjuán Dante, y otros, 2008). La información para que pueda utilizarse y genere ventajas competitivas de tener tres características básicas: completa, confiable y oportuna. (Pérez Rodríguez, y otros, 2005) Además, debe emplearse para establecer relaciones con clientes, colaboradores, distribuidores; realizar procesos en la organización, crear productos y servicios con un alto grado de valor que le proporcionen a la organización una ventaja competitiva (Soto, 2010).
Las nuevas estrategias para la gestión de la información deben estar en consecuencias con las nuevas tendencias organizacionales.

Según (Pérez Rodríguez, y otros, 2005) se pueden determinar las siguientes funciones:

- Delimitar las principales necesidades tanto internas con externas de información, específicamente todo el contenido relacionado con los procesos administrativos de la organización.
- Perfeccionar el flujo organizacional de la información y el nivel de comunicación.
- Manejar eficientemente los recursos organizacionales de información, mejorar las inversiones sucesivas en los mismos y mejorar su aprovechamiento.
- Entrenar a los miembros de la organización en el manejo o la utilización de los recursos informacionales.
- Contribuir a modernizar o perfeccionar las actividades organizativas y sus procesos administrativos.
- Determinar las necesidades de información externa de la organización y satisfacerlas (Pérez Rodríguez, y otros, 2005).

Sistema de información (SI)

Con el fin de gestionar de manera eficiente la información que se genera en todas las esferas de la sociedad, se crearon los SI para agilizar todos los procesos relacionados con el procesamiento de información. Los SI son la integración de diferentes elementos interrelacionados entre sí y que realizan actividades como la recepción o entrada al sistema, procesamiento, almacenamiento y distribución de información y proporcionan además un mecanismo de retroalimentación para cumplir con sus funciones. Además estos tienen como objetivo principal apoyar las actividades de una entidad siempre y cuando estén disponibles los activos necesarios y el recurso humano indispensable para operar el sistema. Los SI realizan cuatro actividades básicas con la información dentro del funcionamiento estándar: entrada, almacenamiento, procesamiento y salida de la información (Stair, y otros, 2010).

A continuación se describen cada una de estas actividades:

Entrada de información: es el proceso mediante el cual se introducen los datos necesarios para procesar la información. Las entradas pueden ser manuales o automáticas en dependencia de la fuente desde la que se proporcionan los datos. Una entrada manual es aquella que se suministra en forma directa por el usuario mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos (Abreu , y otros, 2009).

Procesamiento de información: es la capacidad del SI para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados. Esta característica de los sistemas permite la transformación de datos fuente en información que puede ser utilizada para la toma de decisiones, lo que hace posible, entre otras cosas, que un tomador de decisiones genere una proyección financiera a partir de los datos que contiene un estado de resultados o un balance general de un año base (Letechi, y otros, 2006).

Almacenamiento de información: el almacenamiento es una de las actividades o capacidades más importantes que tiene un SI, ya que a través de esta propiedad el sistema puede acceder la información guardada en la sección o proceso anterior. Esta información suele ser almacenada en estructuras de información dependiendo del tipo de SI utilizado. La unidad de almacenamiento de información está descrita en dependencia del formato de esta, pueden clasificarse como formatos duros o digitales según el SI utilizado, algunos ejemplos son: discos magnéticos o discos duros, Informes, discos flexibles o diskettes, libros, documentos discos compactos (CD-ROM), entre otros (Castellanos Guevara, y otros, 2012).

Salida de información: es la capacidad de sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas de salida son las impresoras, terminales, diskettes, cintas magnéticas, la voz, los graficadores, documentos, los plotters, entre otros (Bermudez, y otros, 2010).

Es importante aclarar que la salida puede constituir la entrada a otro SI.

Los SI constituyen hoy, no sólo soportes de los negocios, sino, además, un instrumento de ventajas competitivas sostenibles al permitir gestionar los activos tangibles e intangibles y convertirse en una herramienta integral de gerencia (Lara, 2006).

Principales objetivos de los SI.

- Proporcionar información para el soporte a la toma de decisiones.
- Procesar datos obtenidos para convertirlos en información necesaria
- Facilitar la información oportuna, conservación y el acrecentamiento que proporciona flexibilidad en la operación de una organización.
- Apoyar a la organización para desarrollar estrategias apropiadas para el negocio establecidas en un entorno competitivo.

Sistema gestión de Información (SGI)

En la actualidad se ha hecho indispensable cada vez mayor la implementación de los SI para apoyar el proceso de toma de decisiones, se ha visto como el mundo empresarial ha implementado en mayor medida los SI como componentes informáticos provocando revolucionar el concepto de GI, constituyendo de esta forma un paso de avance en el mejoramiento de los procesos asociados a la gestión de información al implementar soluciones con un alto grado de confiabilidad y eficiencia.

1.2 Proceso de gestión de planteamientos en la Oficina de Atención a la Población de la UCI

El proceso de gestión de los planteamientos comienza cuando la Oficina de Atención a la Población recibe, por diferentes vías, los planteamientos realizados por la población. Estos planteamientos son recibidos por correo electrónico, informes realizados de las personas interesadas en promover su inquietud, entrevista con la persona encargada de gestionar los planteamientos o por notas anónimas que puedan ser enviadas a esta oficina. A partir de la recepción de un planteamiento, la persona encargada de su gestión procede a clasificar el planteamiento según su contenido, además se tienen en cuenta otros aspectos importantes como el área implicada, tiempo disponible para dar respuesta al planteamiento, cargo de la persona que lo realizó, fecha de recepción, tipo de caso que comprende y una breve

descripción de su contenido. Una vez creado el planteamiento se procede a realizar un informe del mismo donde quedan plasmadas las principales causas que lo originaron, en este informe se especifica además, si el nivel de importancia del contenido del planteamiento requiere la creación de una comisión disciplinaria o realizar alguna auditoría para darle solución. Una vez concluido este proceso se emite una respuesta a la persona que realizó el planteamiento y se archiva además si esta persona quedó satisfecha o no con la respuesta emitida. Posteriormente se realiza una evaluación final del planteamiento y se adjunta al contenido del mismo.

Estos planteamientos contienen la información necesaria para realizar reportes que necesita el MES para llevar un control de los planteamientos realizados por la comunidad universitaria y de esta forma tomar las medidas pertinentes con el fin de solucionar la mayor cantidad de inconformidades o inquietudes detectadas.

Como dato de interés, se observa que estos reportes se realizan de forma manual, lo que implica el empleo de tiempo en revisar los planteamientos realizados en un período de tiempo especificado por el MES.

1.3 Soluciones existentes en el mundo

La gestión de información, unido al progresivo desarrollo tecnológico, condiciona que muchas instituciones y entidades en el aspecto informativo, presenten una excesiva centralización de la información y el flujo abundante de documentos impresos; y sucede además que quienes necesitan la información no disponen de ella en el momento y espacio adecuados. Los directivos con frecuencia se encuentran abrumados por documentos e informaciones innecesarias; en muchos casos, se dispone de software y plataformas incompatibles entre ellas en una misma institución y se desaprovechan los espacios y recursos tecnológicos. Es por ello que surgen los sistemas que gestionan la información (SGI) provocando un avance cuantitativo en la gestión del conocimiento empresarial y despertando la admiración y reconocimiento de todos en cuanto a su importancia para la ayuda a la toma de decisiones. Un estudio sobre diferentes SIG existentes en el mundo detalló algunas soluciones como IsoCloud, Liberum Help Desk y CEDRUX, los cuales a continuación se describen puntualizando los aspectos más importantes de cada una de ellas.

IsoCloud

IsoCloud es una aplicación que permite adaptarse a las empresas de una forma rápida y a través de la web. Contiene tres servicios fundamentales: Sistema de Gestión Integrado (14 módulos), Sistema de Gestión de Calidad (10 módulos) y Sistema de Gestión Medio Ambiental (13 módulos). Algunas de sus características fundamentales son:

- Generación de informes de incidencias, objetivos y metas, y resultados de encuestas de satisfacción.
- Gestión documental y proceso de control de firmas.
- Gestión de diferentes tipos de incidencias (no conformidades internas, no conformidades de proveedor, acciones correctivas, entre otras.), relacionadas con distintas procedencias (encuestas de satisfacción, planes de formación y auditorías) (indracompany.com, 2012).

Esta herramienta resulta una novedosa aplicación para la gestión de información, pero no se ajusta a las especificaciones deseadas para resolver el problema en cuestión, esto se evidencia en las especificaciones del negocio detalladas en la descripción de la problemática de la investigación, una posible adopción implicaría realizar gastos económicos adquiriendo la licencia de este software la cual es privativa, además no bastaría ya que sería necesario realizar una reestructuración del sistema para adaptarlo a las funciones deseadas. Es válido destacar que varias de sus características sirvieron para guiar el proceso de desarrollo de la solución propuesta, estas delimitaron varias pautas a seguir mediante la generación de informes de incidencias ya que el sistema a implementar debe poder plasmar los resultados recogidos en el proceso de recopilación de información y de esta forma poder mejorar el proceso de toma de decisiones y por otro lado el procesamiento y gestión de incidencias dentro de las cuales se encuentran las no conformidades, permitieron tener una idea de cómo pudieran gestionarse los planteamientos que constituyen quejas o denuncias debido a que estos son catalogados como no conformidades de la población universitaria.

Liberum Help Desk

Liberum Help Desk es un software para la GI el cual se describe como una solución completa de ayuda para los sistemas las pequeñas y medianas empresas. Este software proporciona una sencilla interfaz fácil de usar y mediante la gestión web proporciona un seguimiento de los problemas de soporte técnico.

Liberum Help Desk es de código abierto bajo la licencia GPL y gratuita para su uso. El software está programado en HTML y ASP y se modifica y personaliza fácilmente, por lo que tiene un gran valor agregado por poseer características de adaptabilidad y flexibilidad al entorno en el que se utiliza. Todo lo que se requiere para ejecutar Liberum Help Desk es Sistema Operativo Windows Server (Liberum, 2010).

Liberum Help Desk es una aplicación que ofrece muchas ventajas de usabilidad y además es un software libre, la desventaja está en que necesita Windows para ser usado, aspecto este que resulta un impedimento atendiendo a las políticas de migración de software libre del país,

además de la desventaja de tener que hacer cambios en su estructura para adaptarlo a las especificaciones requeridas por lo que no constituye una solución factible para dar solución al problema planteado.

De este software se pudo apreciar como elementos de apoyo a la investigación, la interfaz iterativa y de fácil manipulación debido a que esta respeta los estándares modernos utilizados en la navegación en Internet y sirvió como guía para un posterior diseño de las interfaces de la solución propuesta.

CEDRUX

Cedrux es un sistema integrado de gestión que se desarrolló en la Universidad de las Ciencias Informáticas. Cuenta con un subsistema para el control de los recursos de cómputo que pretende servir de apoyo al proceso de toma de decisiones en las entidades. El subsistema es capaz de gestionar recursos de forma dinámica, tributa mediante operaciones al sistema contable de la empresa. Incluye además temas como la agrupación de recursos y la generación de documentos. (Blanco, 2009)

Este sistema permite la gestión de recursos pero los procesos que realiza no se ajustan a las especificaciones de la entidad por lo que no constituye una solución factible ya que su uso implicaría hacer adaptaciones en la distribución de los procesos que se llevan a cabo en la organización.

A partir del estudio realizado sobre las soluciones existentes, se concluyó que no son factibles para la gestión de información que se desea realizar, siendo la principal razón por la que no constituyen una solución factible, la de no ajustarse a las necesidades del cliente debido a que no están enfocados al negocio detallado que se enuncia en la descripción de la situación problemática planteada, por lo que su adopción constituiría un esfuerzo en redefinir las funcionalidades requeridas mucho mayor que desarrollar una solución desde su inicio. Además la mayoría de los sistemas analizados están implementadas sobre tecnologías privadas y su uso constituiría un gasto elevado por concepto de pago en licencias de software o no estarían acorde a los esfuerzos desarrollos por el país de migrar a tecnologías libres. Por otra parte la aplicación implementada sobre tecnología libre se descarta ya posee características que dificultan la posibilidad de adaptarla a los requerimientos deseados debido a que está implementada con el objetivo de satisfacer las necesidades del sistema contable de una entidad, por lo que sería necesario una reestructuración casi en su totalidad del software analizado.

1.4 Metodología, herramientas y tecnologías.

A continuación se presentan las principales herramientas, tecnologías y metodologías utilizadas para dar solución al problema planteado. Para su selección se tuvieron en cuenta las necesidades existentes, tendencias actuales y el entorno donde se desplegará la aplicación.

1.4.1 Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas, herramientas y soporte documental para el desarrollo de un producto (*software*). Se basan en una combinación de los modelos de proceso genéricos (cascada, evolutivo e incremental.). Son para estructurar, planear y controlar el proceso de desarrollo del *software*. Constituyen una guía que define las tareas y actividades que se deben realizar para obtener un *software* de buena calidad (Universidad Politécnica de Valencia, Departamento de Sistemas Informáticos y Computación, 2013).

Además una metodología de desarrollo de sistemas se refiere al marco que se utiliza para estructurar, planificar y controlar el proceso de desarrollo de un sistema de información. Una amplia variedad de tales marcos han evolucionado a lo largo de los años, cada uno con sus fortalezas reconocidas y propias debilidades (CMS, 2008).

AUP-UCI

Como metodología utilizada para guiar el proceso de desarrollo de software en la presente investigación se seleccionó el Proceso Unificado Ágil (AUP por sus siglas en inglés) variación UCI. Describe a su vez de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP.

AUP en su variación UCI propone un ciclo de vida que abarca siete flujos de trabajos, cuatro ingenieriles y tres de apoyo: Modelado, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyectos y Ambiente y adopta muchas de las técnicas ágiles de XP y otros procesos ágiles manteniendo la formalidad de RUP. De las cuatro fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola titulada Ejecución y se agrega la fase de Cierre.

Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En la fase de ejecución se realizan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la

arquitectura. En la fase de cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI (Sánchez, M, y otros, 2007).

Para el desarrollo de la aplicación se utilizó el segundo escenario definido debido a que es el que se ajusta con las especificaciones del negocio planteadas en la problemática planteada, dado que el objetivo primario de esta investigación es la gestión y presentación de información.

1.4.2 Lenguaje de modelado de sistemas

Un lenguaje de modelado es un lenguaje artificial diseñado para expresar modelos. Ya que los modelos habitualmente se muestran en forma de diagramas por comodidad, los lenguajes de modelado suelen incorporar notaciones gráficas. Igual que los lenguajes naturales, los lenguajes de modelado poseen un léxico, es decir, un conjunto de palabras que existen en el lenguaje; y una sintaxis, es decir, un conjunto de reglas que nos dice cómo se pueden combinar dichas palabras para componer "frases" que tengan sentido (ConML, 2012).

Lenguaje de modelado como soporte a la metodología (UML) 2.0

El Lenguaje de Modelado Unificado (UML), es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema(modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

En modelo UML consta de tres categorías principales de los elementos del modelo, cada uno de los cuales pueden ser utilizados para hacer declaraciones sobre los diferentes tipos de elementos individuales dentro del sistema que está siendo modelado. Estas categorías son:

Clasificadores. Un clasificador describe un conjunto de objetos. Un objeto es un individuo con un estado y las relaciones de otros objetos. El estado de un objeto identifica los valores para ese objeto de propiedades del clasificador del objeto.

Eventos. Un evento describe un conjunto de posibles ocurrencias. Una ocurrencia es algo que sucede que tiene alguna consecuencia en relación con el sistema.

