Sistema para el registro y control de los procesos de la vida interna del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas

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

Autores:

Lorena García Lima

Isidro Montero Reyes

Tutores:

Ing. Alexander Rodríguez Mompié

Ing. Yasmany Tellez Collazo

Ing. Nayilet Martín Soler

Ing. Yarelis González Collado

La Habana, Cuba

Junio, 2015

Declaración de autoría

Declaración de autoría

Declaramos ser autores de la presente tesis que tiene por título: Sistema para el registro y control de los procesos de la vida interna del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmamos la presente tesis a los __ días del mes de Junio del año 2015.

Lorena García Lima

_____

Firma de la autora

Isidro Montero Reyes

_____

Firma del autor

Ing. Alexander Rodríguez Mompié

_____

Firma del tutor

Ing. Yasmany Tellez Collazo

_____

Firma del tutor

Ing. Nayilet Martin Soler

_____

Firma de la tutora

Ing. Yarelis González Collado

_____

Firma de la tutora

Agradecimientos

Agradecimientos

Agradezco a mis padres y mi abuelo por haberme brindado su apoyo y confianza en todo momento.

Gracias a todos los miembros de mi familia por su preocupación y ayuda.

A mis amigos y compañeros, gracias por su amistad durante estos 5 años, por apoyarme en los malos momentos y llenar de risas los buenos.

A mis tutores, gracias por su apoyo en el desarrollo del trabajo, por su paciencia, sus consejos y todo el tiempo que dedicaron en ayudarnos.

A todos los miembros de Núcleo por ayudarnos en el desarrollo del sistema y por atendernos y responder todas nuestras dudas en todo momento.

A todos los que de una forma u otra han aportado algo en la realización de este trabajo. Gracias

Lorena García Lima

A mis padres, que son parte esencial e incondicional de cada meta que me propongo y donde siempre encuentro ayuda oportuna, consejo sabio y sobre todo amor infinito.

A mi familia, mi primera escuela e inquebrantable fuente de energías y dicha.

A mis amigos, que con naturalidad y humildad me brindan ese imprescindible tesoro del cual siempre me nutro.

A mis maestros; educar solo puede quien sea un evangelio vivo.

A mis profesores, que han compartido desinteresadamente tantos valiosos conocimientos.

A Fidel, grande y único.

A mi Cuba.

A todos, los que de una forma u otra han contribuido a forjarme tal cual a este modesto resultado arribo, mi más sincera gratitud.

Isidro Montero Reyes

Dedicatoria

A mis padres por siempre impulsarme a estudiar más y superarme.

A mi abuelo Juan por apoyarme incondicionalmente desde pequeña en todas las ideas que se me ocurrían y de tener la paciencia de siempre tratar de entenderme.

Lorena García Lima

A los estudiantes, profesores y trabajadores de la Universidad de las Ciencias Informáticas.

A mis padres.

A mi gran amor: Cuba.

Isidro Montero Reyes

Resumen

Resumen

La presente investigación tiene como objetivo el desarrollo de un sistema que permita agilizar y centralizar el registro y control de los procesos de la vida interna de la militancia del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas integrado al Sistema de Gestión Universitaria. Esta organización maneja constantemente grandes volúmenes de información para su funcionamiento, lo que en ocasiones dificulta la consulta inmediata y la actualización de la misma, así como la emisión de reportes. El Sistema de Gestión de Organizaciones gestiona la estructura y composición de la organización, los militantes con sus respectivas responsabilidades en las estructuras, la realización de trámites como son los ingresos, las incorporaciones, los traslados, las desactivaciones y las sanciones, así como el control del pago de la cotización y el registro de los elementos significativos de las actas de reuniones. Además permite la búsqueda y ubicación de expedientes físicos de forma rápida y exacta. Para el desarrollo de la aplicación se utilizó la metodología de Desarrollo Ágil con Calidad y el marco de trabajo GUUD definido por el Departamento de Desarrollo de la Dirección de Informatización. El sistema facilita las búsquedas de información para la confección de reportes que con frecuencia son solicitados al PCC, brinda una manera viable de realizar consultas para obtener la información que se necesita durante cualquier proceso de la vida interna de la organización y permite consultar el estado de los trámites realizados a los militantes.

Palabras clave: organizaciones, Partido Comunista de Cuba, procesos de la vida interna.

Índice

Índice

Índice

Índice

Índice de figuras

Índice

Índice de tablas

Introducción

Introducción

El vertiginoso desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) crea las bases para la construcción de un nuevo entorno social en el que surgen múltiples posibilidades de nuevos servicios y facilidades para la vida diaria de la humanidad, tales como: correo electrónico, redes compartidas, acceso a bases de datos globales, así como rápidos y eficientes sistemas de comunicaciones. Estos cambios obligan a tomar medidas y realizar ajustes estratégicos en las organizaciones, para adaptarse a esta nueva era tecnológica (MIRANDA ESCALONA, 2004). Los diversos tipos de organizaciones contemporáneas tanto políticas, como de masas o sociales a nivel mundial que aspiren al éxito, requieren de sistemas que permitan la gestión de la información para ganar en eficiencia y mejorar la toma de decisiones.

El contexto nacional no se ve exento de la informatización en las organizaciones y la necesidad de perfeccionar el trabajo con la información que en estas se genera. La informatización que se aplica en Cuba desde hace varios años demuestra la voluntad política del país por acercar cada vez más las nuevas tecnologías a la población, lo que está refrendado en los Lineamientos de la Política Económica y Social del Partido y la Revolución, que rigen las transformaciones en curso y parten de que no es posible una sociedad próspera y sostenible sin subordinar a tales objetivos las herramientas que garanticen el acceso al conocimiento, la eficiencia, la productividad y la excelencia (GRANMA, 2014).

Es por ello que las diferentes organizaciones políticas y de masas, como el Partido Comunista de Cuba (PCC), requieren adaptar las amplias posibilidades que sustenta el uso adecuado de las TIC a sus necesidades como organización pues se requiere de un estricto manejo de sus datos por ser de gran relevancia para el país.

La Universidad de las Ciencias Informáticas (UCI) se especializa en el desarrollo de soluciones informáticas, que contribuyen tanto a la informatización del país como de la propia universidad. Una de estas soluciones es el Sistema de Gestión Universitaria (SGU), que abarca las áreas de procesos sustantivos de la universidad como docencia, investigaciones y extensión. Este sistema almacena los datos primarios de las personas de la institución y se encuentra al alcance de todos los miembros de la institución. Sin embargo, a pesar de sus grandes potencialidades, dicho sistema, hasta la fecha, no brinda la posibilidad de gestionar la información relativa a los procesos de las organizaciones políticas y de masas.

Introducción

Los procesos de la vida interna del PCC constituyen una de las principales tareas que se deben informatizar. En la UCI, pese al acceso mayoritario a las nuevas tecnologías, solo se ha logrado vincularlas al trabajo en el PCC con la entrega de la información en formato digital y el uso del correo electrónico para la comunicación. Los métodos para el procesamiento y la gestión de la información son los tradicionales pues las personas encargadas de revisarla, analizarla y elaborar los reportes aún lo realizan manualmente.

Actualmente el PCC en la UCI está integrado por más de 600 militantes, distribuidos en 44 núcleos, cinco comités primarios y un Comité PCC-UCI de los cuales es necesario archivar por cada uno, un expediente con toda la información del militante. Los expedientes se almacenan en archivos, donde la ubicación y búsqueda se realiza de forma manual a partir de un listado con las capacidades libres de los archivos, dando margen a errores en el almacenamiento y asignación de los números de expediente.

La gestión de la plantilla de la militancia de la organización con sus respectivas responsabilidades, los trámites de ingreso, incorporación, traslado, desactivación y sanción, así como el registro de los elementos significativos de las actas de reuniones se realizan haciendo uso de herramientas ofimáticas, las cuales contribuyen a facilitar el trabajo pero no están diseñadas para tener en cuenta las reglas del negocio de la vida interna del PCC.

Cada tres meses, se realiza para los núcleos, comités primarios y el Comité del Partido UCI una reunión para discutir y aprobar el plan de trabajo del trimestre y comprobar el cumplimiento de las actividades del trimestre anterior. Muchas veces no existe correspondencia entre las actividades planeadas por el comité y las realizadas en las organizaciones base o núcleos pues luego de aprobado el plan del trimestre no se actualiza con las actividades que surgen a partir de los trámites y los acuerdos tomados en las reuniones ordinarias.

Mensualmente cada estructura se reúne al menos una vez generando un acta, que se entrega de manera digital por correo electrónico al Comité del Partido UCI donde son procesadas de forma manual para extraer los datos generales, entre los que se pueden encontrar: asistencia, cantidad de acuerdos tomados, temas discutidos, entre otros; para generar un resumen estadístico que posibilite un mejor análisis. Estos resúmenes se elaboran luego de analizar las actas de cada uno de los núcleos, lo cual consume mucho tiempo de elaboración y al ser tan extensos, pueden contener errores.

La permanencia de un militante en un núcleo se registra en la Plantilla de Control de la Militancia. Cada trámite que se realice a un militante por el organismo superior, se debe informar a las direcciones de los

núcleos implicados en esos trámites para actualizar los datos de este. Esta función se realiza a través del correo electrónico, por teléfono o personalmente, lo que conlleva en ocasiones a la desincronización entre la información real y la registrada en la documentación digital.

Además de los procesos anteriores, la organización requiere del pago mensual de la cotización por parte de sus miembros y resulta complejo llevar un control de los salarios mensuales y de la cuota a cotizar, puesto que la información que se maneja en plantilla no siempre es la que aparece en los listados de cotización.

Teniendo en cuanta la situación problemática planteada se define el siguiente **problema de investigación:** ¿Cómo agilizar y centralizar el registro y control de los procesos de la vida interna de la militancia del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas?

Se plantea como **objeto de estudio** la gestión de los procesos de las organizaciones políticas y de masas. El **campo de acción** se centra en la gestión de los procesos de registro y control de la vida interna de la militancia del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas.

Para resolver el problema enunciado se propone como **objetivo general** de la investigación**:** desarrollar un sistema para el registro y control de los procesos de la vida interna de la militancia del Partido Comunista de Cuba con tecnologías libres en la Universidad de las Ciencias Informáticas que permita agilizar y centralizar la gestión de los procesos.

De acuerdo con la propuesta anterior se trazan los siguientes **objetivos específicos**:

- Caracterizar los fundamentos teóricos-metodológicos para el desarrollo de la propuesta de solución.
- Caracterizar el registro y control de los procesos de la vida interna de la militancia del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas.
- Implementar el sistema para el registro y control de los procesos de la vida interna de la militancia del Partido Comunista en la Universidad de las Ciencias Informáticas.
- Evaluar la propuesta de solución mediante la realización de pruebas de software.

La investigación se sustenta en la siguiente **idea a defender**: el Sistema de Gestión de Organizaciones contribuirá a agilizar y centralizar los procesos de la vida interna de la militancia del Partido Comunista de Cuba en la Universidad de las Ciencias Informáticas.

Introducción

Para dar cumplimiento a los objetivos específicos de la investigación se plantean las siguientes **tareas de investigación:**

- Análisis del estado del arte de los diferentes sistemas que informaticen la gestión de los procesos de organizaciones políticas y de masas.
- Definición de los conceptos relacionados con el marco teórico de la investigación.
- Obtención de los requisitos funcionales y no funcionales de la propuesta de solución.
- Caracterización de los elementos que componen la arquitectura de la propuesta de solución y las tecnologías necesarias para su desarrollo.
- Modelado de la base de datos que permita establecer el modelo lógico y físico de la estructura de datos asociada a la solución.
- Implementación de las funcionalidades definidas para la solución propuesta.
- Realización de las pruebas de software para evaluar la solución.

Para el desarrollo de la presente investigación, se hace necesario la utilización de los siguientes métodos científicos.

Métodos teóricos:

- **Histórico-lógico:** se evidencia en el estudio de los temas relacionados con el desarrollo de aplicaciones que gestionan los procesos de las organizaciones políticas y de masas y analizando los resultados del estudio se llega a conclusiones lógicas que contribuyen al desarrollo y cumplimiento del objetivo de la investigación.
- **Analítico-sintético:** se evidencia en el análisis de los elementos de la situación problemática relacionándolos entre sí y vinculándolos como un todo para descubrir relaciones y características generales entre los elementos de la realidad. Se emplea para analizar la información encontrada sobre los elementos que intervienen en la gestión de la información, específicamente del PCC, y extraer ejemplos generales y específicos, tratando de establecer relación entre los diferentes criterios que puedan incidir y que tributen a la correcta gestión de la información.

Métodos empíricos:

- **Análisis documental:** se evidencia en el estudio de los documentos que establecen procedimientos para la gestión de los procesos de la vida interna del Partido Comunista de Cuba como son los reglamentos y estatutos de esta organización.

- **Entrevista:** se entrevistó al Secretario del Comité del Partido UCI encargado del trabajo relacionado con la esfera de la vida interna del PCC con el fin de obtener toda la información necesaria sobre el funcionamiento de la organización e identificar los requisitos funcionales del sistema. Las entrevistas realizadas se encuentran documentadas en el **Anexo 1.**

Justificación de la investigación:

Como resultado de la investigación se espera obtener el Sistema de Gestión de Organizaciones que permitirá agilizar el registro y control de los procesos de la vida interna del Partido Comunista de Cuba en la UCI a través de una aplicación web, centralizando la información dentro del Sistema de Gestión Universitaria. Esto contribuirá a que los usuarios accedan fácilmente a la información de la organización tributando a un mejor aprovechamiento del tiempo de trabajo. Además permitirá mejorar el acceso a la información manejada referente a esta organización.

Estructura del documento

El trabajo está estructurado en tres capítulos que abarcan todo el proceso para el desarrollo de la solución informática.

Capítulo 1: Fundamentación teórica: se expone el estado del arte del objeto de estudio de la investigación y se definen conceptos fundamentales asociados al dominio del problema. Además se analizan varias herramientas para la gestión de procesos de organizaciones políticas y de masas y se caracteriza el entorno tecnológico para el desarrollo de la propuesta de solución.

Capítulo 2: Descripción de la propuesta de solución: se describe la propuesta de solución, sus requisitos funcionales y no funcionales, así como los procesos y actores que intervienen en el registro y control de los procesos de la vida interna del PCC. También se describen los patrones de diseño y el patrón de arquitectura utilizados por el marco de trabajo, así como el modelo de datos definido para el registro de la información a gestionar.

Capítulo 3: Implementación y evaluación de la solución: se abordan las fases implementación y despliegue de la propuesta de solución teniendo en cuenta las técnicas de codificación empleadas, así como la realización de pruebas para evaluar y verificar la calidad del sistema, tales como las pruebas de integración, de sistema (funcionales, rendimiento y resistencia) y de aceptación para satisfacer las necesidades del cliente.

Capítulo 1: Fundamentación teórica

1.1. Introducción

En el presente capítulo se estudian diversos sistemas informáticos que gestionan los procesos de las organizaciones políticas y de masas, tomando elementos sobresalientes que puedan ser utilizados para dar solución al problema planteado. Además se caracterizan las tecnologías utilizadas para el desarrollo del Sistema de Gestión de Organizaciones dentro del Sistema de Gestión Universitaria, así como los conceptos y elementos teóricos que sustentan la presente investigación.

1.2. Conceptos asociados al dominio del problema

Para lograr un mejor entendimiento del tema se considera imprescindible definir los siguientes conceptos que giran alrededor del objeto de estudio y del campo de acción:

Organización: asociación de personas regulada por un conjunto de normas en función de determinados fines (RAE, 2015).

Partido Comunista de Cuba (PCC): partido único, mantiene una labor sistemática y tenaz por el desarrollo y consolidación en nuestra sociedad de la ideología de la Revolución Cubana, que resume e integra lo específico de nuestra Revolución (PCC, 1998).

Comité PCC: estructura directiva principal que dirige a los comités primarios y núcleos que se subordinan a esta. Son constituidos para agrupar y organizar el trabajo de la organización en los centros de trabajo, de estudio o unidades militares (PCC, 2013).

Comité primario: órgano de dirección que dirige dos o más núcleos y se subordina al Comité PCC (PCC, 2013). De acuerdo con su complejidad, las actividades que desarrolla y el número de militantes, adopta diferentes estructuras en la que el núcleo es su fundamento (PCC, 1998).

Núcleo: organización de base de la estructura partidista, que actúa en el centro de trabajo, unidad militar o en la comunidad, donde existan, como mínimo, tres militantes (PCC, 2013). El núcleo constituye el vínculo indisoluble de la vanguardia con los trabajadores y el pueblo en general y lleva a la práctica la política del partido en el lugar donde actúa (PCC, 1998).

Militante: es militante del PCC el ciudadano cubano que se identifica con su política, trazada por los órganos y organismos superiores y acepta sus estatutos, pertenece a uno de sus núcleos, actúa en él, en uno de sus organismos o en ambos, abona la cuota establecida, cumple las decisiones y acuerdos del partido, lucha y trabaja por llevar adelante la construcción del socialismo (PCC, 1998).

Secretario General: el Secretario General es el representante máximo del núcleo del partido, preside sus reuniones, controla de forma cotidiana las actividades partidistas de los militantes, adopta decisiones en el marco de sus atribuciones para enfrentar las situaciones diarias que se presenten en su radio de acción informándolas posteriormente al núcleo cuando ello sea necesario y responde ante el partido por el adecuado cumplimiento de las tareas organizativas y político-ideológicas de la organización de base (PCC, 2013).

Sanción: medida disciplinaria aplicada a un militante por infringir o incumplir el reglamento establecido por la organización (PCC, 2013). Al militante del partido le pueden ser aplicadas las sanciones de amonestación, separación del cargo, suspensión temporal de derechos del militante, separación de las filas del partido y la expulsión (PCC, 1998).

Crecimiento o ingreso: proceso de admisión en el partido. El ingreso al PCC es voluntario, se realiza mediante las asambleas de elección de trabajadores ejemplares y por selección individual (PCC, 1998).

Desactivación: la aplicación de las desactivaciones en el partido posibilita que las organizaciones de base y las organizaciones del partido resuelvan los casos de militantes que no desarrollan correctamente sus funciones y han disminuido o perdido las cualidades que les permitieron recibir la condición de militantes del partido (PCC, 2013).

Doble militancia: al ser admitidos en el partido los miembros de la Unión de Jóvenes Comunistas (UJC) que no hayan arribado a la edad límite en la organización juvenil pueden continuar militando en ella; aquellos otros que ingresen al partido al arribar a la edad máxima en la organización dejan de militar en esta, salvo casos excepcionales (PCC, 1998).

Cotización: la cotización de los militantes es un principio organizativo del partido. La cuota a pagar cada mes se determina según las normas aprobadas por el secretariado del Comité Central (PCC, 2013).

Reunión: el núcleo se reúne, como regla, una vez al mes y cuantas veces sea necesario, convocado por su dirección o por un organismo superior. Sus acuerdos son de obligatorio cumplimiento para sus integrantes (PCC, 2013).

Traslado: cuando un militante se mueve de un núcleo a otro. Los traslados de los militantes pueden ser internos o externos (PCC, 2013).

Incorporación: cuando un militante se traslada desde algún núcleo, desde otro municipio, desde otra provincia o desde el extranjero y se incorpora al núcleo final (PCC, 2013).

1.3. Análisis de sistemas homólogos

Debido a los avances tecnológicos en el desarrollo de las actividades cotidianas, las organizaciones políticas y de masas necesitan aprovechar los beneficios que brinda la tecnología para agilizar sus procesos.

En tal sentido esta investigación busca dar respuesta a los requisitos presentados por el PCC a través de un sistema que permita agilizar y centralizar el registro y control de la información referente a los procesos de la vida interna de la militancia de la organización. Entre las condiciones o características necesarias para obtener una solución completa se definieron los siguientes puntos:

- La solución debe poder integrarse al SGU para obtener los datos primarios de las personas de la universidad que sean militantes de la organización.
- Debe abarcar los procesos de la vida interna del PCC: traslado, incorporación, crecimiento, desactivación, sanciones, cotización, registro de actas y plan de trabajo.
- Debe poder utilizarse en todas las áreas administrativas de la UCI.

Para conocer las tendencias actuales y elementos comunes e identificativos en los sistemas dedicados a la gestión de organizaciones, se realizó un estudio que permitió determinar características posibles a incluir en la propuesta de solución y comprobar la inexistencia de un sistema que cumpla con las condiciones antes mencionadas.

1.3.1. Sistemas para la gestión de organizaciones políticas y de masas

- ***Sistema de Información Estadística del PCC (INFOEST)***

INFOEST, sistema desarrollado en el 2011 por el Comité Central del PCC con el objetivo de automatizar la gestión de la información estadística del PCC. Actualmente la herramienta se encuentra en funcionamiento en todos los Comités Municipales del PCC de la provincia La Habana (GARCÍA CHACÓN, 2015).

El sistema esta implementado usando el lenguaje PHP con el IDE[1] de desarrollo NetBeans y PostgreSQL en su versión 8.4 como gestor de bases de datos (GARCÍA CHACÓN, 2015).

[1] **IDE:**(*Integrated Development Environment*, Entorno Integrado de Desarrollo), entorno de programación que ha sido empaquetado como un programa de aplicación. Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica.

Principales funcionalidades del sistema

INFOEST se encarga de recoger toda la información correspondiente a la militancia del PCC gestionando los militantes y las estructuras a las que pertenecen y manteniendo el control de la vida del militante, las altas y bajas. El sistema gestiona sanciones internas y externas, fallecimientos, anulaciones de militancia y los traslados. Además, muestra los listados nómina de cada estructura y realiza reportes estadísticos con respecto a períodos de tiempo y trámites realizados (GARCÍA CHACÓN, 2015).

Este sistema está diseñado para ser utilizado por todas las estructuras y niveles del PCC en el país pero al no contar con una red privada para la transmisión de los datos y que la información gestionada es delicada y confidencial, solo se utiliza en el Comité Central del PCC y en los Comités Municipales de La Habana y no es posible utilizarlo en las estructuras que se subordinan a estos, por lo cual el Comité PCC-UCI no tiene acceso al uso del mismo. En consecuencia, el sistema no permite la comunicación entre las estructuras ocasionando que puedan ocurrir errores de pérdida o duplicación de los datos en la documentación y que no exista una centralización de la información.

- ***Sistema para la gestión de los procesos de la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas***

Este trabajo de diploma realizado en el 2014 muestra los resultados de la implementación de la herramienta dataFEU. Esta es una aplicación informática que asegura la calidad del proceso de evaluación estudiantil y apoya a sus principales dirigentes en el proceso de toma de decisiones. Se centra principalmente en el proceso de integralidad y caracterización de los estudiantes registrando mediante evidencias la trayectoria de cada uno de ellos en un expediente digital (GONZÁLEZ CARDOSO, 2014).

El sistema dataFEU fue desarrollado sobre los marcos de trabajo Symfony y Ext JS empleando MySQL como servidor de base de datos, ajustándose al uso de tecnologías libres. Cuenta con cinco subsistemas que se integran entre sí y permiten el correcto funcionamiento, integridad y persistencia de los datos que se introducen y procesan. Brinda información a sistemas externos mediante servicios web que garantizan la interoperabilidad del mismo (GONZÁLEZ CARDOSO, 2014).

Principales funcionalidades del sistema

Entre las funcionalidades está gestionar los méritos, sanciones, cargos, bonificaciones, alumnos ayudantes, resultados académicos, desempeño en el proyecto de producción y las tareas evaluadas. Permite gestionar todo lo relacionado con los eventos y actividades, su planificación, participación de los estudiantes y evaluación. Además cuenta con un subsistema alertas y notificaciones que es el encargado

de gestionar todas las notificaciones y avisos tanto vía correo como dentro del propio sistema (GONZÁLEZ CARDOSO, 2014).

Este sistema fue diseñado exclusivamente para los procesos de la FEU y sería preciso adaptarlo y desarrollar las funcionalidades necesarias para el uso en el PCC donde los procesos se efectúan de manera diferente. Además, como la herramienta es una aplicación web ya implementada, no se puede simplemente reutilizar pues no se tiene acceso al código fuente de la aplicación para realizar las modificaciones.

- ***Sistema Gestor de Información para las Organizaciones: Módulo UJC***

El Sistema Gestor de Información para las Organizaciones fue desarrollado en el año 2012 con el objetivo de mejorar la gestión de la información en los procesos de la organización Unión de Jóvenes Comunistas (UJC) en la Universidad de las Ciencias Informáticas, y potencialmente en otras instituciones cuya organización estructural sea similar. Está desarrollada usando el marco de trabajo *Symfony 2* (CRESPO ESTÉVEZ, 2012).

Principales funcionalidades del sistema

Permite gestionar las estructuras, los militantes, la Plantilla de Control de la Militancia y las actas con sus respectivas opiniones, acuerdos, inquietudes y orden del día. Además, posibilita gestionar los usuarios del sistema, los roles de estos y la configuración (CRESPO ESTÉVEZ, 2012).

Este sistema mejora varios de los procesos de la organización UJC que se realizaban de forma manual, y a su vez que la información se encuentre accesible y centralizada para evitar pérdidas innecesarias. El control de acceso mejora la confidencialidad de la información que se maneja en el mismo (CRESPO ESTÉVEZ, 2012).

Este sistema cuenta con algunas de las funcionalidades necesarias para la propuesta de solución, pero no abarca todos los procesos que se necesitan informatizar debido a que no gestiona los crecimientos, traslados, sanciones y desactivaciones, que son procesos claves en su funcionamiento, y no permite mantener un control del pago de la cotización. Además su integración con el SGU se hace difícil teniendo en cuenta la arquitectura sobre las cuales está desarrollado cada uno.

- ***Análisis de un sistema para el control de los procesos políticos en la Facultad 4***

Este trabajo de diploma tuvo como objetivo realizar el análisis de un sistema automatizado como punto de partida para su posterior diseño e implementación (CONCEPCIÓN SINGLER, 2007).

La propuesta de este trabajo consistía en realizar el análisis de un sistema que centralice, gestione y publique información referente a los procesos que se desarrollan en las organizaciones políticas de la Facultad 4. Este garantizaría rapidez en el manejo de información, integridad y seguridad de la misma.

Se pretendía que este sistema pudiera ser utilizado por todas aquellas personas que intervienen en las actividades de estas organizaciones en la facultad y debía permitir a cada uno de ellos desarrollar solo aquellas tareas que sean de su interés (CONCEPCIÓN SINGLER, 2007).

Principales funcionalidades del sistema

El sistema de ser implementado permitiría principalmente gestionar los usuarios, los militantes, la evaluación del militante, la participación en actividades, la participación en eventos y registrar sus resultados. Además permitiría gestionar el universo juvenil, el pago de la cotización, las sanciones, las reuniones y las actas (CONCEPCIÓN SINGLER, 2007).

Este trabajo de diploma se enfoca en los procesos claves necesarios para la gestión de las organizaciones políticas y de masas, pero está centrado al trabajo en una facultad por lo que no se tiene en cuenta todos los pasos de los procesos cuando se realizan a nivel de universidad.

1.4. Aportes de los sistemas homólogos estudiados

Luego de estudiar estos sistemas se puede concluir que a pesar de ser sistemas de gestión de organizaciones políticas o de masas, no cumplen en totalidad con las características mencionadas en el epígrafe 1.3 para la obtención de una solución completa.

Los sistemas estudiados necesitan ser adaptados y requieren funcionalidades para realizar correctamente el registro y control de los procesos dela vida interna del PCC. Además, su integración al SGU resulta compleja puesto que no están desarrollados con el mismo marco de trabajo y no se tiene acceso al código fuente de las herramientas para su uso y modificación.

No obstante el estudio de estos sistemas ha aportado un conjunto de características relevantes para la realización de la propuesta de solución.

La aplicación web dataFEU ofrece características que pueden ser aplicadas a la solución propuesta como la definición de los niveles de acceso según el tipo de usuario y la responsabilidad que desempeñe en la organización para permitir el acceso al sistema y a la información. Además, permite enviar correos a destinatarios seleccionados con mensajes previamente definidos por el usuario o por los administradores del sistema y notifica a los usuarios cuando se realizan cambios en el perfil o se aproximan actividades de su interés. Toda la información generada por las diferentes áreas de la vida universitaria es concentrada

en un expediente digital, donde se muestra el accionar de cada estudiante y se almacena en un archivo histórico logrando un grado mayor de persistencia de la información.

Con el estudio del sistema INFOEST se obtuvo información de cómo realizar la gestión de los datos de la militancia y de los procesos de sanciones, desactivación y traslados, y permitió que se tomara en cuenta la gestión de los fallecimientos en la propuesta de solución.

1.5. Metodología y herramientas utilizadas en la implementación de la solución

La solución de software, se desarrolla utilizando las tecnologías y herramientas usadas en el Sistema de Gestión Universitaria al que se desea integrar. Este sistema establece utilizar como lenguaje de programación PHP, GUUD como marco de trabajo, PostgreSQL como gestor de base de datos y se hace uso de la metodología Desarrollo Ágil con Calidad (DAC).