Comportamientos. Un comportamiento describe un conjunto de posibles ejecuciones. Una ejecución es una representación de un conjunto de acciones (potencialmente sobre un cierto período de tiempo) que pueden generar y responder a las ocurrencias de eventos, acceder y cambiar el estado de los objetos (OMG (Object Management Group), 2011).

Se decidió utilizar este lenguaje de modelado por las facilidades y ventajas que aportan su utilización para lograr un modelado del sistema a implementar. Además brindó un vocabulario y reglas estándar para los desarrolladores propiciando una fácil comprensión de los diagramas y conceptos relacionados con la problemática planteada.

Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, en español Ingeniería de Software Asistida por Ordenador) de modelado, permiten la aplicación de métodos y técnicas que dan utilidades a los programas por medios de otros, procedimientos y respectiva documentación. Estas herramientas incluyen un conjunto de programas que facilitan la optimización de un producto ofreciendo apoyo permanente a los analistas, ingenieros de software y desarrolladores (Juan D Castellanos_Fundación Universitaria, 2008).

Son un conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida, mediante su uso se reduce el tiempo, costo del desarrollo y mantenimiento del software, así como se mejora su calidad (Pressman, 2010).

Visual Paradigm 8.0

Es una herramienta CASE multiplataforma que cuenta con versiones gratuitas y provee fácil integración con el resto de las herramientas de desarrollo. Ayuda a una más rápida construcción de aplicaciones de calidad y con menor costo. También permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación (Visual Paradigm, 2015).

Se decidió utilizar esta herramienta en el desarrollo de la aplicación por las múltiples ventajas que posee al brindarle soporte a todo el ciclo de vida de implementación del software a desarrollar.

1.4.3 Lenguajes de programación

Un lenguaje de programación es aquel elemento dentro de la informática que nos permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes (Editboard.com, 2008).

PHP 5.5

PHP (acrónimo recursivo de *PHP: Hypertext Preprocessor*) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. Además, permite aplicar técnicas de programación orientada a objetos. Es un lenguaje completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. No requiere definición de tipos de variables aunque se pueden evaluar por el tipo que estén manejando en tiempo de ejecución.

Dentro de las ventajas que posee el mismo se encuentran que es de fácil aprendizaje, multiplataforma, multiparadigma, puede ser conectado con la mayoría de los gestores de base de datos e incluye una gran cantidad de funciones. Su uso está muy difundido en el mundo de los programadores debido a su flexibilidad (Alvarez, 2001). Por estas razones y facilidades que presenta este lenguaje para la programación web del lado del servidor se decidió utilizar y de esta forma aprovechar las ventajas argumentadas.

JavaScript

JavaScript (a veces abreviado como JS) es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web. Es un lenguaje script multiparadigma, basado en prototipos, dinámico y soporta estilos de programación funcional, orientada a objetos. Además permite ejecutar instrucciones como respuesta a las acciones del usuario, permitiendo la validación de los datos introducidos por los usuarios (Mozilla Developer Netwok, 2015).

jQuery

jQuery es una biblioteca de JavaScript rápida, pequeña, y rica en características. Hace cosas como recorrido y manipulación de documentos HTML, manejo de eventos, animación, y Ajax mucho más simple con un API (en español interfaz de programación de aplicaciones) fácil de usar que funciona a través de una multitud de navegadores. Está catalogado como software libre y de código abierto permitiendo su uso en proyectos libres y privados.

jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio (The jQuery Foundation, 2016).

Lenguaje de Etiquetado *de Hipertexto* (HTML5)

Lenguaje de Etiquetado de Hipertexto *(HyperText* Markup Language). Es un lenguaje comúnmente utilizado para la publicación de hipertexto en la Web y desarrollado con la idea de

que cualquier persona o tipo de dispositivo pueda acceder a la información en la Web. HTML utiliza etiquetas que marcan elementos y estructuran el texto de un documento (W3C, 2010).

La quinta revisión de este lenguaje pretende remplazar al (X) HTML, corrigiendo problemas con los que los desarrolladores web se encuentran, así como rediseñar el código y actualizándolo a nuevas necesidades que demanda la web de hoy en día. Establece una serie de nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos. Proporciona nuevas funcionalidades a través de una interfaz estandarizada y permite renderizar elementos en tercera dimensión en tercera dimensión (Departamento de Ciencias de la Computación en Inteligencia Artificial, Universidad de Alicante, 2011)

1.4.4 Framework

Un framework, en el desarrollo de software, es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. (CODEBOX, 2010)

Symfony 2.8

Symfony es un marco de trabajo o framework muy completo, posee una librería cohesiva de clases escritas en PHP. Provee una arquitectura, componentes y herramientas a los desarrolladores para construir aplicaciones web complejas de forma rápida y a la vez permite proporcionar un mantenimiento constante. Está basado en la experiencia, usa las mejores prácticas del desarrollo web y ha sido un proyecto de Código Abierto por más de cuatro años. Se ha convertido en el framework de PHP más popular hoy en día y se ha creado una gran comunidad donde se puede encontrar soporte, documentación y aplicaciones libres. (Symfony, 2011)

Algunas de las ventajas que brinda son:

- Fácil de configurar y adaptable a sus necesidades.
- Independiente del sistema gestor de bases de datos. Permite cambiar con facilidad de Sistema Gestor de Base de Datos (SGBD) en cualquier fase del proyecto.
- Sigue la mayoría de mejores prácticas y patrones de diseño web.
- Open Source.
- Flexibilidad + diseño + configuración + plugins.
- Reutilización de componentes.
- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.
- Plataforma independiente.

- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5 o superior.
- Accesos simples a la Base de Datos. (Ruiz, 2008)

Se decidió utilizar este framework por las ventajas que posee en la programación web, además de tener una estructura robusta y aportar facilidades en cuanto a funcionalidades específicas desarrolladas en bundles de terceros, algunos de estos componentes se describen a continuación:

knp-snappy: bundle para generar archivos pdf a partir de una página con formato HTML. Este bundle resulta de gran ayuda para exportar informes o reportes a formato pdf y de esta forma poder almacenar para realizar un análisis de la información plasmada cuando sea necesario.
easyadmin-bundle: este bundle es una solución interesante para la administración de sitios debido a su estructura e implementación constituye un elemento de apoyo a los programadores al sintetizar de manera eficiente la estructura del código de las aplicaciones a desarrollar.
uploader-bundle: este bundle realiza la función de cargar archivos con formato pdf para complementar la información recopilada en un sistema que gestione información.

Bootstrap 3.0

Bootstrap es un framework o conjunto de herramientas de software libre desarrollado por twitter para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales.

Estructura y Función

Bootstrap es modular y consiste esencialmente en una serie de hojas de estilo dinámicas (LESS) que implementan la variedad de componentes de la herramienta. Una hoja de estilo llamada bootstrap.less incluye los componentes de las hojas de estilo. Los desarrolladores pueden adaptar el mismo archivo de Bootstrap, seleccionando los componentes que deseen usar en su proyecto. Los ajustes son posibles en una medida limitada a través de una hoja de estilo de configuración central. Los cambios más profundos son posibles mediante las declaraciones LESS. El uso del lenguaje de hojas de estilo dinámico permite el uso de variables, funciones y operadores, selectores anidados, así como clases mixin. Desde la versión 2.0, la configuración de Bootstrap también tiene una opción especial de "Personalizar" en la documentación. Por otra parte, los desarrolladores eligen en un formulario los componentes y ajustes deseados, y de ser necesario, los valores de varias opciones a sus necesidades. El paquete consecuentemente generado ya incluye la hoja de estilo CSS (Flexible information technology, 2016).

Algunas características:

- Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como tablets y móviles a distintas escalas y resoluciones.
- Se integra perfectamente con las principales librerías JavaScript, por ejemplo JQuery.
- Ofrece un diseño sólido usando LESS y estándares como CSS3/HTML5.
- Funciona con todos los navegadores, incluido Internet Explorer.
- Dispone de distintos layout predefinidos con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos (GENBETA, 2012).

En las características detalladas previamente

1.4.5 Servidor web

Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente y generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se usa el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al ordenador que ejecuta el programa. (Valle, 2015)

Apache 2.2

Apache 2 es un servidor web multiplataforma de código abierto. Es modular (basado en módulos), donde cada módulo ofrece un grupo de funcionalidades específicas al servidor. Es uno de los servidores web más utilizado en Internet, lo que facilita el acceso a la documentación. Provee un alto nivel de seguridad y eficiencia, permitiendo además el uso de una versión local, la cual hace posible que el servidor actúe como servidor y cliente al mismo tiempo, creando así la posibilidad de pre visualizar y probar el código mientras este es desarrollado (The Apache Software Fundation, 1997)

Partiendo de las características analizadas se resolvió usar para la implementación del sistema el servidor web Apache 2.2, teniendo en cuenta además, el lenguaje de programación web escogido para la programación del lado del servidor.

1.4.6 Servidor de Base de Datos

Los sistemas de gestión de bases de datos o SGBD (en inglés Database Management System, abreviado DBMS) son un tipo de software dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Son grandes proveedores de información para todo

tipo de usuarios. Ellos deben ofrecer soluciones de forma fiable, rentable y de alto rendimiento. A estas tres características, se le debe añadir una más: debe proporcionar servicios de forma global y, en la medida de lo posible, independientemente de la plataforma. Ejemplos de estos servidores son: MySQL, Oracle-XE y PostgreSQL (DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA, 2015).

Para el desarrollo del sistema se decidió usar el gestor de bases de datos PostgreSQL 9.3 por las características y ventajas que presenta.

PostgreSQL 9.3

Es un sistema de gestión de base de datos relacional orientada a objetos, libre y publicado bajo la licencia BSD. Además es el gestor de bases de datos de código abierto más avanzado en la actualidad, ofreciendo control de concurrencia multiversión, soportando casi todas las sintaxis SQL como subconsultas, transacciones y funciones definidas por el usuario, cuenta además con un amplio conjunto de enlaces con lenguajes de programación como C, C++, Java y Python. PostgreSQL está dirigido por una comunidad bien organizada de desarrolladores y organizaciones comerciales, de ahí el nivel de aceptación y de calidad del mismo.

Algunas de las características que posee son:

- Poseer una gran escalabilidad. Es capaz de ajustarse al número de Unidades Centrales de Procesamiento (CPU) y a la cantidad de memoria que posee el sistema de forma óptima.
- Tener la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- La simplificación del proceso de administración de licencias de software, que no es necesario cuando se usa software libre.
- Poder lidiar con grandes volúmenes de datos. (PostgreSQL, 2015)

1.4.7 Aplicación gráfica para gestionar el gestor de bases de datos. PgAdmin III.

PgAdmin es la más popular plataforma de código abierto para el desarrollo con PostgreSQL. Rico en funciones para la administración y desarrollo para PostgreSQL. Puede ser usado en las plataformas Linux, FreeBSD, Solaris, MacOSX y Windows. Está diseñado con el fin de satisfacer la necesidad de todos los usuarios, para escribir simples sentencias SQL y desarrollar complejas bases de datos. La interfaz gráfica soporta todas las características de PostgreSQL y hace más fácil la administración. La aplicación además incluye un editor de sintaxis SQL, editor de código del lado del servidor y una consola. Las conexiones al servidor pueden ser realizadas usando TCP/IP o Unix Domain Sockets (para plataformas Unix). Además de tener la posibilidad de usar encriptado SSL para seguridad. No requiere drivers

adicionales para conectarse con el servidor de base de datos. PgAdmin es desarrollado por expertos de la comunidad de PostgreSQL alrededor del mundo y se encuentra disponible en una docena de idiomas. Fue liberado bajo licencia libre PostgreSQL. (Guía de Ubuntu, 2010)

1.4.8 Entorno integrado de desarrollo IDE

Para la ejecución del sistema se hizo necesario escoger un entorno integrado de desarrollo que proporcionara el uso y la integración de todas las tecnologías y lenguajes antes mencionados, por lo que se decidió utilizar el PhpStorm-9.0, por ser un IDE ligero y tener un autocompletado muy aceptable en la programación web, específicamente para el lenguaje PHP y además por integrarse muy bien al framework symfony.

JetBrains PhpStorm 9.0

JetBrains PhpStorm es un IDE multiplataforma para PHP desarrollado sobre la plataforma JetBrains IntelliJ IDEA. Proporciona un editor para PHP, HTML y JavaScript con el análisis de código en la marcha, la prevención de errores y refactorizaciones automáticas para PHP y el código JavaScript. También posee una compatibilidad ajustada y óptima con diferentes frameworks cómo son Symfony, Drupal, WordPress, Zend Framework, laravel, Magento, CakePHP, Yii, y otros frameworks de menor trascendencia (JetBrains, 2015).

La idea de seleccionar PhpStorm 9.0 como IDE parte de su relación con Symfony, el framework de desarrollo escogido, ya que PhpStorm en esta versión le da soporte, facilitando el trabajo del programador con el framework. De esta manera se tiene completamiento de código no solo de la sintaxis de PHP sino de las clases creadas que se generen dentro del proyecto.

1.5 Conclusiones del capítulo

En este capítulo se esclarecieron los conceptos asociados con el dominio del problema y se definieron las principales tecnologías que contribuirán con el desarrollo de la solución propuesta, aportando así una base para que los investigadores tengan una mejor visión del ámbito donde se desarrolla la investigación. A partir de estos conceptos se encaminó el estudio del estado del arte de sistemas que gestionan información, evidenciándose que no resuelven el problema planteado. Aun así, estos sistemas aportan elementos importantes para el desarrollo del sistema que se propone realizar. Se hizo además un análisis profundo del objeto de estudio, el cual permitió tener una noción más nítida del problema a resolver. Finalmente la selección de las herramientas y tecnologías seleccionadas permitió definir el entorno técnico donde se desarrollará la solución propuesta.

Capítulo 2: Características y diseño del sistema para la gestión de información.

En el capítulo se definen las características y se realiza el diseño del sistema propuesto. Incluye el modelo de dominio, la definición de los requisitos funcionales y no funcionales. Contiene además los actores y casos de uso representados en el diagrama de casos de uso del sistema y las descripciones textuales correspondientes a cada caso de uso. Se presentan los patrones de diseño junto al estilo arquitectónico utilizado. Se muestra el diagrama de clases de diseño del caso de uso Gestionar Planteamiento. Además se muestra el diagrama de despliegue, con una breve descripción de los nodos que lo conforman, así como el modelo de datos con la descripción de las tablas más significativas.

2.1 Modelo de Dominio

Un modelo de dominio captura los tipos más importantes de objetos en el contexto del sistema. Los objetos del dominio representan los eventos que suceden en el entorno en el que trabaja el sistema (Sommerville, 2009).

Para la presente investigación se plantea el uso de un modelo de dominio ya que no se tiene una estructura definida de los procesos del negocio, no se cuenta con un manual de procedimientos y los flujos de información son difusos. A continuación, se presenta el modelo de dominio.