1.5.1. Lenguajes

Nombre: HTML 4

Descripción de su uso: El HTML (*Hyper Text Markup Language*) es el lenguaje con el que se escriben las páginas web. Es un lenguaje de hipertexto que permite escribir texto de forma estructurada y que está compuesto por etiquetas, que marcan el inicio y el fin de cada elemento del documento. Un documento hipertexto no sólo se compone de texto, puede contener imagen, sonido, vídeo, y el resultado puede considerarse como un documento multimedia. Los documentos HTML deben tener la extensión html o htm, para que puedan ser visualizados en los navegadores. Los navegadores se encargan de interpretar el código HTML de los documentos y de mostrar a los usuarios las páginas web resultantes del código interpretado. El HTML 4 desarrolla el lenguaje HTML con mecanismos para hojas de estilo, ejecución de scripts, marcos, objetos incluidos, soporte mejorado para texto de derecha a izquierda y direcciones mezcladas, tablas más ricas y mejoras en formularios, ofreciendo mejoras de accesibilidad para personas con discapacidades (W3C, 2015)

Nombre: PHP 5.6.7

Descripción de su uso: PHP es un acrónimo recursivo que significa ***P****HP* ***H****ypertext* ***P****re-processor* y está publicado bajo la *PHP License.* La *Free Software Foundation* considera esta licencia como software libre. Es un lenguaje de programación, interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (*server-side scripting*) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos

de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt[2] o GTK+[3]. Es un lenguaje multiplataforma y tiene capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL (PHP, 2015).

Nombre: JavaScript 1.8

Descripción de su uso: JavaScript es un lenguaje de programación interpretado, se define como orientado a objetos y basado en prototipos. Es dinámico, responde a eventos en tiempo real como presionar un botón, pasar el puntero del ratón sobre un determinado texto o el simple hecho de cargar la página o caducar un tiempo. Se utiliza principalmente en su forma del lado del cliente (*client-side*), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, aunque existe una forma de JavaScript del lado del servidor (*Server-side JavaScript*). JavaScript puede incluirse en cualquier documento y es compatible con HTML en el navegador del cliente, ya sea PHP, ASP[4], JSP[5] y SVG[6] (EGUILUZ, 2015).

Nombre: XML 1.0

Descripción de su uso: XML (e***X****tensible* ***M****arkup* ***L****anguage)* es un lenguaje de marcas desarrollado por el *World Wide Web Consortium* (W3C) utilizado para almacenar datos en forma legible. Deriva del lenguaje SGML[7] y permite definir la gramática de lenguajes específicos para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones se deben comunicar entre sí o integrar información. Después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se pueda continuar utilizando sin complicación alguna. Si un tercero decide usar un documento creado en XML, es sencillo

[2] Qt es una biblioteca multiplataforma ampliamente usada para desarrollar aplicaciones con interfaz gráfica de usuario.

[3] GTK+ o *The GIMP Toolkit* es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME.

[4] *Active Server Pages* (ASP) es una tecnología de Microsoft del tipo "lado del servidor" para páginas web generadas dinámicamente.

[5] *JavaServer Pages* (JSP) es una tecnología que ayuda a los desarrolladores de software a crear páginas web dinámicas.

[6] Gráficos Vectoriales Redimensionables (del inglés *Scalable Vector Graphics*) o SVG son una especificación para describir gráficos vectoriales bidimensionales, tanto estáticos como animados.

[7] SGML son las siglas de *Standard Generalized Markup Language* o "Estándar de Lenguaje de Marcado Generalizado".

entender su estructura y procesarla. XML mejora la compatibilidad entre aplicaciones. Se puede comunicar aplicaciones de distintas plataformas, sin que importe el origen de los datos (W3C, 2008).

Nombre: CSS 3
Descripción de su uso: Las hojas de estilo en cascada (en inglés *Cascading Style Sheets*, CSS) constituyen un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). La información de estilo puede ser adjuntada como un documento separado o en el mismo documento HTML. En este último caso podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo *"<style>".* CSS permite el control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo. Los navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad. Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario (W3C, 2011).

Nombre: Lenguaje de Modelado Unificado (UML) 2.0
Descripción de su uso: *Unified Modeling Language* (UML) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. 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 (OMG, 2.0).

Nombre: Notación para el Modelamiento de Procesos de Negocio (BPMN) 2.0
Descripción de su uso: *Business Process Modeling Notation* (BPMN) es una notación gráfica estandarizada que permite el modelado de procesos de negocio, en un formato de flujo de trabajo (*workflow*). BPMN tiene la finalidad de servir como lenguaje común para cerrar la brecha de comunicación que frecuentemente se presenta entre el diseño de los procesos de negocio y su implementación. BPMN está planeada para dar soporte únicamente a aquellos procesos que sean aplicables a procesos de negocios (BPMN, 2015).

1.5.2. Herramientas

Nombre: NetBeans 8.0

Licencia del producto: Doble licencia; *Common Development and Distribution License (CDDL)* y *GNU General Public License* versión 2 *with Classpath exception (GPL2).*

Descripción de su uso: El IDE NetBeans es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java pero puede servir para otros lenguajes de programación. Existe además un número importante de módulos para extender el IDE NetBeans. Es un producto libre y gratuito sin restricciones de uso. NetBeans es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. Soporta el desarrollo de todos los tipos de aplicación Java (J2SE[8], web, EJB[9] y aplicaciones móviles) (NETBEANS, 2015).

Nombre: Evolus Pencil 2.0.5

Licencia del producto: Se encuentra bajo licencia de código abierto GPL (*GNU General Public License*).

Descripción de su uso: Herramienta de prototipado libre y de código abierto con interfaz gráfica de usuario, que se instala con facilidad y crea maquetas de plataformas de escritorio populares. Los proyectos se pueden exportar en los formatos HTML, PNG, documento *Open office*, documento de *Word* y PDF. Permite instalar plantillas definidas por el usuario, operaciones de dibujo estándar: alinea, escala y rota, y la adición de los objetos externos (PENCIL PROJECT, 2015).

Nombre: Suite de Visual Paradigm 5.0

Descripción de su uso: Suite de productos para desarrollar software de manera eficiente, rápida y de forma colaborativa. VP Suite es una solución total de modelado de procesos de negocio para la generación de código. VP Suite está diseñado para una amplia gama de usuarios, incluidos los ingenieros de software, analistas de sistemas, analistas de negocios, arquitectos de sistemas por igual, que están interesados en la construcción de sistemas de software a gran escala de forma fiable. VP Suite se compone de todos los mejores productos de Visual Paradigm incluyendo: Visual Paradigm for UML 8.0

[8] *Plataforma Java 2, Standard Edition* (J2SE) es una colección de interfaces de programación de aplicaciones del lenguaje de programación Java útiles para muchos programas de la Plataforma Java.

[9] *Enterprise JavaBeans* (también conocidos por sus siglas EJB) son una de las interfaces de programación de aplicaciones que forman parte del estándar de construcción de aplicaciones empresariales de *Java Platform, Enterprise Edition o Java EE*.

Enterprise Edition, Smart Development Environment 6.0 Enterprise Edition y DB Visual ARCHITECT 6.0 Professional Edition (VISUAL PRADIGM, 2015).

Nombre: PostgreSQL 9.4.1

Licencia del producto: Distribuida bajo la licencia BSD[10].

Descripción de su uso: PostgreSQL es un sistema de gestión de bases de datos objeto-relacional. Es compatible con una gran parte del estándar SQL y ofrece muchas características modernas como: consultas complejas, claves externas, disparadores, vistas actualizables e integridad transaccional. También, PostgreSQL se puede extender por el usuario en muchas maneras, por ejemplo mediante la adición de nuevos tipos de datos, funciones, operadores, funciones de agregado y métodos de índice. Debido a la licencia liberal, PostgreSQL puede ser utilizado, modificado y distribuido por cualquier persona de forma gratuita para cualquier propósito, ya sea privada, comercial o académico. (POSTGRESQL, 2015).

Nombre: PgAdmin III 1.14.0

Licencia del producto: Licencia BSD.

Descripción de su uso: Es una aplicación de código abierto para el diseño y manejo de bases de datos para su uso con PostgreSQL. La aplicación se puede utilizar para manejar PostgreSQL 7.3 y superiores, funciona sobre casi todas las plataformas. PgAdmin III por su diseño permite desde escribir consultas SQL[11] simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. Incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados, por citar algunos. La conexión al servidor se hace mediante conexión TCP/IP[12] y puede encriptarse mediante SSL[13] para mayor seguridad (PGADMIN, 2015).

[10] La licencia BSD es la licencia de software otorgada principalmente para los sistemas BSD (*Berkeley Software Distribution*).

[11] SQL (*Structured query language*) es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

[12] TCP/IP es un conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre computadoras.

[13] *Secure Sockets Layer* o *SSL* es un protocolo criptográfico que proporciona comunicaciones seguras por una red, comúnmente Internet.

Nombre: Apache 2.4.7

Descripción de su uso: Apache es software libre y un proyecto de la Fundación de Software Apache, con el objetivo de suministrar un servidor seguro, eficiente y extensible que proporcione servicios HTTP[14] en sincronía con los estándares HTTP actuales. Apache es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos HTTP. Es multiplataforma y modular, puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona y con la API[15] de programación de módulos, para el desarrollo de módulos específicos. Además es extensible ya que gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor. El Servidor HTTP Apache es un servidor web HTTP de código abierto para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras (OPENSUSE, 2015).

Nombre de la herramienta: Apache JMeter 2.3.1

Descripción de su uso: JMeter es un proyecto de Apache que puede ser utilizado como una herramienta de prueba de carga para analizar y medir el desempeño de una variedad de servicios, con énfasis en aplicaciones web. JMeter puede ser usado como una herramienta de pruebas unitarias para conexiones de bases de datos con FTP, LDAP, Servicios web, HTTP y conexiones TCP genéricas. Apache JMeter es una aplicación de escritorio de código abierto diseñada para cargar el comportamiento funcional de prueba y medir el rendimiento. Originalmente fue creada para probar aplicaciones web, pero se ha expandido a otras funciones de prueba. Se puede utilizar para simular una carga pesada en el servidor, de red o un objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga (JMETER, 2015).

1.5.3. Marco de trabajo

Nombre: GUUD 2.0

Descripción de su uso: Marco de trabajo desarrollado por el departamento de desarrollo de la Dirección de Informatización para la creación de aplicaciones web escritas en PHP. Constituye un híbrido entre el marco de trabajo de PHP CodeIgniter y la biblioteca de Javascript, JQuery. Implementa el patrón Modelo-

[14] HTTP (*Hypertext Transfer Protocol*) es el protocolo usado en cada transacción de la *World Wide Web*.

[15] API (*Application Programming Interface*) es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Vista-Controlador (MVC) así como la programación orientada a aspectos. Realiza un manejo de excepciones y mensajes. Modulariza el marco de trabajo CodeIgniter e implementa como estrategia de comunicación entre módulos la Inversión de Control (IoC). Hace uso de *templates* para el renderizado de las vistas. Contiene una serie de componentes visuales que permiten la fácil interacción del usuario con la aplicación tales como el calendario y el *grid*, entre otros (TELLEZ COLLAZO, 2013).

Nombre: CodeIgniter 1.7.3

Descripción de su uso: CodeIgniter es un marco de trabajo para el desarrollo de aplicaciones, una herramienta para personas que crean webs usando PHP. Es de código libre. Su meta es permitir desarrollar proyectos mucho más rápido que si se hiciera escribiendo el código desde cero, proporcionando una gran variedad de bibliotecas para las tareas más corrientes, así como una interfaz simple y una estructura lógica para acceder a estas bibliotecas. CodeIgniter contiene una serie de bibliotecas que sirven para el desarrollo de aplicaciones web y además propone una manera de desarrollarlas que debemos seguir para obtener provecho de la aplicación. CodeIgniter implementa el proceso de desarrollo llamado Modelo-Vista-Controlador (MVC), que es un estándar de programación de aplicaciones, utilizado tanto para hacer sitios web como programas tradicionales (ÁLVAREZ, 2015).

Nombre: JQuery 1.9.2

Licencia del producto: Posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2.

Descripción de su uso: JQuery es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM[16], manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX[17] a páginas web. Es software libre y de código abierto. 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 (JQUERY, 2015).

[16] DOM (*Document Object Model*) es esencialmente una interfaz de programación de aplicaciones que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

[17] AJAX (*Asynchronous JavaScript And XML*) es una técnica de desarrollo web para crear aplicaciones interactivas.

1.5.4. Metodología de desarrollo

Metodología Desarrollo Ágil con Calidad (DAC)

La metodología DAC es un proceso colaborativo, recursivo-iterativo, incremental y guiado por procesos y requisitos. Su modelo del proceso es una adaptación del modelo en Cascada a los modelos Programación Extrema y Desarrollo Concurrente (SÁNCHEZ MÉNDEZ, 2015).

DAC plantea que el problema una vez identificado y definido debe ser descompuesto en problemas más pequeños, y si es necesario, realizar con estos la misma operación. Cada sub-problema será resuelto mediante un componente y el problema resuelto será el software o producto final; por lo que las entregas en DAC son a nivel de iteración, en la que habrá obligatoriamente un incremento del producto a partir de la solución de un componente del mismo. Además en cada iteración se define como mínimo un hito a cumplir por cada fase del proceso. Al finalizar cada iteración se realiza la integración del componente al producto obtenido hasta el momento realizando pruebas de integración. Al finalizar las iteraciones se pueden realizar liberaciones del componente y transición del mismo dentro de la fase cierre de iteración. Las iteraciones no tienen que desarrollarse todas al mismo tiempo sino que al contar con un equipo pequeño este se va a ir moviendo de una iteración a otra a medida que estas vayan terminando de acuerdo a un orden de prioridad establecido en el plan del proyecto. Este proceso tiene 8 actividades del marco de trabajo del proceso común llamadas fases o procesos del ciclo de vida: inicio, análisis y diseño arquitectónico, requisitos, construcción, cierre de iteración, liberación, transición y cierre, ocurriendo las iteraciones concurrentes entre las fases de requisitos, construcción y cierre de iteración. Además entre las fases de requisitos y construcción puede ocurrir un ciclo pues a medida que los requisitos son especificados estos pueden ir entrando a la fase de construcción (SÁNCHEZ MÉNDEZ, 2015).

1.6. Conclusiones parciales

Con el estudio y análisis de los sistemas de gestión de organizaciones, se logró fundamentar las bases teóricas de la investigación y se analizaron los conceptos básicos asociados al PCC para un mayor entendimiento. Además, se pudieron adquirir los conocimientos previos sobre las características principales de estos sistemas, aunque no cumplen con todas las condiciones requeridas. La caracterización de las herramientas, tecnologías, metodología y lenguajes de programación permitió conocer sus beneficios, así como formar las bases propicias para crear una propuesta de solución, que a su vez, cumpla con el objetivo de la investigación.