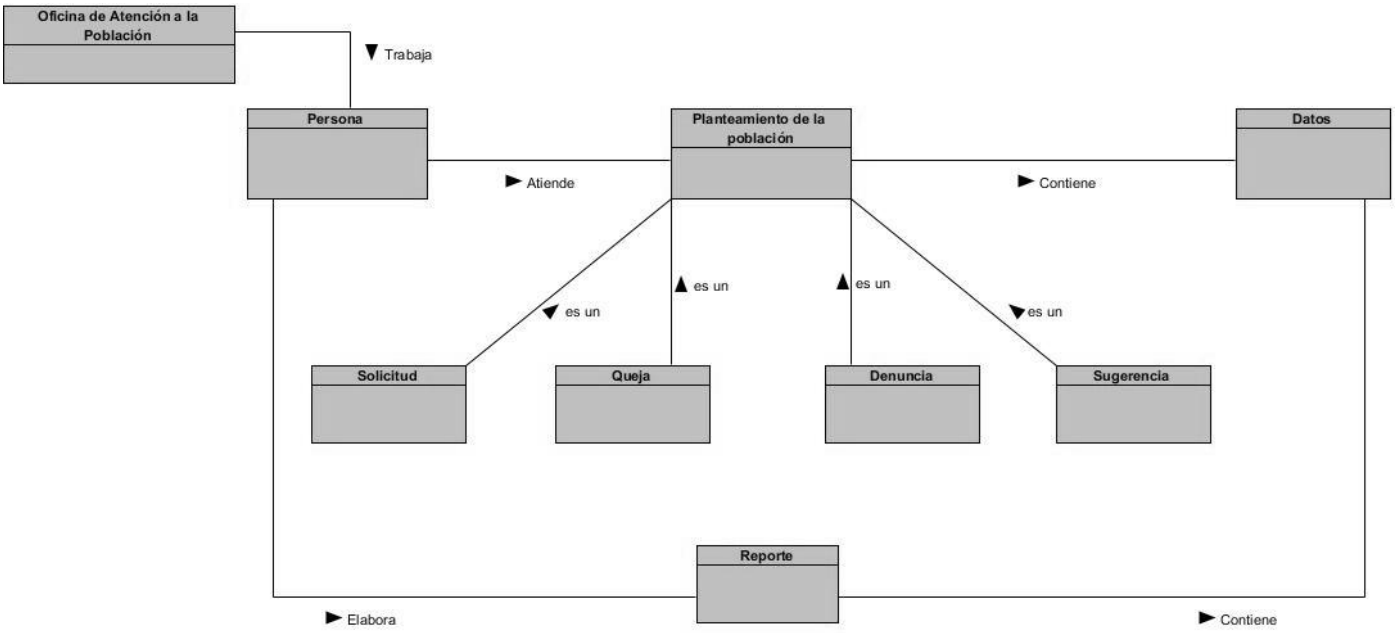


Fig. 1: Modelo de Dominio

2.1.1 Descripción general del Modelo de Dominio

En la Oficina de Atención a la Población trabaja una persona la cual se encarga de atender los diferentes planteamientos realizados por la población universitaria. Estos planteamientos están compuestos por una serie de datos y a su vez se clasifican en Solicitudes, Quejas y Denuncias. La persona encargada además realiza una serie de reportes los cuales contienen la información recogida en los planteamientos administrados.

2.1.2 Descripción de los conceptos del Modelo de Dominio

Oficina de Atención a la Población: Hace referencia al local donde se lleva a cabo todo el proceso de gestión de planteamientos.

Usuario: Es el personal del centro que interactúa con los recursos existentes.

Solicitud: Es una estructura organizacional que agrupa y recoge información planteada en su estructura relacionada con peticiones a la entidad.

Queja: Es una estructura organizacional que agrupa y recoge información planteada en su estructura relacionada con inconformidades de la población.

Denuncia: Es una estructura organizacional que agrupa y recoge información referente a desacuerdos o inconformidades.

Sugerencia: Es una estructura organizacional que agrupa y recoge información planteada en su estructura referente a diferentes aspectos mejorables que denoten alguna mejora de funcionamiento el algún proceso llevado a cabo.

Departamento: Es la estructura con mayor nivel organizacional, dentro de la cual se encuentran las Solicitudes, Quejas, Denuncias y Sugerencias hechas por la población.

Datos: Se compone por un conjunto de datos especificados por el usuario, asociados a un planteamiento realizado.

Reporte: Archivos generados por Sistema Gestión de Información a partir de información obtenida de la en los datos recopilados.

2.2 Requisitos del software

Los requisitos de software constituyen las necesidades de los clientes, las funcionalidades que debe cumplir el sistema software y las restricciones que debe cumplir. Los mismos se clasifican en funcionales y no funcionales (Pressman, 2010). A continuación se presentan los requisitos del sistema para ver una descripción más detallada de cada uno de estos:

2.2.1 Requisitos Funcionales

Los requisitos funcionales son condiciones o capacidades con los que debe cumplir el sistema, son acuerdos entre el cliente y los desarrolladores sobre lo que debe o no debe hacer el sistema (Jacobson, y otros, 2000).

RF 1 Adicionar usuario.

Descripción: Se registra en el sistema un nuevo usuario.

Entrada: Se registran los detalles del nuevo usuario (usuario, nombre, apellidos y contraseña).

Salida: Se introduce un nuevo usuario en la base de datos y se actualiza el listado de usuarios.

RF 2 Modificar usuario.

Descripción: Permitirá modificar cualquiera de los campos que componen a la entidad usuario.

Entrada: Cambio en los campos que se desean modificar del usuario.

Salida: Se modifican los datos del usuario en la base de datos.

RF 3 Eliminar usuario.

Descripción: Se elimina del sistema el usuario seleccionado previamente.

Entrada: Se selecciona de la lista de usuarios el que se desea eliminar.

Salida: Se actualiza la lista de usuarios.

RF 4 Visualizar usuario.

Descripción: Se visualizan los datos del usuario seleccionado previamente.

Entrada: Se selecciona de la lista de usuarios el que se desea visualizar.

Salida: Se visualizan todos los datos del usuario seleccionado.

RF 5 Autenticar usuario.

Descripción: Permitirá a los usuarios registrados realizar el proceso de autenticación en la aplicación ofreciendo a la misma los datos correspondientes (usuario y contraseña)

Entrada: Contraseña y usuario de la persona que desea autenticarse en el sistema.

Salida: El usuario autenticado ingresa al sistema donde puede acceder y ejecutar todas las funcionalidades.

RF 6 Listar usuario.

Descripción: Permitirá a acceder al listado de usuarios creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Gestión de Usuarios.

Salida: El sistema muestra un listado con los usuarios creados, en caso de no existir ninguno creado el listado estará vacío.

RF 7 Adicionar planteamiento.

Descripción: Se registra en el sistema un nuevo planteamiento.

Entrada: Detalles del nuevo planteamiento que se desea adicionar (nombre, apellidos, fecha de recepción, fecha de respuesta, título, estado, tiempo de expiración, clasificación, tipo de caso, involucrado, área, evaluación, satisfacción, medida disciplinaria y resultados de la auditoría).

Salida: Se introduce en la base de datos un nuevo planteamiento y se actualiza el listado de planteamientos.

RF 8 Modificar planteamiento.

Descripción: Permitirá modificar cualquiera de los campos que componen a la entidad Planteamiento.

Entrada: El cambio en los campos que desean modificar del planteamiento.

Salida: Planteamiento modificado.

RF 9 Eliminar planteamiento.

Descripción: Se elimina del sistema el planteamiento seleccionado previamente.

Entrada: Se selecciona de la lista de planteamiento el que se desea eliminar.

Salida: Se actualiza el listado de planteamientos.

RF 10 Visualizar planteamiento.

Descripción: Se visualizan los datos del planteamiento seleccionado previamente.

Entrada: Se selecciona de la lista de planteamiento el que se desea visualizar.

Salida: Se visualizan todos los datos del planteamiento.

RF 11 Listar planteamiento.

Descripción: Permitirá a acceder al listado de planteamientos creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Planteamientos.

Salida: El sistema muestra un listado con los planteamientos creados, en caso de no existir ninguno creado el listado estará vacío.

RF 12 Adicionar clasificación.

Descripción: Se registra en el sistema una nueva clasificación.

Entrada: La nueva clasificación que se desea a adicionar.

Salida: Se introduce en la base de datos una nueva clasificación y se actualiza el listado de clasificaciones.

RF 13 Modificar clasificación.

Descripción: Permitirá modificar el campo que compone a la entidad Clasificación.

Entrada: El cambio en el campo que compone a la clasificación.

Salida: Clasificación modificada.

RF 14 Eliminar clasificación.

Descripción: Se elimina del sistema la clasificación seleccionada previamente.

Entrada: Se selecciona de la lista de clasificación la que se desea eliminar.

Salida: Se actualiza el listado de clasificaciones.

RF 15 Listar clasificación.

Descripción: Permitirá a acceder al listado de clasificaciones creadas.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Clasificación.

Salida: El sistema muestra un listado con las clasificaciones creadas, en caso de no existir ninguna creada el listado estará vacío.

RF 16 Adicionar comisión.

Descripción: Se registra en el sistema una nueva comisión.

Entrada: Detalles de la nueva comisión que se desea a adicionar (título, descripción e informe).

Salida: Se introduce en la base de datos una nueva comisión y se actualiza el listado de comisiones.

RF 17 Modificar comisión.

Descripción: Permitirá modificar cualquiera de los campos que componen a la entidad Comisión.

Entrada: El cambio en los campos que desean modificar de la comisión.

Salida: Comisión modificada.

RF 18 Eliminar comisión.

Descripción: Se elimina del sistema la comisión seleccionada previamente.

Entrada: Se selecciona de la lista de comisión la que se desea eliminar.

Salida: Se actualiza la lista de comisiones.

RF 19 Visualizar comisión.

Descripción: Se visualiza el dato que compone la comisión seleccionada previamente.

Entrada: Se selecciona de la lista de comisión la que se desea visualizar.

Salida: Se visualizan los datos de la comisión.

RF 20 Listar comisión.

Descripción: Permitirá a acceder al listado de comisiones creadas.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Comisión.

Salida: El sistema muestra un listado con las comisiones creadas, en caso de no existir ninguna creada el listado estará vacío.

RF 21 Adicionar tipo de caso.

Descripción: Se registra en el sistema un nuevo tipo de caso.

Entrada: Detalles del nuevo tipo de caso que se desea a adicionar (tipo de caso).

Salida: Se introduce en la base de datos un nuevo tipo de caso y se actualiza el listado de tipos de casos.

RF 22 Modificar tipo de caso.

Descripción: Permitirá modificar el campo que compone la entidad Tipo_Caso.

Entrada: El cambio en el campo que compone el tipo de caso.

Salida: Tipo de caso modificado.

RF 23 Eliminar tipo de caso.

Descripción: Se elimina del sistema el tipo de caso seleccionado previamente.

Entrada: Se selecciona de la lista de tipos de casos el que se desea eliminar.

Salida: Se actualiza la lista de tipo de caso.

RF 24 Listar tipo de caso.

Descripción: Permitirá a acceder al listado de tipos de casos creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Tipo de Caso.

Salida: El sistema muestra un listado con los tipos de casos creados, en caso de no existir ninguno creado el listado estará vacío.

RF 25 Adicionar involucrado.

Descripción: Se registra en el sistema un nuevo involucrado.

Entrada: Detalles del nuevo involucrado que se desea a adicionar (involucrado).

Salida: Se introduce en la base de datos un nuevo involucrado.

RF 26 Modificar involucrado.

Descripción: Permitirá modificar el campo que compone a la entidad Involucrado.

Entrada: El cambio en el campo que compone el involucrado.

Salida: Involucrado modificado.

RF 27 Eliminar involucrado.

Descripción: Se elimina del sistema el involucrado seleccionado previamente.

Entrada: Se selecciona de la lista de involucrado el que se desea eliminar.

Salida: Se actualiza la lista de involucrados.

RF 28 Listar involucrado.

Descripción: Permitirá a acceder al listado de involucrados creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Involucrado.

Salida: El sistema muestra un listado con los involucrados creados, en caso de no existir ninguno creado el listado estará vacío.

RF 29 Adicionar área.

Descripción: Se registra en el sistema una nueva área.

Entrada: Detalles de la nueva área que se desea a adicionar (nombre del área).

Salida: Se introduce en la base de datos una nueva área y se actualiza el listado de áreas.

RF 30 Modificar área.

Descripción: Permitirá modificar el campo que compone a la entidad Area.

Entrada: El cambio en el campo que compone el área.

Salida: Área modificada.

RF 31 Eliminar área.

Descripción: Se elimina del sistema el área seleccionada previamente.

Entrada: Se selecciona de la lista de áreas la que se desea eliminar.

Salida: Se actualiza el listado de áreas.

RF 32 Listar área.

Descripción: Permitirá a acceder al listado de áreas creadas.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Área.

Salida: El sistema muestra un listado con las áreas creadas, en caso de no existir ninguna creada el listado estará vacío.

RF 33 Adicionar tipo de satisfacción.

Descripción: Se registra en el sistema un nuevo tipo de satisfacción.

Entrada: Detalles del nuevo tipo de satisfacción que se desea a adicionar (tipo de satisfacción).

Salida: Se introduce en la base de datos un nuevo tipo de satisfacción y se actualiza el listado de tipo de satisfacción.

RF 34 Modificar tipo de satisfacción.

Descripción: Permitirá modificar el campo que compone a la entidad Satisfaccion.

Entrada: El cambio en el campo que compone el tipo de satisfacción.

Salida: Tipo de satisfacción modificado.

RF 35 Eliminar nivel de satisfacción.

Descripción: Se elimina del sistema el tipo de satisfacción seleccionado previamente.

Entrada: Se selecciona de la lista de tipo de satisfacción el que se desea eliminar.

Salida: Se actualiza el listado de tipo de satisfacción.

RF 36 Listar nivel de satisfacción.

Descripción: Permitirá a acceder al listado de niveles de satisfacción creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Nivel de Satisfacción.

Salida: El sistema muestra un listado con los niveles de satisfacción creados, en caso de no existir ninguno creado el listado estará vacío.

RF 37 Adicionar resultados de auditoría.

Descripción: Se registra en el sistema un nuevo resultado de auditoría.

Entrada: Detalles del nuevo resultado de la auditoría que se desea adicionar (resultado de auditoría).

Salida: Se introduce en la base de datos un nuevo resultado de auditoría y se actualiza el listado de resultados de auditorías.

RF 38 Modificar resultados de auditoría.

Descripción: Permitirá modificar el campo que compone la entidad Auditoria.

Entrada: El cambio en el campo que compone resultado de auditoría.

Salida: Campo resultado de auditoría modificado.

RF 39 Eliminar resultados de auditoría.

Descripción: Se elimina del sistema el resultado de auditoría seleccionado previamente.

Entrada: Se selecciona del listado de resultados de auditorías el que se desea eliminar.

Salida: Se actualiza el listado de resultados de auditorías y se actualiza el listado de resultados.

RF 40 Listar resultados de auditoría.

Descripción: Permitirá a acceder al listado de resultados de auditorías creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Resultados de Auditoría.

Salida: El sistema muestra un listado con los resultados de auditorías creados, en caso de no existir ninguno cread el listado estará vacío.

RF 41 Adicionar medida disciplinaria.

Descripción: Se registra en el sistema una nueva medida disciplinaria.

Entrada: Detalles de la nueva medida disciplinaria que se desea a adicionar (medida disciplinaria).

Salida: Se introduce en la base de datos una nueva medida disciplinaria y se actualiza el listado de medidas disciplinarias.

RF 42 Modificar medida disciplinaria.

Descripción: Permitirá modificar el dato del campo que compone a la entidad MedidaDisc.

Entrada: El cambio en el dato que compone la medida disciplinaria.

Salida: Se modifica el dato del campo que compone la medida disciplinaria.

RF 43 Eliminar medida disciplinaria.

Descripción: Se elimina del sistema la medida disciplinaria seleccionada previamente.

Entrada: Se selecciona de la lista de medida disciplinaria la que se desea eliminar.

Salida: Se actualiza la lista de medidas disciplinarias.

RF 44 Listar medida disciplinaria.

Descripción: Permitirá a acceder al listado de medidas disciplinarias creadas.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Medida Disciplinaria.

Salida: El sistema muestra un listado con las medidas disciplinarias creadas, en caso de no existir ninguna creada el listado estará vacío.

RF 45 Adicionar estado.