Capítulo 2. Descripción de la propuesta de solución

2.1. Introducción

En el presente capítulo se realiza la descripción de la propuesta solución. Además se expone las principales características del sistema, se describen los procesos de la vida interna del PCC como organización política identificando las actividades que se informatizarán y se definen los artefactos generados en el análisis y diseño. Se describen las técnicas utilizadas para la obtención de requisitos de software y se realiza la descripción de los mismos. Se caracteriza la arquitectura y patrones de diseño implementados en el marco de trabajo.

2.2. Modelado de procesos del negocio

Un proceso de negocio es un conjunto de tareas relacionadas lógicamente, llevadas a cabo para generar productos y servicios. Modelar los procesos de negocio es una parte esencial de cualquier proceso de desarrollo de software. Permite capturar el esquema general y los procedimientos que gobiernan el negocio. Este modelo provee una descripción de dónde se va a ajustar el sistema de software considerado dentro de la estructura organizacional y de las actividades habituales. También provee la justificación para la construcción del sistema de software al capturar las actividades manuales y los procedimientos automatizados habituales que se incorporarán en el nuevo sistema (SPARKS, 2015). En el **Anexo 2** se encuentran los diagramas de procesos de negocio realizados para describir el flujo de los procesos a informatizar.

2.2.1. Flujo actual de los procesos de la vida interna del PCC en la UCI

Traslados

Los traslados de los militantes pueden ser internos o externos. Los traslados internos se realizan dentro de la propia universidad o del municipio, y los externos hacia otros municipios de La Habana, otras provincias, el MINFAR, el MININT o al exterior.

El proceso inicia cuando un militante solicita el traslado y el núcleo al que pertenece el militante acuerda, en una reunión ordinaria o extraordinaria, el traslado del militante. Si el traslado es interno no se genera documentación y el movimiento del militante se registra en la Plantilla de Control de la Militancia del núcleo, además se envía una notificación por correo electrónico certificando el traslado o la incorporación del militante a los Secretarios Generales de los núcleos implicados y a los miembros del comité que atienden estos núcleos, también debe notificársele al Activista de cotización del Comité PCC-UCI.

En caso de ser un traslado externo dentro del país el Secretario del Comité PCC-UCI encargado de la esfera de la vida interna del PCC recibe una solicitud de traslado externo y el militante que se traslada debe presentar una carta de aceptación del Secretario General del núcleo al cual se va a incorporar, documento que se archiva en expediente. Luego de la aprobación del traslado, el Secretario del Comité PCC-UCI elabora el modelo de traslado e incorporación, lo firma y archiva una copia en el expediente del militante. La otra copia es entregada en el Departamento de estadísticas del Comité Municipal del PCC. Se entrega el expediente del militante al organismo superior del núcleo al que se va a trasladar el militante a través de este mismo. Por último, se actualiza la Plantilla de Control de la Militancia del núcleo donde causa baja.

Si el traslado externo es hacia el exterior del país el Secretario del Comité PCC-UCI recibe la solicitud de traslado externo. Luego de la aprobación del traslado, el Secretario del Comité PCC-UCI elabora el modelo de traslado al exterior, lo firma y archiva una copia en el expediente del militante. En este caso no se entrega el expediente al militante, este se guarda en el archivo de militantes en el exterior y el militante debe entregar carné del PCC. Por último, se actualiza la Plantilla de Control de la Militancia del núcleo donde causa baja.

Incorporaciones

El militante que se incorpora entrega el modelo de traslado e incorporación conjuntamente con el expediente del militante en el Comité PCC-UCI, donde el Secretario firma la boleta y le asigna un número de expediente según la disponibilidad de los archivos. Se introducen los datos del nuevo militante en la Plantilla de Control de la Militancia del núcleo hacia donde se dirige y se realiza una reunión en el núcleo donde se informa la incorporación.

Sanciones

Las sanciones a aplicar a un militante pueden ser internas como: amonestación, separación del cargo y suspensión temporal de derechos; o externas: separación de las filas y expulsión de las filas.

La sanción se propone y analiza en la reunión del núcleo donde se toma como acuerdo. En la reunión el Activista de acta elabora un acta de reunión la cual se entrega al Secretario General del núcleo para que la firme y este posteriormente hace entrega de la misma al miembro del Comité PCC-UCI que atiende sanciones.

Esta sanción se analiza en el Comité PCC-UCI y si es una sanción interna se realiza un análisis de la sanción y se genera un acta donde se ratifica o rectifica esa sanción y se archiva posteriormente en el

expediente del militante. Si la sanción es externa, se realiza un acta que el miembro del Comité PCC-UCI que atiende sanciones envía a la Comisión de sanciones del Comité Municipal del PCC conjuntamente con el acta del núcleo. Las comisiones de sanciones de los comités municipales pueden ratificar, rectificar o anular las sanciones adoptadas por los núcleos. Allí se elabora un documento de resolución de sanción con el resultado del análisis de la comisión, el miembro del Comité PCC-UCI que atiende sanciones lo recoge en el municipio y hace entrega del mismo al Secretario General del núcleo donde, en otra reunión extraordinaria, el sancionado firma el documento conjuntamente con el Secretario General del núcleo. Este hace entrega de dicho documento al Secretario del Comité PCC-UCI que lo archiva en el expediente del militante y guarda el expediente en el archivo de militantes inactivos.

Si la sanción es separación o expulsión de las filas, el militante causa baja del núcleo al que pertenece y se actualiza la Plantilla de Control de la Militancia. El militante debe entregar el carné de PCC.

Desactivaciones

El inicio de un proceso de desactivación a un militante puede ser considerado en un núcleo u organismo del partido, por las siguientes vías:

- Solicitud verbal o por escrito del militante a su núcleo.
- Solicitud en el núcleo o por decisión de un organismo de dirección del partido si se considera que el militante no está en condiciones y/o posibilidades para continuar militando.

La desactivación de un militante se decide por acuerdo del núcleo y debe ser ratificada por el organismo inmediato superior al núcleo. Para esto se crea una comisión de desactivación donde el Jefe de la comisión elabora un expediente de desactivación.

Posteriormente se realiza una reunión extraordinaria donde se elabora un acta que se entrega al Secretario General del núcleo para que este la firme y la apruebe, si dicha acta no es aprobada se le informa a la Comisión de desactivación para que vuelva a revisar el caso. Si se aprueba el acta se archiva en el expediente de desactivación y el Secretario General del núcleo la envía al Secretario del Comité PCC-UCI. A continuación se realiza la reunión del Comité PCC-UCI y se elabora un acta extraordinaria la que se le hace entrega al Secretario General del PCC para que este la firme y la apruebe. Si no es aprobada la desactivación se le informa al núcleo del PCC para que vuelvan a revisar el caso. En caso contrario el miembro del Comité PCC-UCI que atiende sanciones y desactivaciones envía esta acta a la Comisión de sanciones del Comité Municipal del PCC. Allí se elabora un documento de Resolución de desactivación y el miembro del Comité del PCC que atiende sanciones y desactivaciones lo recoge en el

municipio y hace entrega del mismo al Secretario General del núcleo donde en otra reunión extraordinaria el militante en proceso de desactivación firma el documento conjuntamente con el Secretario General del núcleo y este hace entrega de dicho documento al Secretario de vida interna del Comité PCC-UCI el cual lo archiva en el expediente del militante junto con el carné del PCC y guarda el expediente en el archivo de militantes inactivos.

Proceso de crecimiento al PCC

El crecimiento al PCC inicia con el proceso sociopolítico donde se realiza el análisis del listado de personas a crecer. El listado es evaluado por el núcleo en una reunión donde los miembros aprueban las personas que van a iniciar el proceso de crecimiento se elabora un acta que es firmada por el Secretario General del núcleo y este hace entrega al miembro de trabajo político ideológico del Comité PCC-UCI para que la guarde en el archivo de personas en crecimiento. Las personas para este proceso son seleccionadas en una reunión de consulta con las masas para militantes de la UJC o asamblea de trabajadores ejemplares para personas no militantes de la UJC. Una vez aprobado el listado de personas a crecer, se determinan los dúos de crecimiento, conformado por militantes de ese núcleo. El jefe de la comisión de crecimiento elabora un acta que es firmada por el Secretario General del núcleo y archiva esta acta en el expediente de la persona en proceso de crecimiento.

La persona en proceso debe realizar una autobiografía de toda su trayectoria de vida que entrega con 3 fotos tipo carné al jefe de la comisión de crecimiento para que lo archive en el expediente. Luego, los militantes designados por el núcleo se entrevistarán con cada una de las personas seleccionadas para crecer y se procederá a la comprobación de la veracidad de las informaciones o datos más importantes sobre cada uno de los compañeros incluidos en el proceso. Al concluir esto, el jefe de la comisión de crecimiento realiza comprobaciones de la persona en proceso, las firma y las archiva en el expediente. Seguidamente realiza un resumen de trayectoria revolucionaria, la firma y la presenta en la reunión de admisión del núcleo del PCC. En dicha reunión los miembros del núcleo emiten opiniones para que el dúo de comisión de crecimiento pueda elaborar el acta de reunión de admisión, que debe ser firmada por el jefe de la comisión de crecimiento y el Secretario General del núcleo y se archivan ambos documentos en el expediente.

Una vez en posesión de todos los elementos necesarios sobre cada uno de los casos, el núcleo se reunirá para evaluar y aprobar o no la admisión como militantes de los compañeros analizados. Una vez concluido todo este proceso el jefe de la comisión de crecimiento entrega al miembro responsable del crecimiento

del Comité PCC-UCI el expediente para su revisión y aprobación. Si este expediente presenta algún inconveniente se le notifica al Secretario General del núcleo y este le hace entrega del expediente al dúo de la comisión de crecimiento para que lo corrija y lo vuelva a enviar.

De lo contrario, si el expediente no presenta ninguna dificultad el miembro del Comité del PCC responsable de la comisión de crecimiento presenta en el Comité PCC-UCI el expediente para su valoración, donde se elabora un modelo del Comité del PCC con las opiniones recogidas, lo firma el Secretario General del PCC y posteriormente se archiva en el expediente del joven en proceso.

Dicho miembro del Comité PCC-UCI envía el expediente al municipio donde hace entrega a la Comisión de Crecimiento Municipal para que ratifique el acuerdo del núcleo y se elabora un documento, el cual recoge la decisión de la comisión de crecimiento y se archiva en el expediente. Al cabo de la semana el miembro del Comité PCC-UCI recoge el expediente y le hace entrega del mismo al dúo de la comisión de crecimiento.

El dúo de la comisión de crecimiento lo presenta ante los miembros del núcleo en la reunión de conclusiones e incorporación al núcleo donde elabora un acta de conclusión e incorporación al núcleo y la firman el Secretario General del núcleo y la persona procesada, seguidamente se archiva en el expediente.

Luego se realiza la asamblea de presentación a las masas, donde se presentan los nuevos militantes del partido. El dúo de la comisión de crecimiento elabora un acta de presentación a las masas, la firma junto al Secretario General y se archiva en el expediente. Posteriormente el dúo de la comisión de crecimiento le entrega dicho expediente al Secretario del Comité PCC-UCI para que este lo guarde en el archivo del centro. Por último se realiza la entrega del carné del partido en un acto solemne o se realiza una entrega informal.

Cotización

La cotización de los militantes es un principio organizativo del partido. La cuota a pagar cada mes se determina sobre la base de los ingresos percibidos por los militantes en ese período, incluidas las diversas fuentes o vías de que provengan. Para controlar el pago se utiliza un listado con el nombre y apellidos de los militantes de ese núcleo, indicando salario devengado en el mes a cotizar y el importe que debe pagar a partir del salario devengado. Este resumen mensual de liquidación de cotización es un documento que elabora el Activista de cotización de cada núcleo. Cada militante abona su cotización en su núcleo de procedencia, donde el Activista de cotización se encarga de recolectar ese dinero y se lo presenta al

Secretario General del núcleo junto con el resumen para que lo apruebe y lo firme, y luego lo despacha al Miembro del Comité PCC-UCI que atiende cotización, este lo firma como recibido y lo entrega en el Comité Municipal del PCC en el Departamento de Finanzas, donde el Especialista Económico especifica el número de Registro de salida del documento y se lo entrega nuevamente al Miembro del Comité PCC-UCI que atiende cotización, y este lo guarda en el archivo de cotización del Comité PCC-UCI.

Registro de actas de reuniones

Mensualmente cada núcleo se reúne al menos una vez generando un acta digital realizada por el Activista de acta y revisada por el Secretario General que luego la entrega al Secretario de vida interna del Comité PCC-UCI. Este realiza el procesamiento de las actas de las reuniones de los núcleos durante el mes a medida que van llegando las actas impresas o en formato digital a través del correo electrónico, teniendo en cuenta que los núcleos deben enviar las actas antes de las 72 horas de realizada la reunión. El último día del mes se realiza el registro general de las actas de las reuniones de los núcleos.

En este registro general de las actas se registra los núcleos, el tipo de reunión, la fecha de la reunión, la fecha de entrada del acta de la reunión, el tiempo transcurrido entre fecha de realizada la reunión y fecha de entrada del acta, el número de acta, la asistencia, por ciento de asistencia, ausentes y las causas de ausencias, la cantidad de puntos del orden del día, el total de acuerdos tomados, si hubo participación del organismo superior, si se realizó rendición de cuentas y los temas discutidos en la reunión.

Plan de trabajo

El plan de trabajo es un documento que se elabora en cada núcleo, comité primario y Comité PCC-UCI de manera trimestral, este documento es elaborado por el Secretario General. Se presenta en reunión ordinaria donde es aprobado por el núcleo y se archiva en el archivo de documentos del núcleo, comité primario y Comité PCC-UCI

Al cierre de cada trimestre se evalúa el cumplimiento del plan de trabajo del trimestre que cierra y se aprueba el del próximo trimestre. Por tanto los meses en que se evalúa este tema en las reuniones de los núcleos son marzo, junio, septiembre y diciembre.

2.2.2. Análisis crítico de la ejecución actual de los procesos

Los procesos descritos anteriormente no son lo suficientemente eficientes a la hora de ser ejecutados a pesar de cumplir con sus objetivos. Esto ocurre debido al tiempo perdido intercambiando información, recogiendo datos, buscando documentos que son necesarios consultar para tomar una decisión

determinada y que no todos se encuentran en el mismo sitio, esto trae consigo que el personal involucrado se agote innecesariamente.