Descripción: Se registra en el sistema un nuevo estado.

Entrada: Detalles del nuevo estado que se desea a adicionar (estado).

Salida: Se introduce en la base de datos un nuevo estado y se actualiza el listado de estados.

RF 46 Modificar estado.

Descripción: Permitirá modificar el dato del campo que compone a la entidad Estado.

Entrada: El cambio en el dato del campo estado.

Salida: Se modifica el dato del campo que compone el estado.

RF 47 Eliminar estado.

Descripción: Se elimina del sistema el estado seleccionado previamente.

Entrada: Se selecciona de la lista de estados el que se desea eliminar.

Salida: Se actualiza el listado de estados.

RF 48 Listar estado.

Descripción: Permitirá a acceder al listado de estados creados.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Estado.

Salida: El sistema muestra un listado con los estados creados, en caso de no existir ninguno creado el listado estará vacío.

RF 49 Generar reporte: Indicadores Significativos

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Indicadores Significativos.

Salida: Reporte Indicadores Significativos exportado a formato pdf.

RF 50 Generar reporte: Expirados por Tipos de Casos.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Expirados por Tipos de Casos.

Salida: Reporte Expirados por Tipos de Casos exportado a formato pdf.

RF 51 Generar reporte: Expirados por Involucrados.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Expirados por Involucrados.

Salida: Reporte Expirados por Involucrados exportado a formato pdf.

RF 52 Generar reporte: Concluidos por Tipos de Casos.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Concluidos por Tipos de Casos.

Salida: Reporte Concluidos por Tipos de Casos exportado a formato pdf.

RF 53 Generar reporte: Concluidos por Involucrados.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Concluidos por Involucrados.

Salida: Reporte Concluidos por Involucrados exportado a formato pdf.

RF 54 Generar reporte: Pendientes por Tipos de Casos.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Pendientes por Tipos de Casos.

Salida: Reporte Pendientes por Tipos de Casos exportado a formato pdf.

RF 55 Generar reporte: Pendientes por Involucrados.

Descripción: Se obtiene el reporte correspondiente en formato HTML para ser exportado a pdf.

Entrada: Se selecciona del listado de reportes el reporte Pendientes por Involucrados.

Salida: Reporte Pendientes por Involucrados exportado a formato pdf.

RF 56 Buscar elemento.

Descripción: Se obtiene el elemento correspondiente que coincida con el parámetro de búsqueda descrita.

Entrada: Se introduce un parámetro de búsqueda en el campo Buscar.

Salida: El sistema muestra el elemento que coincida con el parámetro de búsqueda introducido.

RF 57 Adicionar evaluación.

Descripción: Se registra en el sistema una nueva evaluación.

Entrada: Detalles de la nueva evaluación que se desea a adicionar (evaluación, clasificación).

Salida: Se introduce en la base de datos una nueva evaluación y se actualiza el listado de evaluaciones.

RF 58 Modificar evaluación.

Descripción: Permitirá modificar los campos que compones a la entidad Evaluacion.

Entrada: El cambio en los campos que componen la evaluación.

Salida: Se actualizan los campos modificados de la evaluación.

RF 59 Eliminar evaluación.

Descripción: Se elimina del sistema la evaluación seleccionada previamente.

Entrada: Se selecciona de la lista de evaluaciones la que se desea eliminar.

Salida: Se actualiza el listado de evaluaciones.

RF 60 Listar evaluación.

Descripción: Permitirá a acceder al listado de evaluaciones creadas.

Entrada: El usuario selecciona del menú lateral izquierdo la opción Administración y posteriormente la opción Evaluación.

Salida: El sistema muestra un listado con las evaluaciones creadas, en caso de no existir ninguna creada el listado estará vacío.

2.2.2 Requisitos no Funcionales.

Según Ian Sommerville los requisitos no funcionales (RNF) "como su nombre lo sugiere, son aquellos requerimientos que no se refieren a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento" (Sommerville, 2009). En resumen, son un conjunto de reglas que hacen del sistema, un producto agradable, usable, rápido o confiable.

Los requisitos no funcionales con que debe contar el sistema son:

Requisitos de Interfaz de usuario:

- El sistema debe tener indicadores que permitan conocer al usuario las acciones que debe realizar, por ejemplo, botones con íconos sugerentes y alternativa textual.

Restricciones de diseño e implementación:

- Lenguaje de programación: JavaScript, jQuery, bootstrap y PHP 5.5
- Lenguaje de marcado: HTML 5.
- Gestor de base de datos: PostgreSQL 9.3.

Requisitos de Software:

Para las PC clientes:

- Un navegador como Mozilla Firefox con versión 16.0 o superior.
- Sistema operativo GNU/Linux, Windows 7 o superior.

Para las PC servidor:

- Sistema operativo GNU/Linux, Ubuntu Server 14.4, Windows 7 o superior.
- Servidor Web XAMPP 3.2 o Apache 2.0 o superior, con módulo PHP 5 configurado con la extensión pgsql incluida.
- PostgreSQL 9.3 como Sistema Gestor de Base de Datos.
- Cliente de Base de Datos: PgAdmin III.

Requisitos de Hardware:

Para las PC clientes:

- 512 MB de RAM como mínimo.
- 1 GB de capacidad disponible en disco duro.

Para las PC servidor:

- 2 GB de RAM y 60 de disco duro como mínimo para el servidor de Base de Datos.
- 2.0 GHz de Procesador como mínimo.

2.3 Descripción del sistema

Según Ian Sommerville, los Casos de Uso son una técnica que se basa en escenarios para la obtención de requisitos. Son una característica fundamental de la notación de UML que se utiliza para describir modelos de sistemas orientados a objetos. En su forma más simple un caso de uso representa el tipo de interacción y los actores involucrados (Sommerville, 2009).

2.3.1 Definición de los actores

Un actor es una entidad externa al sistema representada por un ser humano, una máquina o un software que interactúa con el sistema. Representa un tipo particular de usuario del negocio más que un usuario físico, debido a que varios usuarios físicos pueden realizar el mismo papel en relación al negocio, o sea, ser instancias de un mismo actor.

La tabla 1 muestra el actor del sistema es cual es: Usuario, además describe la función que ejerce este actor dentro del funcionamiento de la aplicación.

Tabla 1: Actores que intervienen en el problema.

Actor	**Descripción**
Usuario	Es el encargado de administrar los usuarios en el sistema los cuales contarán con los mismos permisos. Además tiene acceso a todas las funcionalidades del sistema para su administración

2.4 Diagrama de casos de uso del sistema

Los diagramas de Casos de Uso del Sistema son ubicados dentro de los diagramas de comportamiento de UML y constituyen una representación gráfica de los procesos y su interacción con los actores, se utilizan para ilustrar los requisitos del sistema al mostrar cómo reacciona una respuesta a eventos que se producen en el mismo (Jacobson, y otros, 2000).

A continuación se presenta el Diagrama de Casos de Usos correspondientes a la solución propuesta en el desarrollo del Sistema Gestión de Información para la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas. Estos diagramas contienen los casos de usos necesarios para satisfacer los requisitos funcionales identificados anteriormente.

Fig. 2: Diagrama de Casos de Uso del Sistema.

2.5 Descripción de casos de uso

A continuación se expone en la siguiente tabla la descripción del caso de uso Gestionar Planteamiento.

Tabla 2: Descripción del caso de uso Gestionar Planteamiento

Objetivo	El caso de uso se lleva a cabo para manejar las operaciones referentes a los planteamientos.	
Actores	Usuario.	
Resumen	El CU se inicia cuando el usuario desea adicionar, modificar, eliminar o visualizar los planteamientos. El CU termina cuando haya sido ejecutada alguna de las funcionalidades anteriormente especificadas.	
Complejidad	Alta	
Prioridad	Crítica	
Precondiciones	El usuario debe estar autenticado previamente en el sistema. Debe existir Planteamiento creado para visualizar, eliminar y modificar un Planteamiento.	
Poscondiciones	Se adicionó un nuevo Planteamiento. Se modificó un Planteamiento. Se visualizó los datos de un Planteamiento. Se eliminó un Planteamiento.	
Referencias	RF7, RF8, RF9, RF10, RF11	
Flujo de eventos		
Flujo básico: Gestionar Planteamiento.		
	Actor	**Sistema**
1.	El caso de uso comienza cuando el usuario selecciona la opción gestionar planteamiento ubicada en el menú lateral izquierdo.	
2.		El sistema muestra un listado con todos los planteamientos, en caso de no existir planteamientos creados se muestra el listado vacío. Permitiendo realizar las siguientes operaciones: - Adicionar Planteamiento. Ver Sección 1: "Adicionar Planteamiento". - Modificar Planteamiento. Ver

		Sección 3: "Modificar Planteamiento". - Visualizar Planteamiento. Ver Sección 2 "Visualizar Planteamiento". - Eliminar Planteamiento. Ver Sección 4: "Eliminar Planteamiento".
Sección 1: "Adicionar Planteamiento".		
Flujo básico: Adicionar Planteamiento.		
	Actor	**Sistema**
1.	Selecciona la opción "Adicionar Planteamiento"	
2.		Muestra un formulario con los datos del Planteamiento:
3.	Llena los campos los siguientes campos: **Obligatorios:** Clasificación, Título, Fecha de recepción, Estado, Tipo de caso, Involucrados, Tiempo de expiración en días y Área. **Opcionales:** Nombre, Apellidos, Anónimo, Comisión disciplinaria, Medida disciplinaria, Evaluación Resultados de la auditoría, Evaluación, Fecha de respuesta, Satisfacción, Adjuntar carta de respuesta, Adjuntar informe y Descripción. **Notas:** Al seleccionar el campo anónimo se deshabilitarán los campos de Nombre y Apellidos. Al seleccionar las opciones "Solicitud o Sugerencia" los campos: Resultados de	

	la auditoría, Medida disciplinaria y Comisión disciplinaria se deshabilitarán. Los siguientes campos se habilitarán una vez que se seleccione una Fecha de respuesta: Satisfacción, Resultados de la auditoría, Medida disciplinaria, Comisión disciplinaria, Satisfacción y Evaluación. Presiona el botón "*Guardar cambios*".	
4.		Verifica que no existan campos obligatorios vacíos.
5.		Guarda los datos del nuevo Planteamiento.
6.		Termina el caso de uso
Flujos alternos		
4.1: El actor mantiene campos obligatorios vacíos.		
	Actor	**Sistema**
4.1.1		Muestra un mensaje notificando que existen campos obligatorios vacíos.
4.1.2		Vuelve a ejecutar el paso 3 de la Sección 1.
Sección 2: "Modificar Planteamiento".		
Flujo básico: Modificar Planteamiento.		
	Actor	Sistema
1	Selecciona la opción "Modificar" de un planteamiento previamente seleccionado.	
2		Muestra un formulario con los datos del Planteamiento:
3	Actualiza los campos deseados: **Obligatorios:** Clasificación, Título, Fecha de recepción, Estado, Tipo de caso, Involucrados, Tiempo de	

	expiración en días y Área. **Opcionales:** Nombre, Apellidos, Anónimo, Comisión disciplinaria, Medida disciplinaria, Resultados de la auditoría, Evaluación, Fecha de respuesta, Satisfacción, Adjuntar carta de respuesta, Adjuntar informe y Descripción. **Notas:** Al seleccionar el campo anónimo se deshabilitarán los campos de Nombre y Apellidos. Al seleccionar las opciones "Solicitud o Sugerencia" los campos: Resultados de la auditoría, Medida disciplinaria y Comisión disciplinaria se deshabilitarán. Los siguientes campos se habilitarán una vez que se seleccione una Fecha de respuesta: Satisfacción, Resultados de la auditoría, Medida disciplinaria, Comisión disciplinaria, Satisfacción y Evaluación. Presiona el botón "*Guardar cambios*.	
4		Verifica que no existan campos obligatorios vacíos.
5		Actualiza los datos del Planteamiento que hayan sido modificados.
6		Termina el Caso de Uso.
Flujos alternos		
4.1: El actor mantiene campos obligatorios vacíos.		
	Actor	Sistema
4.1.1		Muestra un mensaje notificando que existen campos obligatorios vacíos.

4.1.2		Vuelve a ejecutar el paso 3 de la Sección 2.
Sección 3: "Visualizar Planteamiento".		
Flujo básico: Visualizar Planteamiento.		
	Actor	Sistema
1		Muestra una vista con los Planteamientos.
2	Selecciona el Planteamiento a visualizar y se presiona la opción "Ver".	
3		Muestra una vista con los datos del Planteamiento.
4		Termina el caso de uso
Sección 4: "Eliminar Planteamiento".		
Flujo básico: Eliminar Planteamiento.		
	Actor	Sistema
1	Selecciona el Planteamiento a eliminar y se presiona la opción "Borrar".	
2		Muestra un mensaje de confirmación "¿Realmente quieres borrar este elemento?" con las opciones "Cancelar" y "Borrar".
3	Presiona el botón "Borrar"	
4		Elimina la el Planteamiento. Muestra una vista actualizada de del listado de planteamientos.
5		Termina el Caso de Uso.
Flujos alternos		
4.1 El actor presiona el botón "Cancelar"		
	Actor	Sistema
		Cancela la operación y muestra la interfaz donde se encuentra el listado de Planteamientos Vuelve al paso 1 de la Sección 4.

Relaciones	CU-incluidos	No procede
	CU-extendidos	Buscar elemento.
Requisitos no funcionales	No procede	
Asuntos pendientes	No procede	

2.6 Patrón arquitectónico

Los patrones constituyen una guía para el diseño del software. Su objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Además deben ser reusables, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. Son de gran utilidad para describir las mejores prácticas, buenos diseños, y encapsulan la experiencia permitiendo su reutilización. Definen la estructura de un sistema de software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema (Gamma, y otros, 1998).

2.6.1 Patrón Modelo-Vista-Controlador (MVC)

El MVC surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC es un paradigma que divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo. A partir del uso de *frameworks* basados en el patrón MVC se puede lograr una mejor organización del trabajo y mayor especialización de los desarrolladores y diseñadores (Patrón Modelo-Vista-Controlador, 2012).

Symfony está basado en el patrón arquitectónico de diseño web MVC, el mismo está estructurado por tres niveles o componentes:

- **El Modelo** representa toda la información con la que trabaja el sistema, es decir, su lógica de negocio.
- **La Vista** transforma el modelo en una página web que permite que los usuarios interactúen con el sistema.
- **El Controlador** se encarga de procesar las interacciones o peticiones realizadas por el usuario y realiza los cambios pertinentes tanto en el modelo como en la vista (Potencier, y otros, 2016).

Symfony toma lo mejor de la arquitectura MVC y la realiza de modo que el desarrollo de aplicaciones sea rápido y sencillo. En el controlador se encuentran las acciones, las cuales son el núcleo de la aplicación, pues contienen toda la lógica de la aplicación. Estas acciones utilizan el modelo y precisan las variables para la vista. Al realizarse una petición web en una aplicación Symfony, la Uniform Resource Identifier que en español significa Identificador Uniforme de Recurso (URL) define una acción y los parámetros de la petición.