Además, cuando se va a notificar de cualquier proceso en específico que involucre a una cantidad de personas determinada, hay que buscar específicamente sus direcciones electrónicas para que los mismos tengan conocimiento de lo que se pretende. En fin, el aprovechamiento que se le da a los recursos no es el esperado y existe descentralización de la información.

2.2.3. Procesos objeto de la informatización

Se desean informatizar los procesos mencionados anteriormente de forma tal que cada uno de los roles del personal que interviene en la ejecución de las actividades, puedan tener acceso total a la parte que le corresponda del sistema que se propone, donde podrán consultar, introducir y modificar datos, además, serán responsables por la información que manejan. También, se brindarán una serie de reportes a partir de estos procesos que facilitarán la toma de decisiones.

2.3. Reglas de negocio

Las reglas del negocio son restricciones que las organizaciones tienen definidas y son vitales para lograr sus objetivos. Los procesos de la vida interna del PCC presentan diversas reglas que permiten que estos procesos se realicen con la mayor calidad posible.

1. Los núcleos y los comités primarios se subordinan al Comité PCC-UCI. Los núcleos también se subordinan a los comités primarios.
2. Todos los militantes pertenecen a un núcleo y cada militante puede cumplir una responsabilidad o cargo en la estructura a la que pertenece.
3. En todas las estructuras debe existir un Secretario General y un Activista de acta.
4. Las estructuras deben estar formadas por al menos tres militantes y un máximo de 35 militantes.
5. La cantidad de militantes en la Plantilla de Control de la Militancia varía de acuerdo a los ingresos, incorporaciones, traslados y las bajas por sanciones externas o desactivaciones.
6. Si la sanción es suspensión temporal de derechos, esta sanción se puede adoptar por un período no mayor de un año, ni menor de tres meses.
7. Si la sanción es de separación de las filas o expulsión de las filas, el militante dejará de cotizar automáticamente.

8. Para realizar un traslado externo o incorporación, se debe elaborar y firmar el Modelo Oficial de Traslado e Incorporación.
9. Las vías de ingreso al PCC son la asamblea de consulta con las masas para militantes de la UJC o la asamblea de trabajadores ejemplares.
10. Un dúo de crecimiento puede atender más de un crecimiento, esto se decide de acuerdo a la cantidad de militantes del PCC en cada núcleo y el número de personas a crecer.
11. Cuando se realiza la reunión de conclusión el militante se incorpora al núcleo y comienza a cotizar.
12. Una persona puede ser militante aunque no se le haya entregado el carné.
13. Las actas de las reuniones se deben entregar en un plazo de 72 horas luego de la realización de la reunión.
14. Las reuniones pueden ser ordinarias o extraordinarias.
15. Las reuniones se deben planificar con antelación en el plan de trabajo del trimestre.

2.4. Requisitos de software

Un requisito de software es una característica que se debe exhibir en el software desarrollado o adaptado para solucionar un problema particular. A menudo los requisitos de sistemas de software se clasifican en funcionales y no funcionales (SOMMERVILLE, 2006):

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Definen los servicios que debe proporcionar el sistema, la forma en que éste debe reaccionar a ciertas entradas y cómo se debe comportar en situaciones particulares. En algunos casos, los requisitos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer (SOMMERVILLE, 2006).

Los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona en sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema (SOMMERVILLE, 2006).

La ingeniería de requisitos es el mecanismo que permite comprender lo que quiere el cliente, analizando necesidades, confirmando su viabilidad, negociando una solución razonable, describiendo la solución sin ambigüedad, validando la descripción y gestionando los requisitos para que se transformen en un sistema operacional (PRESSMAN, 2001).

2.4.1. Identificación de requisitos

Para la identificación de requisitos existen varias técnicas que permiten establecer una correcta comunicación con los interesados y su equipo de trabajo. En principio se pregunta al cliente, a los usuarios y a los que están involucrados en los objetivos del sistema o producto, luego se investiga cómo los sistemas o productos se ajustan a las necesidades del negocio, y finalmente, cómo el sistema o producto va a ser utilizado en el día a día (PRESSMAN, 2001).

A continuación se describen las técnicas utilizadas durante el proceso de desarrollo del software para recopilar los requisitos de la propuesta de solución:

- **Entrevistas:** se realizaron entrevistas al Secretario del Comité PCC-UCI encargado del funcionamiento de la esfera de la vida interna del PCC, lo que posibilitó la interacción con el experto en el área a tratar, para arribar en conjunto a la definición de los requisitos del sistema.
- **Sistemas existentes:** se realizó un análisis a distintos sistemas ya desarrollados que están relacionados con la gestión de organizaciones. Estos permitieron la obtención de ideas para la realización del sistema a desarrollar. Muchas de las funcionalidades presentes en estos sistemas sirvieron de base para la propuesta de solución.
- **Sesiones de tormentas de ideas (*brainstorming*):** se realizaron reuniones en grupo con la participación de las analistas, programadores, arquitectos e integrantes del proyecto con el objetivo de generar una gran variedad de vistas del problema y formularlo de diferentes formas.
- **Prototipos desechables:** la elaboración de los prototipos permitió mostrar las funcionalidades de la propuesta de solución para su validación por parte de los interesados. Este tipo de prototipo sirve para eliminar dudas sobre lo que realmente quiere el cliente además para desarrollar la interfaz que más le convenga al cliente.

A partir de la utilización de las técnicas anteriores se obtuvo un total de 195 requisitos funcionales y 16 no funcionales.

2.4.2. Requisitos funcionales

Para la descripción de los requisitos funcionales se utilizaron los artefactos propuestos por la metodología de desarrollo ágil DAC. En la Tabla 1 se muestran los requisitos funcionales que debe cumplir la solución para un total de 195 requisitos funcionales, los cuales se dividen de acuerdo a la prioridad para el cliente en 48 con prioridad Alta, 27 Media y 120 Baja. A partir de la complejidad para el desarrollar se dividen en 22 con complejidad Alta, 38 Media y 135 Baja.

Capítulo 2: Descripción de la propuesta de solución

Para determinar la complejidad y prioridad para el cliente de cada requisito funcional se tuvo en cuenta los criterios establecidos en el documento *SGU-SGO-010107_ER* del expediente del proyecto para la evaluación de requisitos de la metodología DAC.

La complejidad de los requisitos se utiliza para estimar el esfuerzo de implementación de este y planificar en qué iteración se implementará. Para determinar la complejidad de los requisitos se analizó el número de transacciones realizadas, la complejidad por interfaces de comunicación con actores, el número de requisitos no funcionales asociados, el tipo de tecnología y la reutilización de los elementos ya existentes.

En el caso de la prioridad se tuvieron en cuenta los beneficios para el cliente, la dependencia entre las funcionalidades, la estabilidad y la frecuencia con la que lo utilizará el cliente.

Tabla 1. Listado de requisitos funcionales de la propuesta de solución.

Nº	Nombre	Prioridad para el cliente	Complejidad
RF1	Listar estructuras de organización	Alta	Alta
RF2	Crear estructura de organización	Alta	Alta
RF3	Modificar estructura de organización	Media	Media
RF4	Ver detalles de estructura de organización	Media	Baja
RF5	Asociar áreas administrativas	Media	Media
RF6	Asociar responsabilidades a estructura	Alta	Alta
RF7	Activar estructuras inactivas	Media	Media
RF8	Exportar plantilla de personal en formato	Alta	Alta
RF9	Mostrar plantilla de responsabilidades	Alta	Media
RF10	Mostrar plantilla de personal	Alta	Media
RF11	Listar archivos existentes	Baja	Alta
RF12	Crear archivo	Alta	Alta
RF13	Modificar archivo	Media	Alta
RF14	Ver detalles de archivo	Baja	Alta
RF15	Exportar plantilla de responsabilidades	Baja	Baja
RF16	Listar militantes	Alta	Media
RF17	Modificar militante	Alta	Alta
RF18	Ver detalles de militante	Alta	Media
RF19	Asociar militante a estructura	Alta	Media
RF20	Ubicar militante en archivo	Alta	Media
RF21	Listar ingreso	Alta	Media
RF22	Iniciar ingreso	Alta	Alta
RF23	Actualizar datos de ingreso	Alta	Alta
RF24	Actualizar datos del militante	Alta	Alta
RF25	Ver detalles del ingreso del militante	Alta	Media

RF26	Listar incorporaciones	Alta	Media
RF27	Incorporar militante a estructura	Alta	Alta
RF28	Actualizar datos de incorporación	Alta	Alta
RF29	Ver detalles de la incorporación	Media	Media
RF30	Crear reunión	Baja	Baja
RF31	Exportar modelo de traslado	Media	Alta
RF32	Listar sanciones	Media	Baja
RF33	Listar visitantes	Alta	Baja
RF34	Crear visitante	Alta	Baja
RF35	Modificar visitante	Alta	Baja
RF36	Ver detalles de visitante	Alta	Baja
RF37	Listar desactivaciones	Media	Baja
RF38	Crear desactivación	Alta	Media
RF39	Actualizar datos de la desactivación	Alta	Media
RF40	Listar sanciones	Media	Baja
RF41	Crear sanción	Alta	Media
RF42	Actualizar datos de la sanción	Alta	Media
RF43	Ver detalles de la sanción	Media	Baja
RF44	Ver detalles de la desactivación	Media	Baja
RF45	Listar traslados	Media	Media
RF46	Crear traslado de militante	Media	Media
RF47	Detalles de traslado	Media	Media
RF48	Listar cotizaciones	Alta	Media
RF49	Generar cómputos de cotización	Alta	Media
RF50	Listar cotización por estructura	Alta	Media
RF51	Actualizar cómputo de cotización	Alta	Alta
RF52	Mostrar cotización por estructura base	Alta	Alta
RF53	Ver detalles de cotización	Media	Alta
RF54	Listar estructuras para cotización	Baja	Baja
RF55	Mostrar cotización por persona	Alta	Alta
RF56	Listar actas de reunión	Alta	Media
RF57	Crear acta de reunión	Alta	Media
RF58	Modificar acta de reunión	Media	Media
RF59	Ver detalles de acta de reunión	Baja	Baja
RF60	Exportar cómputo de actas de reunión	Alta	Alta
RF61	Listar temas de actividad	Media	Media
RF62	Crear tema de actividad	Media	Media
RF63	Modificar tema de actividad	Media	Media
RF64	Ver detalles de tema de actividad	Baja	Baja
RF65	Listar acuerdos de tema de actividad	Alta	Media

RF66	Crear acuerdo de tema de actividad	Alta	Baja
RF67	Modificar acuerdo de tema de actividad	Media	Baja
RF68	Ver detalles de acuerdo de tema de actividad	Baja	Baja
RF69	Listar planes de trabajo	Alta	Baja
RF70	Crear plan de trabajo	Alta	Media
RF71	Modificar plan de trabajo	Media	Media
RF72	Ver detalles de plan de trabajo	Baja	Baja
RF73	Modificar actividad de plan de trabajo	Media	Baja
RF74	Ver detalles de actividad de plan de trabajo	Baja	Baja
RF75	Crear actividad de plan de trabajo	Media	Media
RF76	Listar actividades de plan de trabajo	Media	Media
RF77	Mostrar actividades de plan de trabajo	Media	Alta
RF78	Asociar responsables a actividad	Media	Media
RF79	Listar estados del carné	Baja	Baja
RF80	Crear estado del carné	Baja	Baja
RF81	Modificar estado del carné	Baja	Baja
RF82	Ver detalles del estado del carné	Baja	Baja
RF83	Asociar configuración a organización	Alta	Alta
RF84	Listar tipo de reunión	Baja	Baja
RF85	Crear tipo de reunión	Baja	Baja
RF86	Modificar tipo de reunión	Baja	Baja
RF87	Ver detalles de tipo de reunión	Baja	Baja
RF88	Listar tipos de sanción	Baja	Baja
RF89	Crear tipo de sanción	Baja	Baja
RF90	Modificar tipo de sanción	Baja	Baja
RF91	Ver detalles de tipo de sanción	Baja	Baja
RF92	Listar tipos de desactivación	Baja	Baja
RF93	Crear tipo de desactivación	Baja	Baja
RF94	Modificar tipo de desactivación	Baja	Baja
RF95	Ver detalles de tipo de desactivación	Baja	Baja
RF96	Listar niveles escolares	Baja	Baja
RF97	Configurar niveles escolares	Baja	Baja
RF98	Listar clasificaciones laborales	Baja	Baja
RF99	Crear clasificación laboral	Baja	Baja
RF100	Modificar clasificación laboral	Baja	Baja
RF101	Ver detalles de clasificación laboral	Baja	Baja
RF102	Listar vías de ingreso	Baja	Baja
RF103	Crear vía de ingreso	Baja	Baja
RF104	Modificar vía de ingreso	Baja	Baja
RF105	Ver detalles de vía de ingreso	Baja	Baja

RF106	Listar formas de entrega del carné	Baja	Baja
RF107	Crear forma de entrega de carné	Baja	Baja
RF108	Modificar forma de entrega del carné	Baja	Baja
RF109	Ver detalles de forma de entrega del carné	Baja	Baja
RF110	Listar organizaciones para asociar configuraciones	Baja	Baja
RF111	Listar estados de cotización	Baja	Baja
RF112	Crear estado de cotización	Baja	Baja
RF113	Modificar estado de cotización	Baja	Baja
RF114	Ver detalles de estado de cotización	Baja	Baja
RF115	Listar tipos de períodos	Baja	Baja
RF116	Crear tipo de período	Baja	Baja
RF117	Modificar tipo de período	Baja	Baja
RF118	Ver detalles de tipo de período	Baja	Baja
RF119	Listar tipos de importe	Baja	Baja
RF120	Crear tipo de importe	Baja	Baja
RF121	Modificar tipo de importe	Baja	Baja
RF122	Ver detalles de tipo de importe	Baja	Baja
RF123	Listar causas de ausencias	Baja	Baja
RF124	Crear causa de ausencia	Baja	Baja
RF125	Modificar causa de ausencia	Baja	Baja
RF126	Ver detalles de causa de ausencia	Baja	Baja
RF127	Listar responsabilidades	Baja	Baja
RF128	Configurar responsabilidades	Alta	Media
RF129	Listar categorías de estructura	Baja	Baja
RF130	Configurar categoría de estructura	Alta	Media
RF131	Listar tipos de alta	Baja	Baja
RF132	Crear tipo de alta	Baja	Baja
RF133	Modificar tipo de alta	Baja	Baja
RF134	Ver detalles de tipo de alta	Baja	Baja
RF135	Listar tipo de trámite	Baja	Baja
RF136	Crear tipo de trámite	Baja	Baja
RF137	Modificar tipo de trámite	Baja	Baja
RF138	Ver detalles de tipo de trámite	Baja	Baja
RF139	Listar configuraciones de pago de cotización	Alta	Baja
RF140	Crear configuración pago de cotización	Alta	Baja
RF141	Modificar configuración pago de cotización	Alta	Baja
RF142	Ver detalles de configuración pago de cotización	Alta	Baja
RF143	Listar formas de pago	Baja	Baja
RF144	Crear forma de pago	Baja	Baja
RF145	Modificar forma de pago	Baja	Baja