La vista es la encargada de originar las páginas que son mostradas como resultado de las acciones, donde se encuentra el layout, que es común para todas las páginas de la aplicación. La vista en Symfony está conformada por varias partes o capas, preparadas cada una de ellas especialmente para ser fácilmente transformable por la persona que normalmente trabaja con cada aspecto del diseño de las aplicaciones, esto se debe a la estructura de clases donde se evidencia la herencia a tres niveles, esta herencia está conformada por la clase base.php la cual contiene los aspectos más elementales de la vista como son: funciones java script, jQuery, elementos básicos de diseño como reglas css entre otras, de esta clase extiende la clase layout.php, en esta se encuentran los elementos de la estructura de las páginas que son generales para todas las vistas, estos elementos están conformados por menús y funcionalidades genéricas de la aplicación y finalmente se encuentran las demás páginas que contienen los aspectos más específicos de las vistas, estas páginas heredan de layout.php y por esta estructura es que las modificaciones resultan más sencillas, por ejemplo si se deseara modificar algún elemento común para las vistas esta modificación se realizaría en la clase layout.php y ahorraría mucho trabajo al no tener que hacer los cambios en cada vista del sistema, por otra parte si solo se deseara cambiar algún elemento específico de alguna página entonces bastaría con hacer el cambio en la clase correspondiente.

En el Modelo se encuentran las clases, que son generadas de forma automática según la estructura de la BD, esto es posible mediante una herramienta ORM *Object Relational Mapper* (Mapeo Relacional de Objeto en español) la cual implementa el propio *framework* y no es más que la traducción lógica de cada objeto a la lógica tradicional, este proceso convierte cada tabla de la base de datos en una clase permitiéndola utilizar como si fuera un objeto. En Symfony, el acceso y la modificación de los datos que se almacenan en la base de datos, se realiza mediante objetos. Doctrine es el motor generador que se encarga de la carga automática de datos para construir sus clases, creando la estructura y generando el código de las mismas. A medida que el desarrollo de un proyecto va avanzando, puede ser necesario agregar métodos y propiedades personalizadas en los objetos del modelo, esto trae consigo que se aumenten las tablas o columnas. Asimismo, cada vez que se modifica se deben regenerar las clases del modelo de objetos. Si se añaden los métodos personalizados en las clases que se generan, cada vez que se vuelvan a generar esas clases estos métodos se borrarían.

El objetivo del controlador es crear y devolver un objeto. Para ello, a veces obtiene información de la petición, o busca un recurso en la base de datos o guarda información en la sesión del usuario. Independientemente de lo que haga, el controlador siempre devuelve un objeto que se utiliza para generar la respuesta que se envía al usuario (Potencier, y otros, 2009).

A continuación se muestra en la siguiente figura una representación de cómo funciona este patrón arquitectónico.

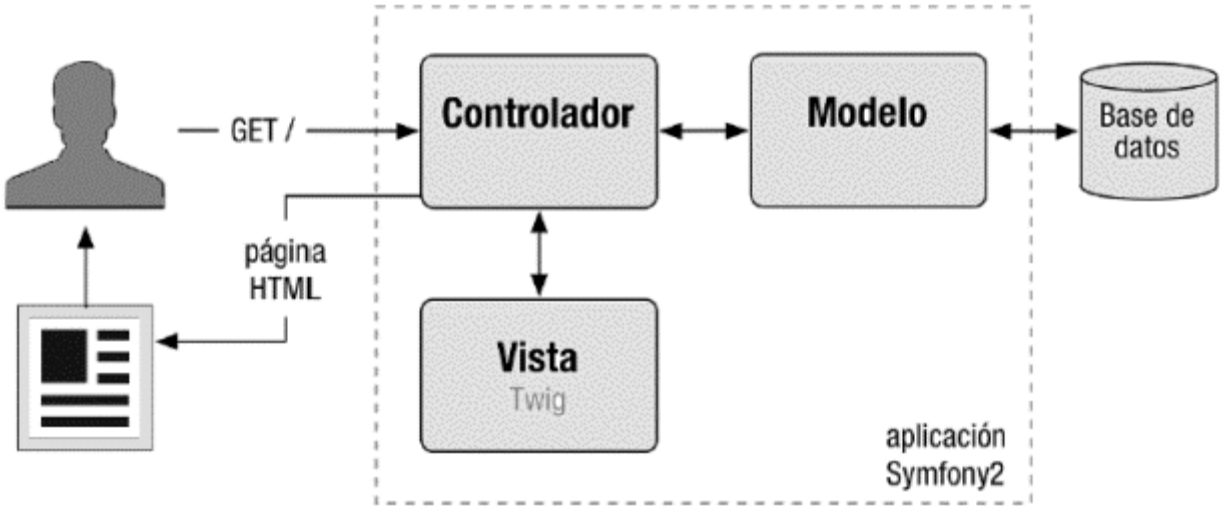


Fig. 3: MCV en symfony

2.7 Modelo de Datos

El Modelo de datos describe la estructura o representación física de las clases persistentes que constituyen las entidades asociadas al dominio del problema y que serán almacenadas en la base de datos de control de acceso. Según Pressman el modelado se responde a una serie de preguntas importantes para cualquier aplicación de procesamiento de datos (Pressman, 2010).

A continuación se presenta en la siguiente figura el diagrama Entidad Relación correspondiente al sistema a implementar:

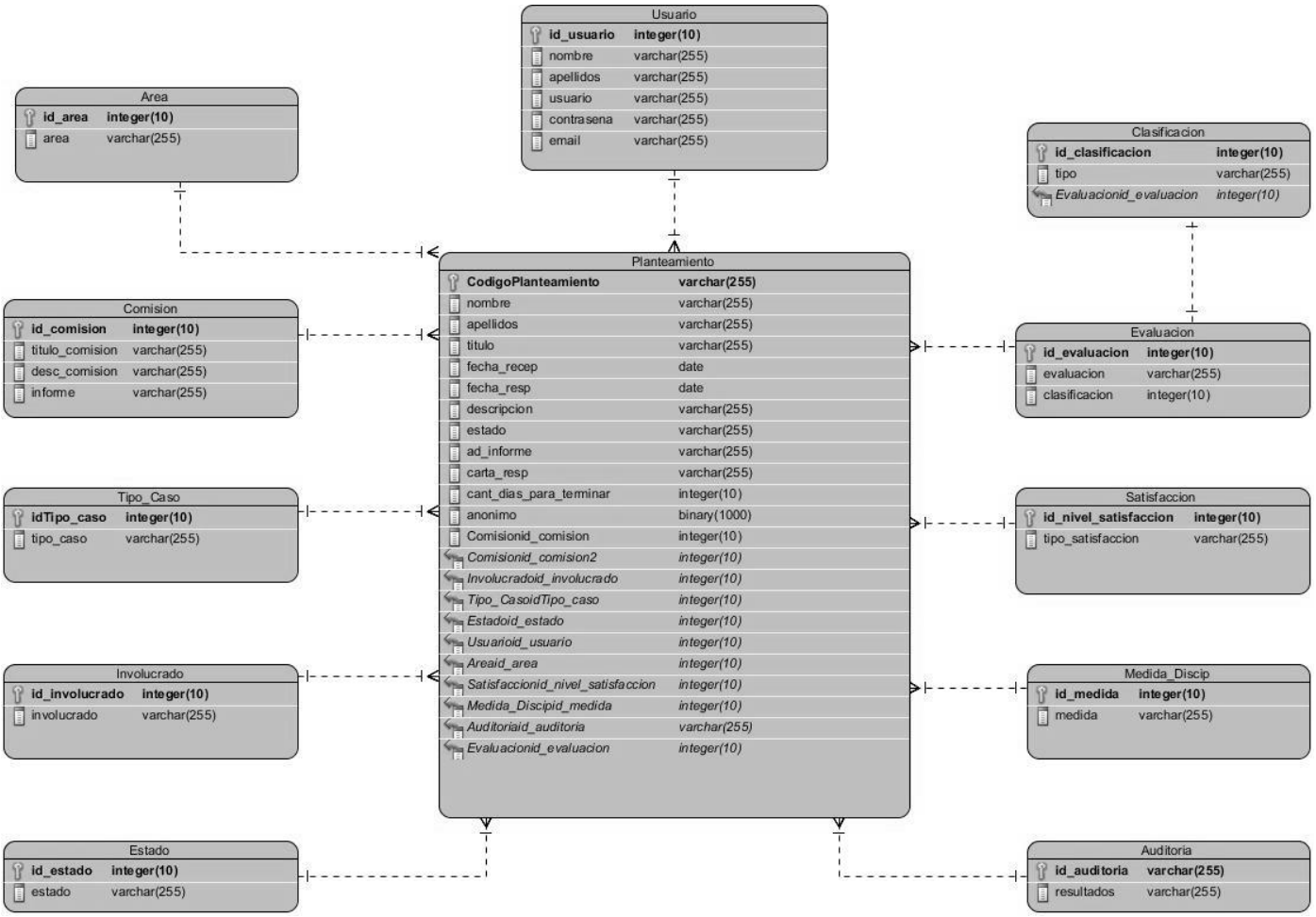


Fig. 4: Diagrama Entidad-Relación

2.7.1 Descripción de algunas de las principales tablas del modelo presentado en la figura 4.

Usuario: Tabla que contendrá los datos de todos los usuarios del sistema.

Planteamiento: Almacena la información relacionada con todos los planteamientos reportados a la Oficina de Atención a la Población y permite realizar la gestión de los mismos.

Comisión: Almacena la información relacionada con las comisiones disciplinarias creadas para atender planteamientos que por su importancia así lo requieran.

Clasificación: Almacena la información relacionada con los distintos tipos de planteamientos que se recogen en la gestión de la información.

Área: Almacena la información relacionada con las diferentes áreas que componen la locación de los planteamientos.

Estado: Almacena una información cualitativa del estado en el que está el proceso de respuesta de un planteamiento.

Medida_Disc: Almacena las diferentes medidas disciplinarias que pueden aplicarse reflejadas en los planteamientos que así lo requieran.

2.8 Patrones de diseño

Los patrones de diseño son un conjunto de estrategias, o buenas prácticas, que pueden facilitar el trabajo en muchas situaciones a la hora de realizar una aplicación orientada a objetos. Son soluciones reutilizables de problemas comunes del diseño orientado a objetos. Constituyen soluciones basadas en la experiencia que permiten flexibilizar el código haciéndolo satisfacer ciertos criterios y describir ciertos aspectos de la organización de un programa (Bass, y otros, 1998).

La asignación de responsabilidades es la habilidad más importante en el análisis y diseño orientado por objetos. Respetar los principios fundamentales es uno de los factores críticos, para obtener diseños reutilizables, de fácil mantenimiento y extensibles. Los patrones GRASP describen los principios fundamentales del diseño de objetos para la asignación de responsabilidades. Las responsabilidades están relacionadas con las obligaciones de un objeto en cuanto a su comportamiento (Bass, y otros, 1998).

Por su parte los llamados patrones GOF se clasifican en tres grupos: estructurales, creacionales y comportamiento. Los estructurales tratan la combinación de clases u objetos, su relación y la formación de estructuras de alta complejidad, mientras que los creacionales tratan la creación de instancias y los de comportamientos tratan la interacción y la cooperación entre clases (Jacobson, y otros, 2000).

2.8.1 Grasp

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns. Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software (Medel Viltres, y otros, 2014).

- **Experto:** utilizado con el objetivo de darle a las clases las responsabilidades necesarias siempre que cuentan con la información para cumplirlas. Con esto se logra un mejor comportamiento haciendo que estas fueran más cohesivas y fáciles de comprender y mantener.

  En la arquitectura de Symfony, en el modelo, existen 2 tipos de clases: las encargadas de la abstracción de datos, es decir las clases encargadas de hacer las consultas a la base de datos utilizando Doctrine ya que poseen las funcionalidades necesarias para realizar esta acción. Por otra parte las clases de acceso a datos las cuales son las responsables de interactuar con las clases de abstracción de datos, y devolver los objetos que necesitan los controladores.

Otro ejemplo de este patrón se evidencia en las clases controladoras ya que poseen la información necesaria para cumplir exactamente con la responsabilidad asignada a cada una.

- **Alta cohesión:** es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Se evidencia en las clases controladoras al repartirse las diferentes funcionalidades entre varias clases para que una clase no realice todas las funcionalidades del sistema.
- **Bajo acoplamiento:** es una medida de la fuerza con que una clase está conectada a otras, las conoce y recurre a ellas. Una clase con bajo acoplamiento no depende de muchas otras clases. Este patrón tiene como idea, tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto, potenciando la reutilización, y disminuyendo la dependencia entre las mismas.
  Con el propósito de que las clases estuvieran relacionadas solo lo necesario unas con otras, se utilizó este patrón. Definiendo para cada una sus respectivos métodos y atributos con el fin de que el acoplamiento entre ellas fuera débil, para lograr la reutilización y el soporte. La funcionalidad de este patrón se aplica en todas las clases implementadas.
- **Controlador:** consiste en asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad, etc.). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Este patrón se pone de manifiesto en las clases App_dev (controlador frontal), AdminController, ReportesController, Contex, los “actions” y el index.php del ambiente. La arquitectura del marco de trabajo (MVC) brinda una capa específicamente para los controladores, que son el núcleo del mismo y especifica la presencia de este patrón.

2.8.2 Patrones Gof

Los patrones GoF se descubren como una forma indispensable de enfrentarse a la programación. Proponen soluciones a problemas concretos, no son teorías genéricas. Se utilizan en situaciones frecuentes. Debido a que se basan en la experiencia acumulada al resolver problemas reiterativos. Favorecen la reutilización de código. Ayudan a construir software basado en la reutilización, a construir clases reutilizables (Gamma, y otros, 1998).

- **Patrón Singleton**: Este patrón garantiza que solamente se cree una instancia de las clases y provee un punto de acceso global a toda la aplicación ya que se encarga de enrutar todas las peticiones realizadas, es por ello que esta clase es utilizada por el controlador frontal de la aplicación y se evidencia su implementación en la clase Routing, utilizándose para activar los

distintos componentes del sistema que son necesarios para ejecutar la acción encomendada por el usuario.

- **Patrón Command**: Este patrón se observa en la clase App_dev, en el método dispatch (). En symfony esta clase se crea por defecto aunque en ella se pueden hacer modificaciones para personalizar su funcionamiento, esta clase es la encargada de cargar el módulo y la acción que se va a usar dependiendo de la petición especificada por el usuario.
- **Patrón Decorator**: Este patrón se evidencia en la clase base.php, padre de todas las vistas, que contiene un decorador para permitir agregar funcionalidades dinámicamente a cada página, esta clase contiene elementos de estilos css y validaciones java script y jQuery los que heredan la clase layout.php que contiene los elementos de diseño que son comunes para todas las páginas y de esta forma no tener que repetirlas en cada una. Por último se encuentran las páginas específicas según las necesidades del usuario, las cuales ya contienen estos elementos generales del layout al heredar de esta. Este patrón se evidencia en la herencia a tres niveles donde la clase base.php sirve como decorador de la clase layout.php y a su vez layout.php decora las páginas o plantillas específicas de cada función del sistema.

2.9 Modelo de diseño

Es la representación física de los Casos de Uso mediante un modelo de objetos. Centraliza su atención en el impacto que tiene en el sistema los requisitos tanto funcionales como no funcionales, así como otras restricciones del entorno de implementación (Jacobson, y otros, 2000).

2.9.1 Diagrama de Clases del Diseño

Los diagramas de clases del diseño son un tipo de diagrama estático que describe gráficamente la estructura de una aplicación. Para contribuir a la calidad del sistema se hace uso de diagramas de clases del diseño, ya que sirven de guía a los desarrolladores al constituir una aproximación del sistema que se desea implementar (Pressman, 2010). En la siguiente figura se presenta el diagrama de clases del diseño del requisito funcional Gestionar Planteamiento.