RF146	Ver detalles de forma de pago	Baja	Baja
RF147	Listar período de pago	Baja	Baja
RF148	Crear período de pago	Baja	Baja
RF149	Modificar período de pago	Baja	Baja
RF150	Ver detalles de período de pago	Baja	Baja
RF151	Listar motivos de traslado	Baja	Baja
RF152	Crear motivos de traslado	Baja	Baja
RF153	Modificar motivo de traslado	Baja	Baja
RF154	Ver detalles de motivo de traslado	Baja	Baja
RF155	Listar tipos de traslado	Baja	Baja
RF156	Crear tipo de traslado	Baja	Baja
RF157	Modificar tipo de traslado	Baja	Baja
RF158	Ver detalles de tipo de traslado	Baja	Baja
RF159	Listar lugares de traslado	Baja	Baja
RF160	Crear lugar traslado	Baja	Baja
RF161	Modificar lugar de traslado	Baja	Baja
RF162	Ver detalles de lugar de traslado	Baja	Baja
RF163	Listar causa de no pago	Baja	Baja
RF164	Crear causa de no pago	Baja	Baja
RF165	Modificar causa de no pago	Baja	Baja
RF166	Ver detalles de causa de no pago	Baja	Baja
RF167	Listar conceptos de pago	Baja	Baja
RF168	Crear concepto de pago	Baja	Baja
RF169	Modificar concepto de pago	Baja	Baja
RF170	Ver detalles de concepto de pago	Baja	Baja
RF171	Listar motivo de desactivación	Baja	Baja
RF172	Crear motivo de desactivación	Baja	Baja
RF173	Modificar motivo de desactivación	Baja	Baja
RF174	Ver detalles de motivo de desactivación	Baja	Baja
RF175	Listar motivo de sanción	Baja	Baja
RF176	Crear motivo de sanción	Baja	Baja
RF177	Modificar motivo de sanción	Baja	Baja
RF178	Ver detalles de motivo de sanción	Baja	Baja
RF179	Listar tipos de acta	Baja	Baja
RF180	Crear tipo de acta	Baja	Baja
RF181	Modificar tipo de acta	Baja	Baja
RF182	Ver detalles de tipo de acta	Baja	Baja
RF183	Listar tipos de actividades de plan de trabajo	Baja	Baja
RF184	Crear tipo de actividad de plan de trabajo	Baja	Baja
RF185	Modificar tipo de actividad de plan de trabajo	Baja	Baja

RF186	Ver detalles de tipo de actividad de plan de trabajo	Baja	Baja
RF187	Listar roles de actividad de plan de trabajo	Baja	Baja
RF188	Crear un rol de actividad de plan de trabajo	Baja	Baja
RF189	Modificar rol de actividad de plan de trabajo	Baja	Baja
RF190	Ver detalles de rol de actividad de plan de trabajo	Baja	Baja
RF191	Listar estados de cumplimiento de acuerdos	Baja	Baja
RF192	Crear estado de cumplimiento de acuerdo	Baja	Media
RF193	Modificar estado de cumplimiento de acuerdo	Baja	Baja
RF194	Ver detalles de estado de cumplimiento de acuerdo	Baja	Baja
RF195	Asociar tipos de alta a tipo de trámite	Baja	Baja

2.4.3. Requisitos no funcionales

El sistema propuesto contará con los siguientes requisitos no funcionales:

Usabilidad

1. El sistema debe presentar un menú lateral y una barra de íconos flotantes que permitan el acceso rápido a la información por parte de los usuarios.
2. Solo se mostrarán a los usuarios aquellas acciones o informaciones a las que por su responsabilidad o rol dentro del negocio necesitan acceder mostrando en la vista mediante íconos y menús el acceso a la misma.
3. Las vistas del sistema deben indicar en cada momento la acción que se está realizando así como los íconos deben estar representados por una imagen acorde a la acción que se realiza mediante el mismo.

Software

4. Para el despliegue del sistema se debe contar en el servidor de bases de datos con *PostgreSQL* 9.4, bajo el sistema operativo CentOS 6.6 o superior.
5. Para el despliegue del sistema se debe contar en el servidor de aplicaciones web con: PHP v 5.6 con las bibliotecas php5-ldap, php5-gd, php5-mcrypt, php5-pgsql, php5-xsl, php5-*openssl*; Apache 2.2 con el módulo *rewrite* activado y JRE[18] 6 o superior.

Confiabilidad

6. El sistema puede permanecer inactivo durante 10 minutos. Al cumplirse este término se cerrará la sesión teniendo que autenticarse el usuario nuevamente.

[18] *Java Runtime Environment* o JRE es un conjunto de utilidades que permite la ejecución de programas Java.

7. Cuando ocurre una excepción el sistema mostrará un mensaje explicativo del error ocurrido y permanecerá en el mismo estado sin realizar ninguna otra operación.

Eficiencia

8. El sistema soportará la conexión simultánea de todos los posibles usuarios habituales un promedio de 25 y un máximo de 600.
9. El sistema deberá tener por cada transacción un tiempo de respuesta promedio de 1,5 segundos y un máximo de 3 segundos.

Soporte

10. El sistema cumplirá con las normas de codificación, conversiones para nomenclatura, bibliotecas de clase definidas para el Sistema de Gestión Universitaria en el documento *SGU-NUC-010217_EC.*
11. El sistema contará con toda la documentación definida en el expediente del proyecto asociada a su proceso de desarrollo para las actividades de soporte.

Restricciones de diseño

12. El sistema estará desarrollado con las herramientas definidas para el Sistema de Gestión Universitaria en el documento *SGU-NUC-010216_8_ATI.*

Interfaz de comunicación

13. La comunicación entre el cliente y el servidor de aplicaciones se lleva a través del protocolo HTTPS[19].
14. La autenticación del sistema se podrá realizar mediante el protocolo LDAP para la consulta del directorio activo o mediante el protocolo SOAP para el consumo del servicio de pasarela.
15. La comunicación entre el servidor de aplicaciones y la base de datos se lleva a través del protocolo TCP.

Hardware

16. Para la ejecución del sistema se requiere que la PC cliente tenga los siguientes componentes de hardware: Pentium 4 o superior, 512 MB RAM y 200 MB disco duro disponible como mínimo.

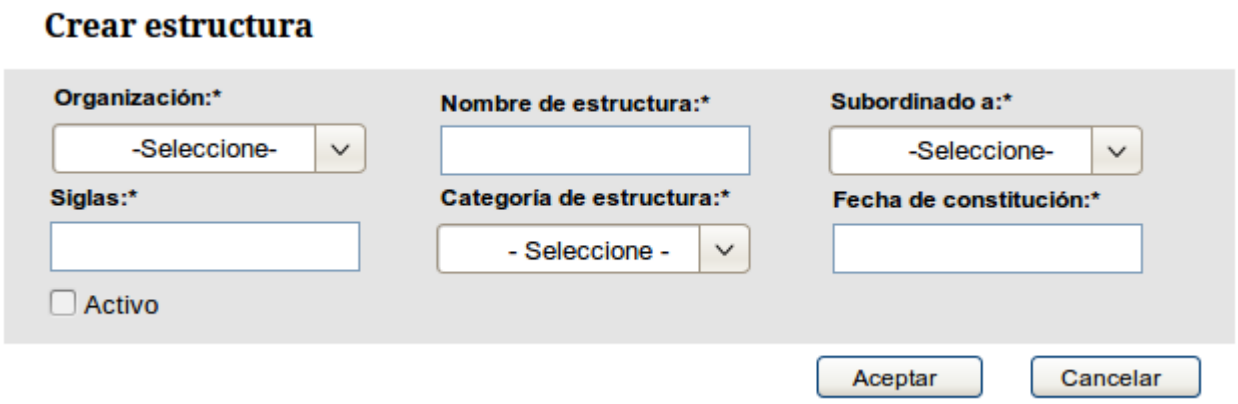
[19] **HTTPS:** (*Hypertext Transfer Protocol Secure,* Protocolo Seguro de Transferencia de Hipertexto) es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto, es decir, es la versión segura de HTTP.

2.4.4. Descripción de requisitos

Todos los requisitos se describieron utilizando escenarios para mostrar el flujo de la funcionalidad y prototipos. A continuación se presenta como ejemplo la descripción del requisito “Crear estructura de la organización”, el resto de las descripciones realizadas se encuentran en el expediente del proyecto.

Tabla 2. Descripción del requisito "Crear estructura de la organización".

Código	**Nombre**	**Descripción**	**Complejidad**	**Prioridad para cliente**
RF2	Crear estructura de la organización**.**	El requisito permite crear una nueva estructura de organizaciones. El administrador selecciona el componente “Estructura” del “Sistema de Gestión de Organizaciones”, luego en el menú lateral la agrupación funcional “Estructura” y la funcionalidad “Estructura”. En el área de íconos flotantes selecciona la acción Crear estructura. Se muestran todos los datos a llenar. Además de la opción Listar en el área de íconos flotantes.	Alta	Alta
Actor		**Precondición**	**Poscondición**	
Administrador		El usuario debe estar autenticado en el sistema con los permisos necesarios para acceder a la funcionalidad. No debe existir una estructura con el mismo nombre en una misma organización.		
Prototipo				



Botones e hipervínculos			
Nº	**Nombre**	**Descripción de flujo básico**	**Descripción de flujo alterno**
1	Botón Aceptar	Verificar que los campos obligatorios estén introducidos. Verificar que los datos introducidos tengan el formato correcto. Consultar si existe una Estructura con los mismos datos. El sistema no permitirá escribir más de los caracteres permitidos en un campo. El sistema no permitirá escribir caracteres no válidos en el campo fecha. Insertar una Estructura. Mostrar mensaje de confirmación: "El elemento ha sido creado satisfactoriamente". Se mantiene en la interfaz de crear estructuras.	Si faltan campos obligatorios por llenar muestra el mensaje en rojo: *"campo requerido"* sobre el campo que debe ser llenado de forma obligatoria. Si existe una estructura con el mismo nombre muestra el mensaje: "*El elemento ya existe".* Si no introduce la cantidad mínima necesaria de caracteres en un campo muestra en rojo el mensaje: *"Entre al menos (cantidad de caracteres requeridos) caracteres"* sobre el campo requerido. Si introduce más caracteres por palabra de las requeridas en un campo muestra el mensaje en rojo: *"Ha excedido el número de letras permitidas para una palabra" sobre el campo requerido.* Si introduce un nombre con formato incorrecto se muestra el mensaje: *"Entre*

			solo letras, números y espacio o guion bajo entre palabras" sobre el campo nombre. Si introduce una fecha en formato incorrecto muestra el mensaje en rojo: *"Entre una fecha válida"* sobre el campo fecha.
2	Botón Cancelar	Muestra un mensaje de confirmación: *"¿Está seguro de realizar la acción?".*	Si no acepta se queda en la misma Interfaz.
Comportamiento adicional			
Elemento		**Nombre**	**Descripción del comportamiento**
Lista de selección		Tipo de organización	Obligatorio. Selección. Permite seleccionar la organización a la que pertenece la estructura.
Lista de selección		Categoría de estructura	Obligatorio. Selección. Permite seleccionar el tipo de unidad organizativa de la estructura.
Lista de selección		Subordinado a	Obligatorio. Selección. Permite seleccionar a que unidad organizativa va a estar subordinada la estructura.
Campos de entrada de datos		Nombre de estructura	Obligatorio. Único. Admite entre 2 y 100 caracteres. Solo admite 30 caracteres por palabra. Solo admite letras, números y espacio o guion bajo entre palabras.
Campos de entrada de datos		Fecha de constitución	Obligatorio. Formato (dd/mm/aaaa).

		Admite solo caracteres numéricos separados por *backslash* (/).
Campos de entrada de datos	Siglas	Obligatorio. Único. Admite entre 2 y 10 caracteres. Solo admite letras, números y espacio o guion bajo entre palabras.
Entidades		
Estructura		

2.5. Descripción del sistema propuesto

Se propone la implementación del Sistema de Gestión de Organizaciones (SGO) de la Universidad de las Ciencias Informáticas que se integre al SGU, utilizando las funciones y servicios brindados por otros sistemas y componentes dentro del mismo. El sistema cuenta con cuatro componentes: Estructura, Movimiento, Funcionamiento y Configuración.

La solución propuesta garantiza una correcta gestión de los procesos de la vida interna del PCC. El SGO gestiona la estructura y composición de las organizaciones, los militantes con sus respectivas responsabilidades en las estructuras, la realización de trámites como son de ingresos, incorporaciones, traslados, sanciones y desactivaciones, así como la entrega de la cotización y el registro de actas de reuniones. Además permite la búsqueda en expedientes físicos de forma rápida y exacta. Los usuarios que acceden tienen distintas funciones según el rol asignado**,** en dependencia de su responsabilidad en la organización, garantizando el correcto acceso al sistema de las personas involucradas.

El componente Estructura realiza la gestión de las estructuras de la organización, teniendo en cuenta la jerarquía de estas, permitiendo asociar las áreas administrativas que las componen, los militantes que atienden los núcleos y la cantidad de militantes que ocupan cada responsabilidad en las estructuras. Este componente permite crear los archivos virtuales donde se almacenan los expedientes y muestra la disponibilidad de espacio en cada archivo. También permite consultar las plantillas del personal y de responsabilidades así como exportar estos reportes en formato PDF o XLS.

Movimiento por otra parte contiene las funcionalidades necesarias para realizar los trámites de ingreso, incorporación, traslado, desactivación y sanción, manteniendo un registro de todos los trámites realizados a los militantes. Permite gestionar la información del militante, la responsabilidad que ocupa en las

estructuras a las que pertenece y la ubicación de su expediente físico en el archivo. Esta opción permite determinar la ubicación física del expediente del militante en el archivo garantizando que no aparezcan números de expedientes repetidos o mal asignados.

En Funcionamiento se controla el estado y cumplimiento de la cotización a todos los niveles de la organización, tratando con el salario real de los militantes para el cálculo del monto a pagar. Permite realizar el registro de las actas de las reuniones para la realización de reportes estadísticos de los temas discutidos, asistencia y acuerdos tomados. También gestiona los planes de trabajo trimestrales permitiendo controlar el cumplimiento de las actividades y la incorporación de nuevas actividades manteniéndolo actualizado.