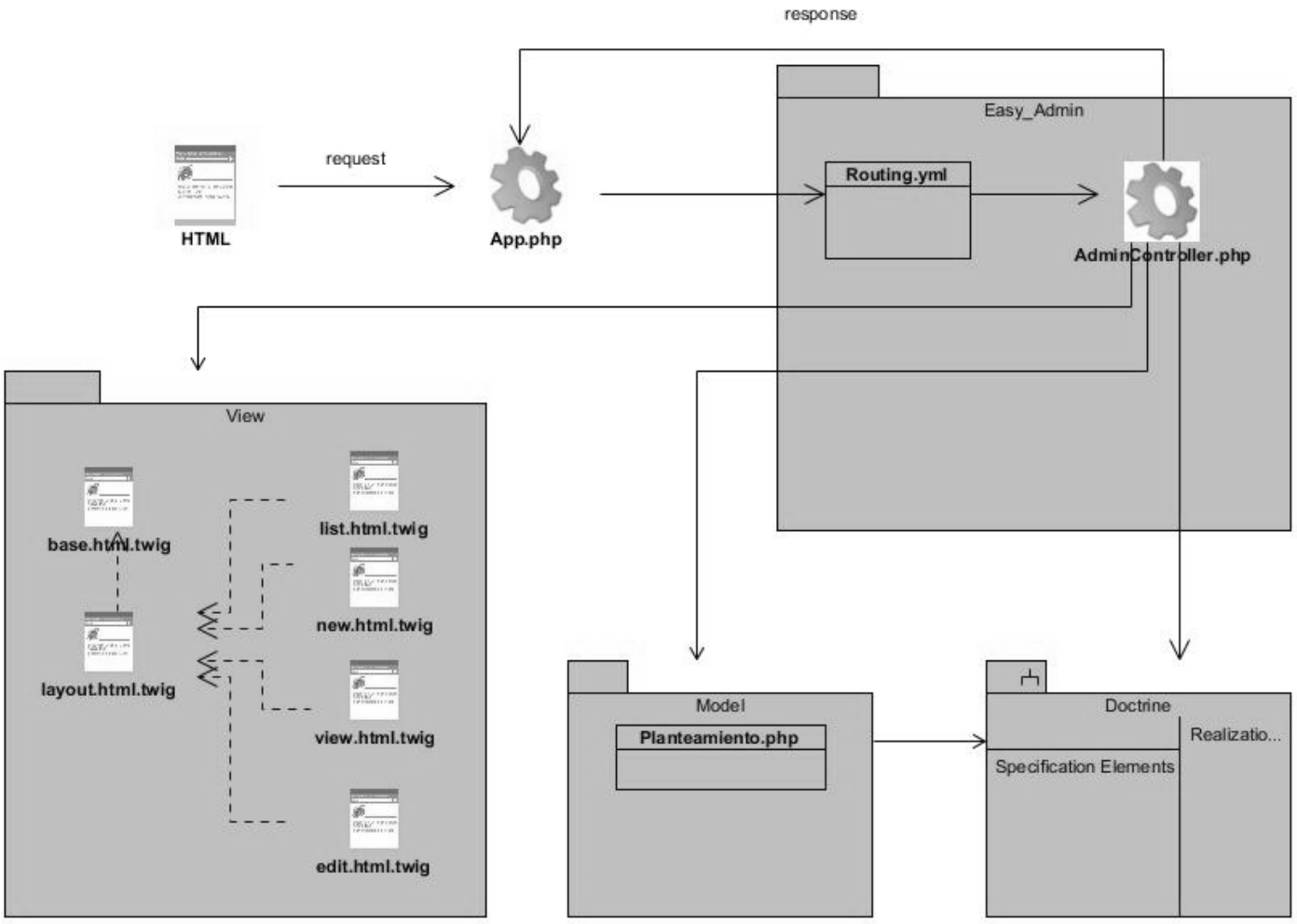


Fig. 5: Diagrama de Clases del Diseño para el CU Gestionar Planteamiento.

2.10 Conclusiones del capítulo

En este capítulo se muestran las principales características del sistema propuesto para darle solución al problema planteado, y de esta forma propiciar una mejor panorámica de las características de la solución propuesta. El diseño del Diagrama de Clases del Dominio permitió comprender a los desarrolladores los principales conceptos del entorno donde se enmarca el sistema. La realización de los diagramas de clases del diseño correspondientes a cada Caso de Uso del sistema permitió lograr una mejor interpretación para desarrollar la implementación del mismo. Es importante destacar que en la solución a implementar se hace uso del patrón Modelo Vista Controlador el cual es el patrón arquitectónico empleado para lograr un mejor funcionamiento del sistema, así como los patrones de diseño más factibles para la construcción del mismo, fundamentalmente los GRASP y GOF, esto permitió implementar el sistema siguiendo esquemas previamente definidos y estudiados por los especialistas, los cuales constituyen buenas prácticas en el ámbito del desarrollo de los sistemas de software.

Capítulo 3: Implementación y pruebas del sistema

Una vez concluido el modelo del diseño, se tienen los detalles suficientes para comenzar la construcción del sistema, y una vez concluido este, se procede a la verificación del cumplimiento de los requisitos funcionales mediante las pruebas de software. En el presente capítulo se describe cómo los elementos del modelo del diseño se implementan en términos de componentes y se muestra una descripción detallada de los paquetes de implementación. Además se presentan algunas imágenes del software para brindar una mejor comprensión del sistema. Posteriormente se pasa a la validación de la aplicación mediante las pruebas de software de caja negra, para comprobar la operatividad de las principales funcionalidades. También se expone el diagrama de despliegue, con una breve descripción de los nodos que lo conforman. Y finalmente se muestra el modelo de pruebas utilizado en el cual se expone la correcta implementación de los requisitos propuestos.

3.1 Modelo de implementación

El modelo de implementación es una correspondencia directa entre los modelos de diseño y de despliegue. Describe tanto los elementos del diseño como las clases. Se implementan en términos de componentes como ficheros de código fuente, ejecutables, entre otros. Representa además cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen los componentes unos de otros (Jacobson, y otros, 2000).

3.2 Código fuente

El código fuente es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Por tanto, en el código fuente de un programa está descrito por completo su funcionamiento. Estas instrucciones son escritas en un lenguaje de programación que consiste en un conjunto de símbolos, reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones (Arenas Cardona, 2011).

A continuación se muestra un fragmento de código perteneciente a la clase controladora de los reportes (ReportesController) que genera el sistema:

```php
<?php
/** Created by PhpStorm. ...*/

namespace RectoradoBundle\Controller;

use JavierEguiluz\Bundle\EasyAdminBundle\Controller\AdminController;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;


class ReportesController extends AdminController
{
    /*Reportes*/

    /*Reporte Indicadores Significativos*/
    public function indexReporteIndicadorAction(Request $request)
    {
        $title = 'Indicadores Significativos';
        return $this->render('Rectorado:Reportes:default', array('pie_pagina'=>false,
            'generated' => false,'title' => $title, 'template' => 'indicador'));
    }
    /*Reporte Pendientes fuera del tiempo acordado*/
    public function indexReportePendienteFTAAction(Request $request)
    {
        $title = 'Pendientes Fuera del Termino Acordado';
        return $this->render('Rectorado:Reportes:default', array('pie_pagina'=>false,
            'generated' => false,'title' => $title, 'template' => 'pendienteFTA'));
    }
```

Fig. 6: Fragmento de código de la clase ReportesController

3.3 Estándares de Codificación

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad, es de gran importancia para la calidad del software y para obtener un buen rendimiento (Microsoft, 2016).

A continuación se exponen algunas de las buenas prácticas o estándares de codificación del lenguaje php empleadas en el desarrollo del sistema.

- Todas las clases php deben estar delimitadas por las etiquetas de apertura y cierre de php (<?php>..... ?>).
- Los bloques de código siempre deben estar encerrados por llaves.
- Los nombres de los métodos deben terminar en la palabra reservada *Action.*
- Los nombres de las entidades deben comenzar con mayúsculas.

- Estilo CamelCase en la declaración de métodos en las clases.
- Tamaño = 4 (espacios) para:
    - Declaraciones dentro de las clases.
    - Enunciado dentro de métodos y funciones.
    - Enunciados dentro de bloques de comandos.

3.4 Diagrama de Componentes

Dentro del Modelo de Implementación se encuentran los diagramas de componentes. Un diagrama de componentes representa cómo un sistema de software es dividido en componentes, mostrando las dependencias que existen entre ellos. Los componentes físicos incluyen: archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables y paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema; estos son utilizados para modelar la vista estática y dinámica de un sistema. Muestran la organización y las dependencias entre un conjunto de componentes (Jacobson, y otros, 2000). La figura 7 representa el diagrama de componentes del caso de uso Gestionar Planteamiento.

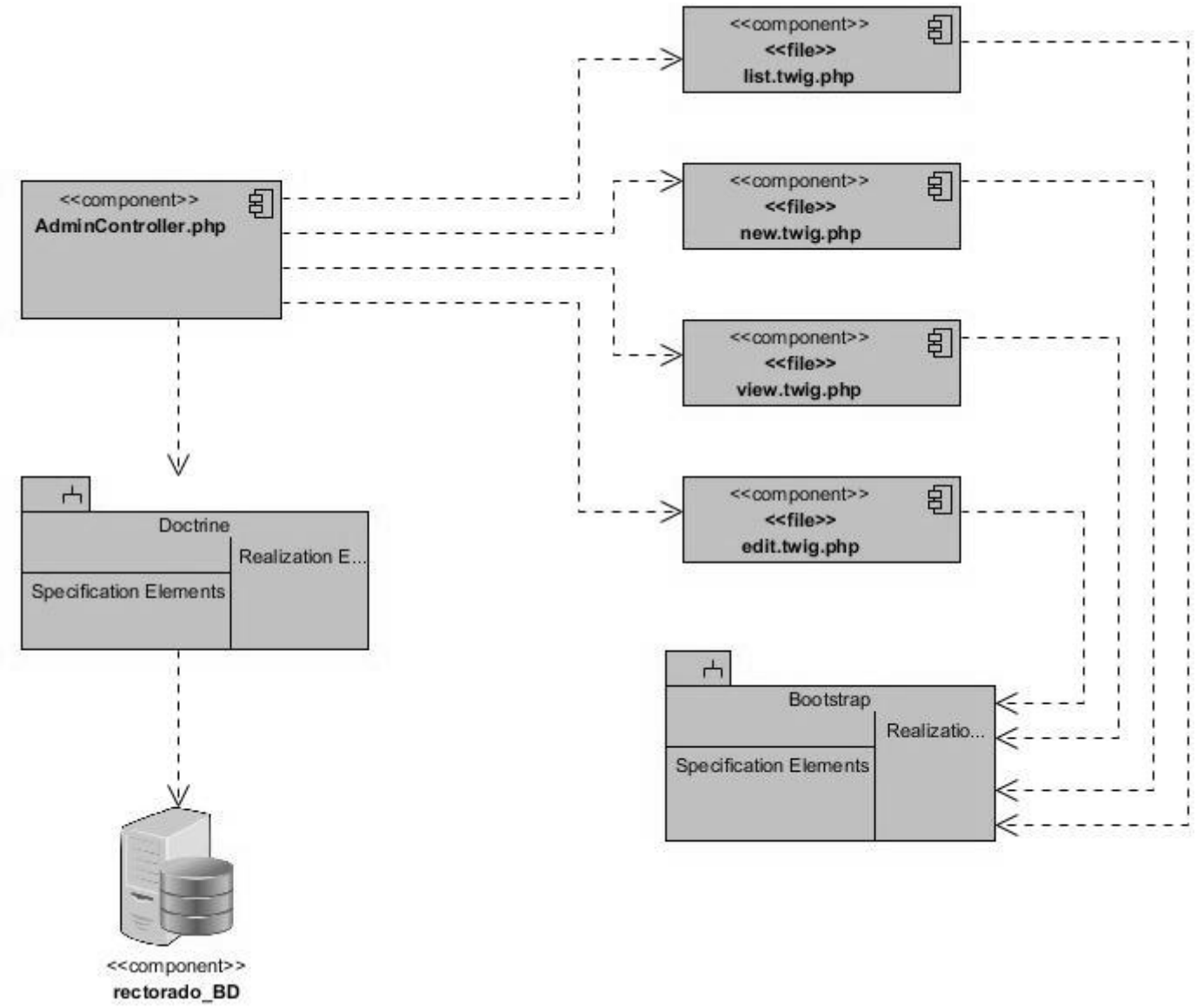


Fig. 7: Diagrama de componentes

3.5 Diagrama de despliegue

Los diagramas de despliegue sirven para visualizar el diseño arquitectónico permitiendo ver los aspectos físicos (computadoras, sistemas, dispositivos) que implementa un sistema mediante la representación de nodos, artefactos y las relaciones entre ellos. Es un elemento físico que existe en tiempo de ejecución y que representa un recurso computacional, que generalmente tiene algo de memoria y a menudo capacidad de procesamiento. Además se pueden organizar agrupándolos en paquetes y se pueden conectar entre sí, normalmente como asociaciones. También pueden tener atributos y operaciones y representan el empaquetamiento de elementos lógicos, bits, código (Banderas Contreras, y otros, 2011). Finalmente describe una topología del sistema.

En la Figura 5 se muestra el diagrama de despliegue del sistema implementado.

Ilustración 1: Diagrama de despliegue.

Descripción de los nodos

Nodo PC_Cliente: Ordenador desde el cual accedería el cliente al sistema.

Nodo Servidor de Aplicación Web: Ordenador donde se ejecutará el sistema.

Nodo Servidor Base Datos: Ordenador donde se almacenará la información del sistema.

3.6 Modelo de pruebas

Las pruebas son una actividad en la cual un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación (Jacobson, y otros, 2000).

En el flujo de trabajo prueba se verifica el resultado de la implementación, probando cada construcción. En las pruebas se definen los casos de prueba y los procedimientos de prueba. Un caso de prueba específica una forma de probar el sistema, incluyendo la entrada o resultado con la que se ha de probar y las condiciones bajo las que ha de probarse. Por su parte, un procedimiento de prueba, especifica cómo realizar uno o varios casos de prueba o partes de estos (Jacobson, y otros, 2000).

Toda prueba consta tradicionalmente de tres elementos: interacciones entre el sistema y la prueba, valores de prueba y resultados esperados. Los dos primeros elementos permiten realizar la prueba y el tercer elemento permite evaluar si la prueba se realizó con éxito o no. Un proceso de pruebas consta generalmente de tres fases: la fase de diseño de pruebas, la fase de ejecución y la fase de análisis de los resultados. En el ámbito de la Ingeniería de Software existen dos métodos fundamentales de pruebas: pruebas de caja blanca y pruebas de caja negra (Gutiérrez, y otros, 2006).

Dentro de las pruebas de funcionalidad se encuentra el método de **pruebas de caja negra**, conocidas también por pruebas de caja opaca, funcionales, de entrada/salida o inducidas por los datos. Se centran en lo que se espera del sistema, es decir, intentan encontrar casos en que el sistema no se atiene a su

especificación. Por ende el probador se limita a suministrarle datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo el sistema por dentro (Jacobson, y otros, 2000).

3.6.1 Pruebas de caja negra

Los casos de prueba de caja negra se enfocan en los requisitos funcionales del software. Así mismo permiten obtener conjuntos de condiciones de entrada que ejercitan completamente todos los requisitos funcionales del sistema. Además este tipo de prueba intenta encontrar errores que se clasifican en diferentes categorías como son: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación.

Para la aplicación de esta prueba se utiliza la técnica de partición equivalente la cual divide el campo de entrada en clases de datos de los cuales se pueden derivar casos de prueba (Pressman, 2010).
Un caso de prueba es un conjunto de acciones con resultados y salidas previstas basadas en los requisitos de especificación del sistema.
A continuación se describe el caso de prueba del Caso de Uso Gestionar Planteamiento, incluyendo los resultados obtenidos en su aplicación durante la última iteración de pruebas realizadas.

Descripción de variables del Caso de Prueba al caso de uso Gestionar Planteamientos

Tabla 3: Descripción de variables del Caso de Prueba

No.	Nombre del campo	Clasificación	Valor Nulo	Descripción
1	Nombre	Campo de texto	No	Debe introducirse caracteres alfabéticos y se deshabilita en caso de que se selecciona la opción “Anónimo”
2	Apellidos	Campo de texto	No	Debe introducirse caracteres alfabéticos y se deshabilita en caso de que se selecciona la opción “Anónimo”
3	Título	Campo de texto	No	Debe introducirse caracteres alfanuméricos.
4	Tiempo de expiración (días)	Campo de texto	No	Debe introducirse caracteres numéricos.
5	Fecha de recepción	Campo de selección	No	Debe seleccionarse un elemento
6	Fecha de respuesta	Campo de selección	Si	Debe seleccionarse un elemento
7	Clasificación	Campo de selección	No	Debe seleccionarse un elemento
8	Involucrados	Campo de selección	No	Debe seleccionarse un elemento

9	Anónimo	Campo de selección	Si	Se selecciona en caso de que el Planteamiento sea anónimo
10	Estado	Campo de selección	No	Debe seleccionarse un elemento
11	Satisfacción	Campo de selección	Si	Debe seleccionarse un elemento
12	Descripción	Campo de texto	Si	Debe introducirse caracteres alfanuméricos además de aceptar también caracteres especiales.
13	Evaluación	Campo de selección	No	Debe seleccionarse un elemento
14	Comisión Disciplinaria	Campo de selección y se activa en caso de que la opción seleccionada en el campo "Clasificación" sea "Queja o Denuncia"	Si	Debe seleccionarse un elemento
15	Resultados de Auditoría	Campo de selección y se activa en caso de que la opción seleccionada en el campo "Clasificación" sea "Queja o Denuncia"	Si	Debe seleccionarse un elemento
16	Medida Disciplinaria	Campo de selección y se activa en caso de que la opción seleccionada en el campo "Clasificación" sea "Queja o Denuncia"	Si	Debe seleccionarse un elemento

Tabla 4: Diagrama de Caso de Prueba para el CU: Gestionar Planteamiento

Nombre de la sección	Escenarios de la sección	Nombre	Apellidos	Título	Tiempo de expiración	Fecha de recepción	Fecha de respuesta	Clasificación	Involucrados	Anónimo	Estado	Satisfacción	Descripción	Evaluación	Comisión Disciplinaria	Resultados de Auditoría	Medida Disciplinaria	Respuesta del Sistema	Flujo Central
SC1:Adicionar Planteamiento.	EC1.1: Adicionar Planteamiento correctamente	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	V	Adiciona el Planteamiento de forma satisfactoria y lo guarda en la base de datos	El usuario después de haberse autenticado entra a la pantalla principal donde selecciona la opción Adicionar Planteamiento y posteriormente llena todos los campos correctamente y selecciona la opción Guardar cambios.
		Pedro	Rodríguez	Gestionar ómnibus para evento.	25	1/6/2016	N/A	Solicitud	Funcionarios	N/A	En proceso	N/A	Se necesita un ómnibus para evento x.	No resuelta	N/A	N/A	N/A		
	EC1.2:	V	V	I	V	V	V	V	V	V	V	V	V	V	V	V	V	El sistema	El usuario después

	Adicionar Planteamiento incorrectamente	Pedro	Rodríguez		30	2/6/2016	N/A	Queja	Entidad	N/A	No iniciado	N/A	N/A	No resuelta	N/A	N/A	N/A	notifica que existen campos vacíos.	de haberse autenticado entra a la pantalla principal donde selecciona la opción Adicionar Planteamiento y posteriormente comienza a llenar los campos pero deja alguno vacío y selecciona la opción Guardar cambios.
		V	V	V	I	V	V	V	V	V	V	V	V	V	V	V	V	El sistema notifica que existen campos numéricos (Tiempo de expiración (en días)) con entrada de datos inválidas	El usuario después de haberse autenticado entra a la pantalla principal donde selecciona la opción Adicionar Planteamiento y posteriormente comienza a llenar los campos pero introduce datos no válidos en un campo numérico y selecciona la opción Guardar cambios.
		Pedro	Rodríguez	Gestionar ómnibus para evento.	Treinta	5/6/2016	N/A	Sugerencia	Cuadros	N/A	En proceso	N/A	N/A	Respondida no procede	N/A	N/A	N/A		

Utilizando los casos de prueba diseñados se realizaron tres iteraciones de pruebas para encontrar la mayor cantidad de errores en el funcionamiento del sistema, las cuales fueron corregidas a medida que se fue avanzando en el proceso de prueba:

A continuación se muestra en la siguiente ilustración las No conformidades detectadas y su clasificación:

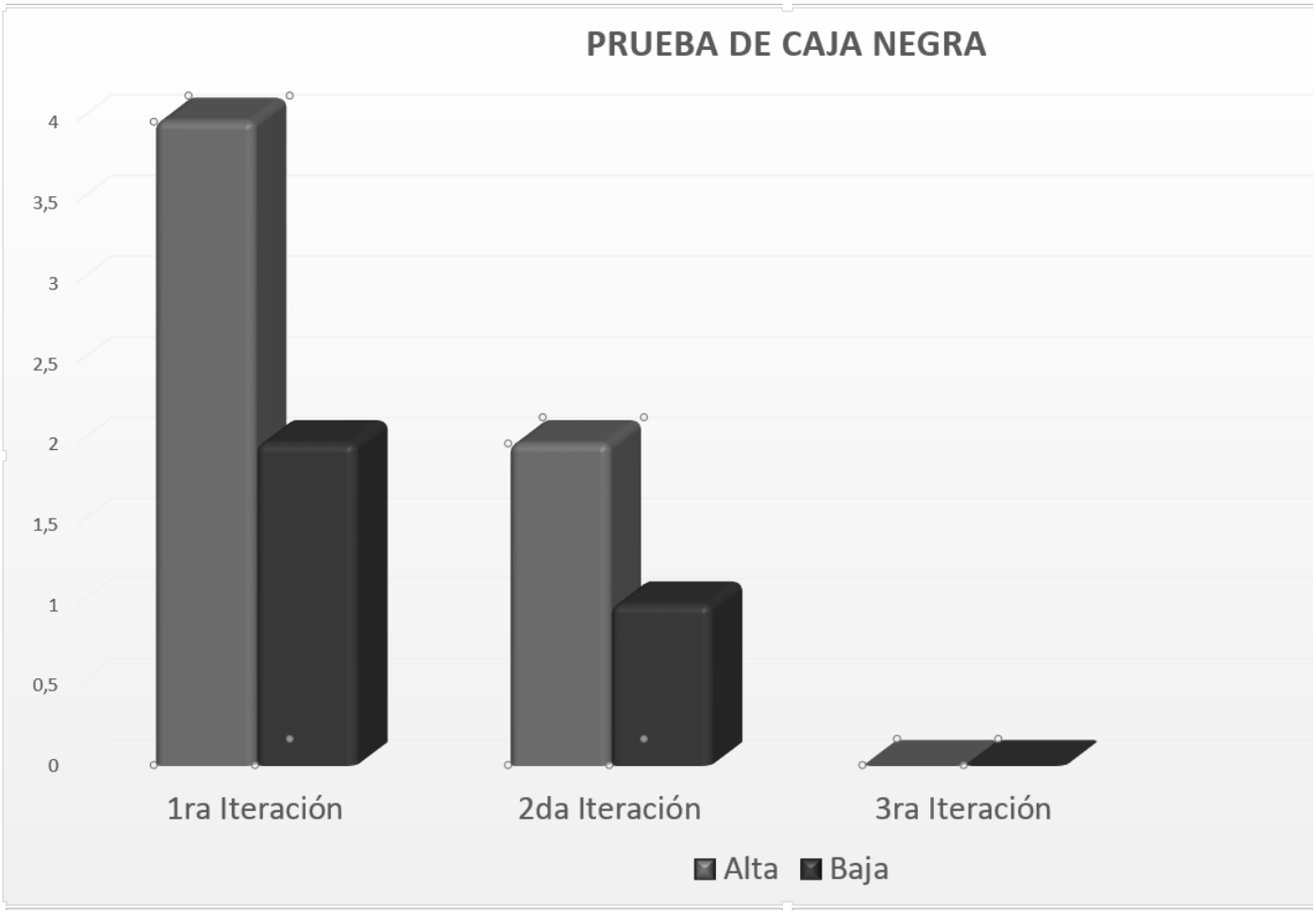


Fig. 8: Prueba de caja negra

En el "**Registro de defectos y dificultades detectados"** que a continuación se muestran ejemplos de las no conformidades detectadas durante las pruebas a la aplicación y la respuesta del equipo de desarrollo:

Tabla 5: Registro de defectos y dificultades detectados

Elemento	No.	No conformidad (NC)	Alta	Baja	Respuesta del equipo de desarrollo

Aplicación	1	Al dejar el campo título vacío el sistema no valida que el campo se encuentre vacío o no	X		En la clase Planteamiento.js se validó que el campo título no esté vacío.
Aplicación	2	En el formulario crear Planteamiento la palabra título no tenía tilde		X	Se corrigió la palabra poniéndole la tilde
Aplicación	3	Al dejar el campo clasificación vacío el sistema no valida que el campo se encuentre vacío o no	X		En la clase Planteamiento.js se validó que el campo clasificación no esté vacío.
Aplicación	4	Al dejar el campo involucrados vacío el sistema no valida que el campo se encuentre vacío o no	X		En la clase Planteamiento.js se validó que el campo involucrados no esté vacío.
Aplicación	5	Al dejar el campo tipo de caso vacío el sistema no valida que el campo se encuentre vacío o no	X		En la clase Planteamiento.js se validó que el campo tipo de caso no esté vacío.
Aplicación	6	En el formulario crear Planteamiento la palabra satisfacción estaba escrita incorrectamente		X	Se corrigió la palabra escribiéndola correctamente
Aplicación	7	En el formulario crear Planteamiento al introducir valores no	X		En la clase Planteamiento.js se validó que el

		numéricos en el campo tiempo de expiración el sistema no valida que el valor proporcionado sea incorrecto			campo tiempo de expiración solo admita valores numéricos
Aplicación	8	En el formulario crear Planteamiento al introducir valores no alfabéticos en el campo nombre el sistema no valida que el valor proporcionado sea incorrecto	X		En la clase Planteamiento.js se validó que el campo nombre solo admita valores alfabéticos
Aplicación	9	En el formulario crear Planteamiento la palabra área no tenía tilde		X	Se corrigió la palabra poniéndole la tilde

3.6.2 Prueba de validación del sistema.

Con el objetivo de verificar que el tiempo de creación de reportes disminuye haciendo uso del sistema desarrollado, se realizaron cuatro pruebas para comparar el tiempo de creación de reportes. Para la implementación de estas pruebas se tuvieron en cuenta dos escenarios de análisis, por lo que se describe un primer escenario (A) como la creación de los reportes de forma manual y contando solo con el apoyo de la herramienta Excel y un segundo escenario (B) como la creación de reportes haciendo uso del sistema propuesto.

En las dos primeras pruebas se crearon 7 reportes a partir de 5 planteamientos almacenados previamente: en el escenario (A) estos planteamientos estuvieron almacenados en documentos Word y en informes redactados en papel, y en el escenario (B) los planteamientos se almacenaron en el sistema desarrollado. En las pruebas 3 y 4 solamente se cambió el número de planteamientos analizados, aumentándose la cantidad de estos a 10, manteniéndose los mismos dos escenarios definidos previamente con el objetivo de ver el comportamiento de estos a mayor volumen de información a revisar.

En las cuatro pruebas realizadas, las salidas del proceso de creación de reportes fueron:

- Mediante el escenario (A): reportes obtenidos en formato Excel y luego convertidos a formato PDF para su almacenamiento final.
- Mediante el escenario (B): reportes obtenidos en formato PDF debido a que el sistema tiene incorporada esta funcionalidad.

Descripción de las pruebas:

Creación de reportes a partir de 5 planteamientos.

Prueba 1: Se utilizó el escenario (A) y se midió el tiempo en el que el usuario demoró en realizar los 7 reportes. Es necesario argumentar que para esta prueba el usuario, en la creación de cada reporte, tuvo que consultar cada uno de los planteamientos almacenados y a partir de estos extraer la información necesaria para llenar cada reporte.

Como resultado se obtuvo que el usuario demoró un tiempo de 20 minutos en realizar los reportes, debido a que el procesamiento de información se tuvo que hacer directamente por el propio usuario.

Prueba 2: Se utilizó el escenario (B) y se midió el tiempo en el que el sistema demoró en realizar los 7 reportes a partir de la interacción del usuario con el mismo. Para esta prueba el usuario solo tuvo que ingresar a la sección de cada reporte y especificar el período de tiempo en el que deseaba visualizar datos de los planteamientos, una vez seleccionados estos valores, se generaron los reportes y se exportaron a formato PDF mediante una funcionalidad del sistema.

Como resultado se obtuvo que el usuario con la utilización del sistema demoró un tiempo de 3 minutos en realizar los reportes debido a que el procesamiento de información fue realizado en su totalidad por el sistema.

Creación de reportes a partir de 10 planteamientos.

Prueba 3: Se utilizó el escenario (A) y se midió el tiempo en el que el usuario demoró en realizar los 7 reportes. Es necesario argumentar que para esta prueba el usuario, en la creación de cada reporte, tuvo que revisar cada uno de los planteamientos almacenados y a partir de estos extraer la información necesaria para llenar cada reporte.

Como resultado se obtuvo que el usuario demoró un tiempo de 45 minutos en realizar los reportes debido a que el procesamiento de información la tuvo que hacer el mismo usuario mencionado anteriormente y el volumen a analizar de información fue mayor que en las pruebas anteriores.

Prueba 4: Se utilizó el escenario (B) y se midió el tiempo en el que el sistema demoró en realizar los 7 reportes a partir de la interacción del usuario con el mismo. Para esta prueba el usuario solo tuvo que ingresar a la sección de cada reporte y especificar el período de tiempo en el que deseaba visualizar datos de los planteamientos, una vez seleccionados estos valores, se generaron los reportes y se exportaron a formato PDF mediante una funcionalidad del sistema.

Como resultado se obtuvo que el usuario con la utilización del sistema como apoyo, demoró un tiempo aproximado de 3,2 minutos.

Para tener una mejor visión de los resultados de las pruebas realizadas se expone la siguiente gráfica donde se aprecia el tiempo empleado en crear los 7 reportes haciendo uso de los escenarios (A) y (B). En una primera iteración para una cantidad de 5 planteamientos y en una segunda iteración para un total de 10 planteamientos.