Con el componente Configuración se logra la gestión de los elementos de configuración general del sistema, permitiendo realizar modificaciones en caso de ser necesario.

El SGO responde a la necesidad de informatización de los procesos de la organización para un correcto funcionamiento y centralización de la información. Se facilitan las búsquedas de información para la confección de reportes que con frecuencia son solicitados al PCC, brinda una manera viable de realizar consultas para obtener la información que se necesita durante cualquier proceso de la vida interna de la organización y permite consultar el estado de los trámites realizados a los militantes.

Para un mayor entendimiento de la ubicación y distribución de las funcionalidades de la propuesta de solución consultar el **Anexo 3:** Mapa de navegación del Sistema de Gestión de Organizaciones.

2.6. Descripción de la arquitectura y diseño

La propuesta de solución está concebida como un sistema dentro del SGU, por esta razón se ajustará a la arquitectura definida para dicho sistema, la cual propone la arquitectura Cliente-Servidor, haciendo uso del patrón Modelo-Vista-Controlador (MVC), ya que el marco de trabajo empleado está basado en dicho patrón, por su flexibilidad y ventajas.

2.6.1. Arquitectura Cliente-Servidor

La arquitectura Cliente-Servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes (Ver Figura 1). Un cliente hace una petición de un servicio y recibe la respuesta a dicha petición; un servidor recibe y procesa la petición y devuelve la respuesta solicitada. Los clientes y servidores se

comunican a través de una red de comunicaciones para realizar una o varias tareas de forma conjunta (KIOSKEA, 2015).

Esta arquitectura facilita la integración entre sistemas diferentes y comparte información, permitiendo por ejemplo que las máquinas ya existentes puedan ser utilizadas pero utilizando interfaces más amenas para el usuario (KIOSKEA, 2015).

El sistema se hospedará en el servidor web de aplicaciones Apache 2.4.7 instalado en uno de los servidores de nodo central de la universidad y se podrá acceder mediante la dirección URL https://gestionuniversitaria.uci.cu/. El uso de esta arquitectura permite que los usuarios puedan acceder al sistema desde cualquier dispositivo conectado a la red UCI logrando centralizar la información y facilitar el acceso a esta, sin necesidad de que todos tengan que utilizar el mismo sistema operativo.

Figura 1. Arquitectura Cliente-Servidor.

2.6.2. Patrón de arquitectura

El patrón Modelo-Vista-Controlador (MVC) es el patrón de desarrollo en el que se basa el marco de trabajo Codelgniter, el cual es utilizado para el desarrollo del sistema. El MVC es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario y el componente encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, es decir, por un lado define componentes para la representación de la información y por el otro la interacción del usuario (CODEIGNITER, 2015).

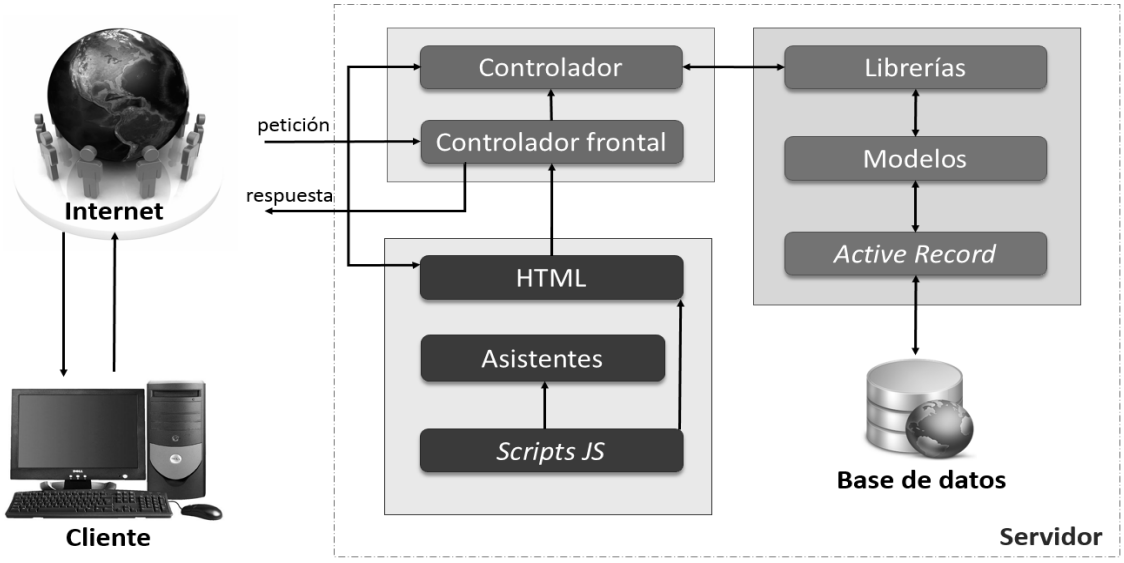


Figura 2. Patrón Modelo-Vista-Controlador.

Implementación del MVC que realiza GUUD

La Figura 2 ilustra la particularidad del MVC implementado por el GUUD y que a continuación se describe. El marco de trabajo GUUD cuenta con un controlador frontal que inicializa los recursos básicos necesarios para usar el Codelgniter (parte estructural del GUUD). Una vez realizada una petición HTTP por un cliente, el controlador frontal se encarga de analizar la URI[20] y a partir de esta determina cuál controlador de aplicación (controlador de un determinado componente) debe ser cargado para atender la petición realizada. Cada controlador de aplicación tiene asociada una o varias librerías responsables de procesar los datos e implementar la lógica del negocio inherente a las acciones relacionadas con dicho controlador. De manera similar cada librería tiene asociado uno o varios modelos encargados del acceso a los datos.

Cuando un controlador de aplicación es cargado, este examina la petición para determinar si solo debe cargar una vista determinada o si es necesario interactuar con la base de datos. En este último caso el controlador de aplicación envía los datos recibidos a la o las librerías. Estas a su vez cargan los modelos necesarios para obtener, registrar o actualizar en la base de datos la información solicitada o enviada.

[20] Un *Uniform Resource Identifier* o URI (en español, Identificador Uniforme de Recurso), cadena de caracteres corta que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico, enciclopedia, etcétera). Su principal diferencia con el *Uniform Resource Identifier* o URL (en español, Localizador Uniforme de Recurso) es que permite especificar que parte del recurso se solicita (segmentos).

Para realizar esta tarea los modelos hacen uso de la clase *Active Record*. Esta clase permite consultar la base de datos con mínima codificación y abstrayéndose del gestor que se use. La sintaxis de la consulta es generada por cada adaptador de base de datos. Cuando los datos son obtenidos, se retornan al controlador de aplicación en un proceso inverso al descrito anteriormente. Posteriormente, el controlador carga estos datos a archivos escritos en HTML los cuales pueden incluir llamadas a archivos escritos en JavaScript para manejar dinámicamente su contenido o hacer uso de asistentes (*helpers*) para la creación de forma simplificada de código HTML. Finalmente, el resultado obtenido de todo este proceso es enviado al navegador web como respuesta a la petición inicial (MARTIN SOLER, 2013).

2.6.3. Patrones de diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño, o sus relaciones, con las que construir sistemas de software.

Patrones de diseño GRASP

Los patrones GRASP[21] representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. El uso de los patrones GRASP es fundamental en el diseño del software orientado a objeto, debido a que brindan una solución a varios de los problemas que se pueden presentar en la programación (SAAVEDRA, 2007).

Para el diseño de la propuesta de solución se tuvo en cuenta el uso de los patrones GRASP implementados por el marco de trabajo utilizado por el sistema. A continuación se realiza una descripción de los patrones utilizados:

- **Experto**: se encarga de asignar una responsabilidad al experto en información, o sea, aquella clase que cuenta con la información necesaria para cumplir la responsabilidad. Este patrón se evidencia en las clases librerías del sistema, las cuales cuentan con la información necesaria para cumplir la responsabilidad sobre los elementos del negocio. En la Figura 3 se muestra un ejemplo de la utilización del patrón en la librería *Archivo_lib* donde se maneja la información referente a la tabla *tb_darchivo.*

[21] *General Responsibility Assignment Software Patterns*, Patrones Generales de Software para Asignar Responsabilidades.

```php
<?php
class Archivo_lib {

    private $_ci;

    public function __construct() {
        $this->_ci = & get_instance();
        $this->_ci->load->model('tb_darchivo_mdl');
        $this->_ci->load->model('tb_rarchivo_militante_mdl');
    }

    public function obtenerCantidadArchivo($filtros) {...3 lines }

    public function obtenerArchivoPorPagina($inicio = null, $limite = null, $elementoOrdenar = null,

    public function obtenerArchivoDadoId($id) {...11 lines }

    public function modificarArchivo($datos) {...8 lines }
```

Figura 3. Patrón Experto.

- **Creador**: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. Se evidencia en la clase *loader* que es el objeto *load* de las clases controladoras. Esta clase se encarga de cargar los elementos del marco de trabajo como librerías y modelos que se utilizaran para completar la petición. A continuación se muestra un ejemplo de su uso en la clase controladora *Archivo* donde se utiliza en el constructor para cargar la librería A*rchivo_lib*.

```php
<?php
class Archivo extends MY_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->library('archivo_lib');

    }
```

Figura 4. Patrón Creador.

- **Alta cohesión y Bajo acoplamiento**: la propia implementación del CodeIgniter contiene estos patrones nivelados, permitiendo el uso de los componentes de forma individual en la implementación de la solución de software que se desarrolla, evidenciando de esta forma el bajo acoplamiento, así como la dependencia entre los componentes o la alta cohesión.

- **Controlador**: asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase. El patrón controlador se refleja en las clases controladoras pertenecientes al sistema

(Ver Figura 5), que son las clases que se encargan de obtener datos y enviarlos a las librerías y las vistas.

```php
<?php

class Archivo extends MY_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->library('archivo_lib');

    }

    public function index() {
        echo $this->template->render('archivo/listar_view');
    }

    public function listar() {
        echo $this->template->render('archivo/listar_archivo_view');
    }

    public function obtenerArchivo() {...9 lines }
```

Figura 5. Patrón Controlador.

Patrones de diseño GoF

Los patrones GoF (*Gang of Four*) se clasifican en tres categorías basadas en su propósito: creacionales, estructurales y de comportamiento. Los patrones creacionales abstraen el proceso de creación de instancias y ocultan los detalles de cómo los objetos son creados o inicializados, mientras que los patrones estructurales se ocupan de cómo las clases y objetos se combinan para formar grandes estructuras y proporcionar nuevas funcionalidades. Por otra parte, los patrones de comportamiento están relacionados con los algoritmos y la asignación de responsabilidades entre los objetos. Son utilizados para organizar, manejar y combinar comportamientos (MARQUEZ, 2015).

- **Instancia única (*Singleton*):** garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Existen muchas clases para las cuales es importante tener únicamente una sola instancia que pueda ser utilizada en muchas partes diferentes del sistema. Este patrón se refleja en las clases controladoras que son instancias únicas y en la IoC para la interacción entre componentes. La Figura 6 ejemplifica el uso del patrón en la librería *Archivo_lib* cuando se crea el objeto *_ci* y se usa el método *get_instance* para gestionar si se ha de crear o no una instancia del objeto, de forma que si ya existe devuelve la instancia existente.

```php
<?php

class Archivo_lib {

    private $_ci;

    public function __construct() {
        $this->_ci = & get_instance();
        $this->_ci->load->model('tb_darchivo_mdl');
        $this->_ci->load->model('tb_rarchivo_militante_mdl');
    }
```

Figura 6. Patrón Instancia única

- **Mediador** (*Mediator*): define un objeto que coordine la comunicación entre objetos de distintas clases, pero que funcionan como un conjunto. Se refleja en las librerías, que funcionan como mediadoras entre las clases controladoras y las modelos o acceso a datos (Ver Figura 8).

```php
<?php
class Archivo_lib {

    private $_ci;

    public function __construct() {
        $this->_ci = & get_instance();
        $this->_ci->load->model('tb_darchivo_mdl');
        $this->_ci->load->model('tb_rarchivo_militante_mdl');
    }

    public function obtenerCantidadArchivo($filtros) {...3 lines }

    public function obtenerArchivoPorPagina($inicio = null, $limite = null, $elementoOrdenar = null,

    public function obtenerArchivoDadoId($id) {...11 lines }

    public function modificarArchivo($datos) {...8 lines }
```

Figura 7. Patrón Mediador.

- **Observador** (*Observer*): define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. Se refleja en la clase *loader* que es el objeto *load* de las clases controladoras (Ver Figura 8), cuya función es cargar los elementos del marco de trabajo dígase librerías, modelos y se encarga de actualizar la controladora instanciada.

```php
<?php

class Archivo extends MY_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->library('archivo_lib');

    }

    public function obtenerArchivo() {
        $filtro = array();
        $datos = $this->input->all_post(TRUE);
        if (isset($datos['cadena'])) {
            $filtro['cadena'] = $datos['cadena'];
        }
        $this->load->helper('grid');
        echo grid_json($this->archivo_lib, 'obtenerCantidadArchivo', 'obtenerArchivoPorPagina',
    }
```

Figura 8. Patrón Observador.

2.7. Modelo de datos

Los modelos de datos son esenciales para el desarrollo de sistemas de información, ya que a través de ellos puede conseguirse la compatibilidad necesaria para manejar grandes cantidades de datos. El modelo de datos de la propuesta de solución cuenta con un total de 71 tablas agrupadas en la base de datos *BD_sgu_organizaciones.* En la Figura 9 y 10 se presentan los modelos de datos físicos utilizados en la implementación de los componentes Estructura, Movimientos y Configuración, implementados en la primera versión del sistema.