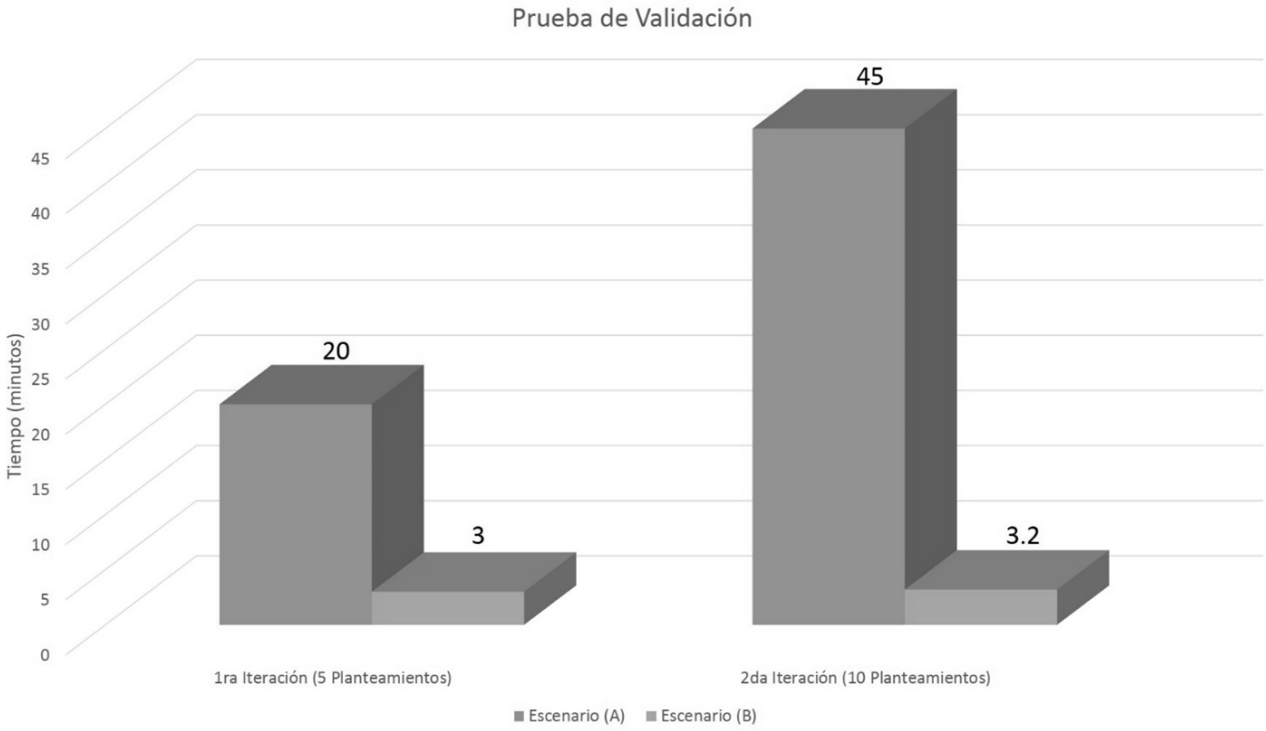


Fig. 9: Prueba de validación del sistema.

Al término de la realización de esta prueba se pudieron apreciar las ventajas que brinda la utilización del sistema desarrollado mediante el procesamiento de información y creación de reportes. Se demostró que la utilización del mismo disminuyó notablemente el tiempo empleado en procesar información para generar los reportes. Además se evidenció que la utilización del sistema para generar los reportes no logró solamente disminuir el tiempo empleado, sino también que el sistema, al realizar todo el análisis de la información contenida en los planteamientos, propició que el trabajo del usuario fuera menor al no tener que realizar este último los análisis correspondientes.

3.7 Conclusiones del capítulo

Durante el desarrollo de este capítulo se definieron los estándares de codificación lo que favoreció la reutilización y mantenimiento del código fuente del sistema desarrollado. Además se describieron los principales elementos que intervienen en la solución mediante el diagrama de componente lo que permitió establecer el alcance de las funcionalidades y las relaciones entre ellas. Por su parte, el diagrama de despliegue evidenció la relación necesaria entre el hardware y el software para beneficiar el entorno físico

con el cual contará el sistema. Una vez concluido el desarrollo del software se realizó la prueba de caja negra, usando la técnica de partición equivalente, esta prueba permitió hacerle una evaluación al sistema y corregir las no conformidades detectadas para contribuir con la calidad del software, también se realizó la prueba de validación, aportando la información necesaria para darle cumplimiento al objetivo general de la investigación.

CONCLUSIONES GENERALES

La investigación realizada arroja como principal resultado, el desarrollo de un Sistema gestión de información para la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas, a partir de lo cual se arriban a las siguientes conclusiones:

- El estudio realizado de los principales conceptos relacionados con el objeto de estudio de la investigación permitió elaborar el marco teórico de la misma.
- El estudio del estado del arte realizado demostró que no existe un sistema capaz de realizar la gestión de los procesos realizados en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas.
- El análisis de un conjunto de herramientas y tecnologías permitió definir el marco tecnológico en ajuste a las políticas de migración a software libre por la cual aboga el país.
- El proceso de desarrollo de la solución propuesta fue guiado por AUP-UCI, la metodología de desarrollo seleccionada, quedando documentada cada etapa del ciclo de vida. Los artefactos generados servirán como base de futuras actualizaciones del sistema de gestión.
- Se demuestra que el desarrollo de la aplicación permitió reducir el tiempo de procesamiento de la información y disminuir notoriamente las probabilidades de errores al aportar un valor agregado al proceso de ayuda a la toma de decisiones en la Oficina de Atención a la Población de la Universidad de las Ciencias Informáticas.

Recomendaciones

Luego de haber arribado a las conclusiones generales de esta investigación se recomienda generalizar el sistema desarrollado en todas las universidades o sedes del país.

Referencias Bibliográficas

Abreu , José Luis y Calzada Cantú, Leticia Mayela. 2009. *The impact of business intelligence tools in executive business.* México : International Journal of Good Conscience, 2009.

Alvarez, Rubén. 2001. desarrolloweb.com. [En línea] 2001. [Citado el: 10 de 12 de 2015.] http://www.desarrolloweb.com.

Arenas Cardona, Paul Andrés. 2011. Correcciones y modificaciones realizadas al SIAF "sistema de información administrativa y financiera". [En línea] 2011. [Citado el: 28 de 3 de 2016.] http://ribuc.ucp.edu.co:8080/jspui/bitstream/handle/10785/111/completo.pdf?sequence=1.

Banderas Contreras, Tomás y Gómez Gil, Pilar. 2011. Instituto Nacional de Astrofísica, Óptica y Electrónica. [En línea] 2011. [Citado el: 15 de 3 de 2016.] https://ccc.inaoep.mx/~pgomez/cursos/ingsw/acetatos/UML%202_0_Tutorial.pdf.

Bass, Len, Clements, Paul y Kazman, Rick. 1998. *Software Architecture Practice.* 1998.

Bermudez, Mauricio Orlando, Vargas Izquierdo, Angela Luisa y Suarez Rivera, Carol Johana. 2010. *Sistema de información para el archivo histórico del Departamento Nacional de Planeación DNP.* s.l. : Corporación Universitaria Minuto de Dios, 2010.

Blanco, Kenia Riverón Ovalle y Yaidel Rodríguez. 2009. *IMPLEMENTACIÓN DEL MÓDULO DESPACHO DEL SUBSISTEMA INVENTARIO DEL SISTEMA INTEGRAL DE GESTIÓN CEDRUX.* Habana : s.n., 2009.

Castellanos Guevara, Julio César , Lemos García, Jairo Alonso y Solarte Martínez, Guillermo Roberto. 2012. *Sistema Integrado de Gestión de Servicios para Áreas de Salud en la Institución Educativa Magdalena Ortega de la Unión V.* s.l. : Scientia et Technica, 2012.

Cembranos, Fernando y Medina, Jose Angel. 2003. *GRUPOS INTELIGENTES: TEORIA Y PRACTICA DEL TRABAJO EN EQUIPO.* 2003.

CMS. 2008. SELECTING A DEVELOPMENT APPROACH. [En línea] 2008. [Citado el: 3 de 12 de 2015.] https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf.

CODEBOX. 2010. CodeBox.es. [En línea] 2010. [Citado el: 10 de 12 de 2015.] www.codebox.es/glosario.

ConML. 2012. ConML. [En línea] 2012. [Citado el: 06 de 12 de 1015.] http://www.conml.org/FAQ.aspx.

Departamento de Ciencias de la Computación en Inteligencia Artificial, Universidad de Alicante. 2011. Departamento de Ciencias de la Computación en Inteligencia Artificial, Universidad de Alicante. [En línea] 2011. [Citado el: 10 de 12 de 2015.] http://www.jtech.ua.es/dadm/2011-2012/restringido/web/sesion02-apuntes.html.

DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA. 2015. ALEGSA.com.ar. [En línea] 2015. [Citado el: 22 de 12 de 2015.] http://www.alegsa.com.ar/Dic/servidor de base de datos.php.

Editboard.com. 2008. catedraprogramacion. [En línea] 2008. [Citado el: 10 de 12 de 2015.]

Flexible information technology. 2016. FLEXIT. [En línea] 2016. [Citado el: 21 de 12 de 2015.] http://www.flexit.net/sp/developers.

Gamma, Erich, y otros. 1998. Design Patterns: Elements of Reusable ObjectOriented Software. [En línea] 1998. [Citado el: 10 de 1 de 2016.] http://www.weibnc.com/wp-content/uploads/brkpdfs/Design-Patterns-Elements-of-ReusableObject-Oriented-Software-by-John-M.-VlissidesFoundational.pdf.

GENBETA. 2012. GEMBETA:dev. [En línea] 2012. [Citado el: 1 de 02 de 2015.] http://www.genbetadev.com/frameworks/bootstrap.

GrandRavine Software Limited. 2007. GrandRavine Software Limited. [En línea] 2007. [Citado el: 3 de 12 de 2015.] http://www.maintscape.com/documents/MaintScape%20Training.pdf.

Guía de Ubuntu. 2010 . GUÍA DOCUMENTADA PARA UBUNTU. [En línea] 2010 . [Citado el: 23 de 12 de 2015.] http://www.guia-ubuntu.com/index.php?title=PgAdmin_III.

Gutiérrez, Javier J, y otros. 2006. MODELOS DE PRUEBAS PARA PRUEBAS DEL SISTEMA. [En línea] 2006. [Citado el: 10 de 3 de 2016.] http://ceur-ws.org/Vol-227/paper07.pdf.

indracompany.com. 2012. indracompany.com. [En línea] 2012. [Citado el: 3 de 12 de 2015.] http://www.indracompany.com/sites/default/files/presentacion_isocloud_modo_de_compatibilidad.pdf.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : s.n., 2000.

JetBrains. 2015. PhpStorm. [En línea] 2015. [Citado el: 23 de 12 de 2015.] https://www.jetbrains.com/phpstorm/.

Juan D Castellanos_Fundación Universitaria. 2008. Herramientas CASE para ingeniería de requisitos. [En línea] 2008. [Citado el: 06 de 12 de 2015.] http://www.revistasjdc.com/main/index.php/ccient/article/view/37/36.

Lara, MSc. Lourdes Portela. 2006. Los Sistemas de Gestión de Información, piedra angular de la Estrategia integral de gerencia. [En línea] 2006. [Citado el: 1 de 12 de 2015.] www.bibliociencias.cu/gsdl/collect/eventos/tmp/LosSistemasdeGestiondeInformacion.html.

Letechi, Luque y Leopoldo, Alex. 2006. Repositorio Dspace. [En línea] 2006. [Citado el: 28 de 3 de 2016.] http://www.dspace.espol.edu.ec/handle/123456789/12243.

Liberum. 2010. Liberum Help Desk. [En línea] 2010. [Citado el: 3 de 12 de 2015.] archive-org.com/page/582136/2012-11-04/http://www.liberum.org/.

Medel Viltres, Yamira y Guerrero Grey, Leodanys Wilber. 2014. *Proceso de mejora del Sistema de Gestión de Proyectos para Cuba y Venezuela.* 2014. 2255-1514.

Microsoft. 2016. [En línea] 2016. [Citado el: 28 de 3 de 2016.] https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

Mozilla Developer Netwok. 2015. MDN. [En línea] 2015. [Citado el: 10 de 12 de 2015.] https://developer.mozilla.org/es/docs/Web/JavaScript.

OMG (Object Management Group). 2011. Unified Modeling Language™ (UML®). [En línea] 2011. [Citado el: 06 de 12 de 2015.] http://www.uml.org/.

Patrón Modelo-Vista-Controlador. **Díaz González, Yanette y Fernández Romero, Yenisleidy. 2012.** 1, s.l. : http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15, 2012, Vol. 11.

Pérez Rodríguez, Yudith y Coutín Domínguez, Adrián. 2005. *La gestión del conocimiento: un nuevo enfoque en la gestión empresarial.* La Habana : s.n., 2005.

Pérez, Fidel. 2005. La entrevista como técnica de investigación social. Fundamentos teóricos, técnicos y metodológicos. [En línea] 2005. [Citado el: 16 de 6 de 2016.] http://datateca.unad.edu.co/contenidos/401560/La_entrevista_como_tecnica_de_investigacion_social_Fundamentos_teoricos.pdf.

Ponjuán Dante, Gloria y Fernández Valdés, María de las Mercedes. 2008. ACIMED. [En línea] 2008. [Citado el: 06 de 12 de 2015.] http://scielo.sld.cu/scielo.php?pid=S1024-94352008000700007&script=sci_arttext&tlng=pt.

PostgreSQL. 2015. PostgreSQL 9.3 Documentation. [En línea] 2015. [Citado el: 22 de 12 de 2015.] http://www.postgresql.org/docs/9.1/interactive/intro-whatis.html.

Potencier, Fabien y Weaver, Ryan. 2009. *Symfony 2.* 2009.

Potencier, Fabien y Zaninotto, François. 2016. Librosweb. [En línea] 2016. [Citado el: 15 de 6 de 2016.] http://librosweb.es/libro/symfony_1_2/capitulo_2/el_patron_mvc.html.

Pressman. 2010. *Ingeniería del Software, un enfoque práctico.* Madrid : s.n., 2010.

Ruiz, Fernando. 2008. [En línea] 2008. [Citado el: 2015 de 12 de 10.] http://www.ferruiz.com/wp-content/uploads/2008/04/sym.pdf.

Sánchez, M, R P y Baños, V, Y. 2007. *Sistema generador de Mapas Temáticos y Gráficos.* 2007.

Sommerville, Ian. 2009. *Ingeniería de Software. Madrid: Pearson Educación.* 2009.

Soto, Lauro. 2010. TECNOLÓGICO NACIONAL DE MÉXICO. [En línea] 2010. [Citado el: 1 de 12 de 2015.] http://www.mitecnologico.com/.

Stair, Ralph y Reynolds, George. 2010. *Principios de sistemas de información.* México, D.F. : Cengage Learning Editores, 2010.

Symfony. 2011. symfony. [En línea] 2011. [Citado el: 10 de 12 de 2015.] http://symfony.com.

The Apache Software Fundation. 1997. Apache. [En línea] 1997. [Citado el: 4 de 12 de 2015.] http://www.apache.org/.

The jQuery Foundation. 2016. jQuery. [En línea] 2016. [Citado el: 9 de 12 de 2015.] https://jquery.com.

Universidad Politécnica de Valencia, Departamento de Sistemas Informáticos y Computación. 2013. [En línea] 2013. [Citado el: 3 de 12 de 2015.] https://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjluvmUy6_NAhWC2R4KHQ2ODk4QFggaMAA&url=http%3A%2F%2Fwww.dsic.upv.es%2Fasignaturas%2Ffacultad%2Flsi%2Fdoc%2FIntroduccionProcesoSW.doc&usg=AFQjCNGAyOzvxjChM6izsIaI_eLoQhtk2Q&sig2=C.

Valle, Guillermo Rafel Pagan Diaz del. 2015. engicode. [En línea] 2015. [Citado el: 22 de 12 de 2015.] http://engicode.com/que-es-un-servidor-web/.

Visual Paradigm. 2015. Visual Paradigm. What VP - UML Provides. [En línea] 2015. [Citado el: noviembre de 20 de 2015.] http://www.visual-paradigm.com/product/vpuml/provides.

W3C. 2010. World Wide Web Consortium. [En línea] 2010. [Citado el: 9 de 12 de 2015.] http://www.w3c.es/Divulgacion/a-z/.