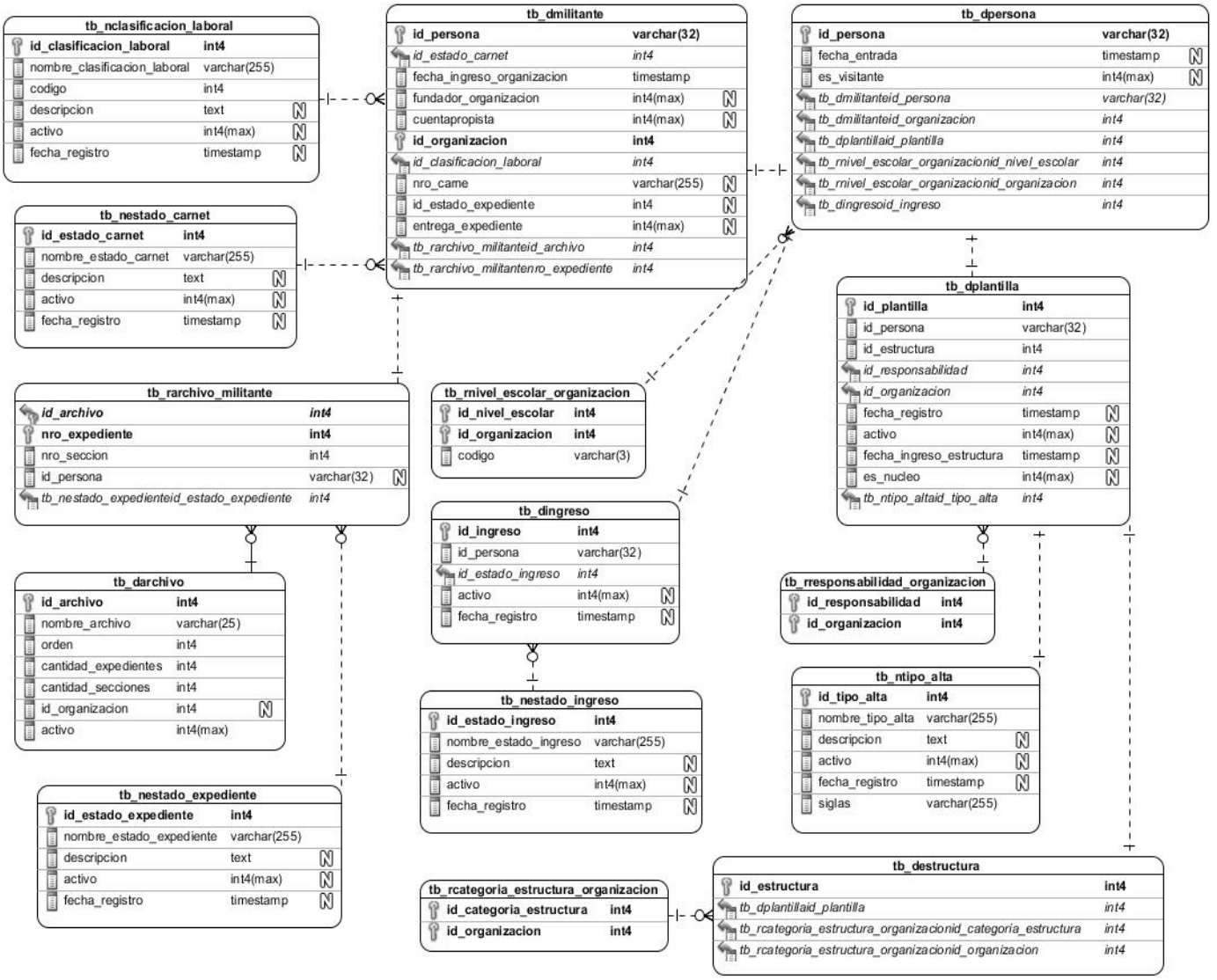


Figura 9. Modelo de datos físicos del esquema *sq_organizaciones*.

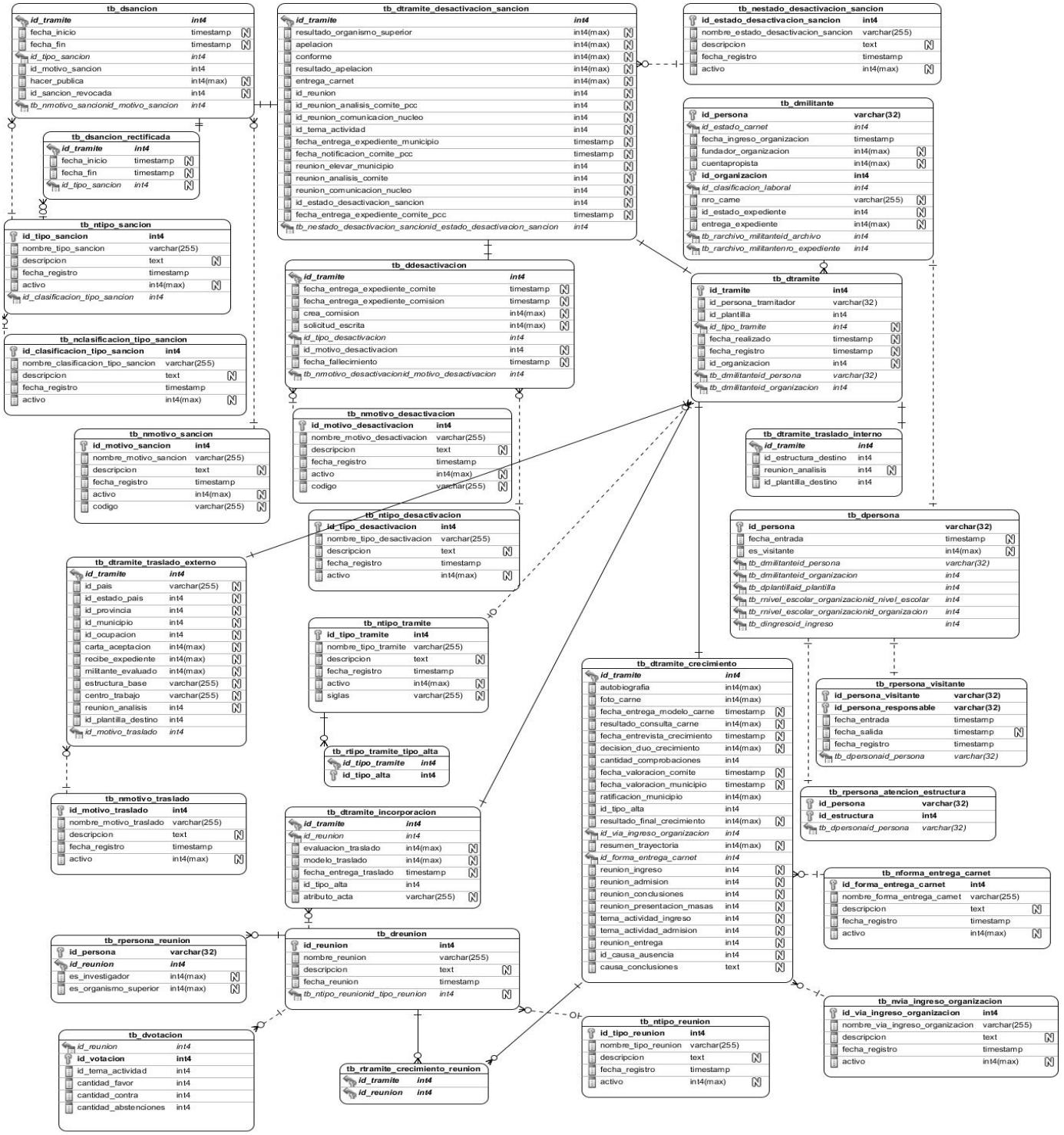


Figura 10. Modelo de datos físicos del esquema *sq_movimientos*.

2.8. Diagrama de despliegue

Un diagrama de despliegue muestra cómo se configuran las instancias de los componentes y los procesos para la ejecución en las instancias de los nodos de proceso (LARMAN, 1999).

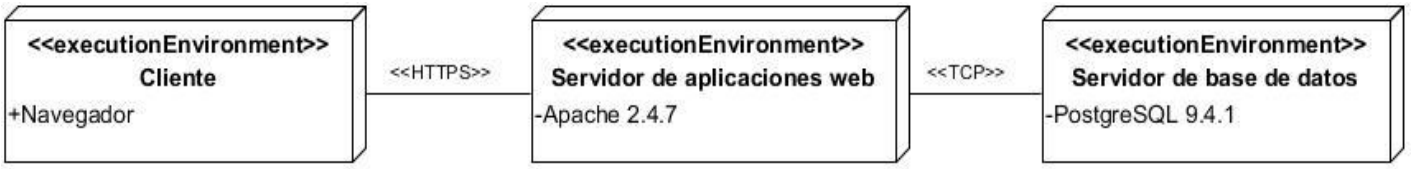


Figura 11. Diagrama de despliegue de la propuesta de solución.

El diagrama de despliegue representado muestra la siguiente distribución:

Cliente: dispositivo cliente capaz de conectarse al servidor de aplicaciones mediante el protocolo de comunicaciones HTTPS.

Servidor de aplicaciones web: computadora en que se encuentra el servidor web Apache, este será el lugar en que se gestione todo el contenido de la aplicación. El mismo establecerá comunicación con los ordenadores clientes mediante protocolo HTTPS y con el servidor de base de datos por medio del protocolo TCP.

Servidor de base de datos: ordenador en que se encuentra el gestor de base de datos PostgreSQL capaz de mantener persistente la información generada y a utilizar.

2.9. Conclusiones parciales

El desarrollo del presente capítulo permitió sentar las bases para la construcción del sistema propuesto a partir del modelado de la propuesta de solución, del análisis y descripción de los procesos de negocio que propició la obtención de las principales funcionalidades que debe cumplir la aplicación y quedaron manifestadas en la descripción de requisitos de software. Además, se logró una visión clara y objetiva de los requisitos del cliente. Con la caracterización de la arquitectura Cliente-Servidor, el patrón MVC, los patrones de diseño, así como la obtención del modelo de datos y la distribución física de la propuesta de solución, se logró comprender toda la arquitectura para el desarrollo de la propuesta de solución.

Capítulo 3: Implementación y evaluación de la solución

3.1. Introducción

Entre las fases que constituyen el proceso de desarrollo se encuentran las de implementación y pruebas. En este capítulo se describen los estándares de codificación utilizados para la implementación de las descripciones de requisitos elaboradas, posibilitando la legibilidad y el mantenimiento del código en lo adelante. Además se detalla la estrategia de pruebas a efectuar a la propuesta de solución para su correcta evaluación y se analizan los resultados obtenidos.

3.2. Integración con otros componentes del Sistema de Gestión Universitaria

El SGO, se encuentra integrado al SGU, haciendo uso de las facilidades provistas para la integración, así como la reutilización de varios componentes ya implementados por otros sistemas integrantes. Esto permite la uniformidad en la arquitectura, la centralización de archivos e información y la eliminación de redundancias en el código, puesto que cualquier componente en cualquier sistema puede utilizar las funcionalidades y componentes implementados por los demás.

SGO utiliza los métodos y servicios brindados por otros sistemas y componentes dentro del SGU, entre los que se encuentran los componentes horizontales:

Seguridad: el componente de seguridad, se implementa a nivel central para todo el Sistema de Gestión Universitaria, garantiza el acceso a la información dados los niveles de privilegio de cada usuario, haciendo uso de la arquitectura sobre la cual está desarrollado el sistema. Esta forma de implementación de la seguridad controla no solo el acceso a un componente determinado, sino también a cada sistema que integra el SGU.

Personal: el componente personal permite la obtención de toda la información referente a las personas, necesaria para la gestión de la militancia.

Estructura y composición: gestiona la información referente a toda la estructura administrativa y la jerarquía de la universidad y se utiliza para la gestión de las estructuras del PCC, las áreas administrativas asociadas a las estructuras y las responsabilidades asignadas a cada militante en la estructura a la que pertenece.

3.3. Estándar de codificación

Para el desarrollo del componente se aplican los estándares de codificación establecidos para el desarrollo del Sistema de Gestión Universitaria, con el propósito de estandarizar las nomenclaturas en la implementación de la solución de software que se implementa y obtener un producto estable.

3.3.1. Indentación, llaves de apertura y cierre y tamaño de las líneas

Se debe usar una indentación sin tabulaciones, con un equivalente a 4 espacios. El uso de las llaves "{}" será en una nueva línea. La longitud de las líneas de código es aproximadamente de 75 a 80 caracteres para mantener la legibilidad del código. En la Figura 8 se muestra un ejemplo de cómo debe quedar el formato del código de acuerdo con el estándar utilizado.

```
public function listar()
{
    echo $this->template->render('archivo/listar_archivo_view');
}
```

Figura 12. Indentación, llaves de apertura y cierre y tamaño de las líneas.

3.3.2. Convención de nomenclatura

Variables: se rigen por la nomenclatura **camelCase**[22]. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula (Ver Figura 9).

```
$organizaciones = null;
$idOrganizacion = null;
```

Figura 13. Variables.

Clases: siempre comienzan con mayúscula, en caso de nombre compuesto las palabras se separan con guion bajo "_" y el resto en minúscula como se muestra a continuación:

```
class Estructura_organizaciones extends MY_Controller
class Archivo extends MY_Controller
```

Figura 14. Clases.

Funciones: se rigen por la nomenclatura **camelCase**. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa. La Figura 14 muestra un ejemplo de su uso.

[22] ***CamelCase*** es un estilo de escritura que se aplica a frases o palabras compuestas. En este caso se usa el *lowerCamelCase*, cuando la primera letra de cada una de las palabras es mayúscula con la excepción de que la primera letra es minúscula. Ejemplo: *ejemploDeLowerCamelCase*

```
public function listar()
{...3 lines }

public function obtenerArchivo()
{...9 lines }
```

Figura 15. Funciones.

Ficheros: el nombre del fichero debe ser siempre en minúscula y en caso de nombres compuestos se usa el guion bajo, seguido del sufijo definido para cada tipo de fichero.

- **Vistas**: intuitivo y relacionado con el formulario y/o vista que representa. Sufijo **"_view".**
  - Ejemplo: registrar_archivo_view
- **Modelos**: mismo nombre de la tabla de la base de datos para la cual se crea. Sufijo **"_mdl".**
  - Ejemplo: tb_darchivo_mdl
- **Librerías**: mismo nombre de la clase que representa. Sufijo **"_lib"**.
  - Ejemplo: archivo_lib
- **Controladoras**: mismo nombre de la clase que representa.
  - Ejemplo: archivo

3.3.3. Estructuras de control

Las estructuras de control incluyen **if, for, foreach, while, switch**. Entre las estructuras de control y los paréntesis debe de existir un espacio. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Con esto se aumenta la legibilidad del código y se disminuye la probabilidad de errores lógicos. A continuación se muestra un ejemplo de su uso en la clase *Estructura_organizaciones.*

```
foreach ($organizaciones as $value)
{
    $cantidad = count($rama);
    for ($index = 0; $index < $cantidad; $index++)
    {
        if ($rama[$index] === $value->id_estructura)
        {
            $idOrganizacion = $rama[$index];
            break;
        }
    }
    if (!empty($idOrganizacion))
    {
        break;
    }
}
```

Figura 16. Estructuras de control.

3.3.4. Documentación

Todos los archivos deben de tener la documentación asociada al mismo. Para esto debe de cumplir con el siguiente bloque al principio de cada clase:

```
/**
*@category Modelo
*@package Archivo
*@author Isidro Montero Reyes imontero@estudiantes.uci.cu
*/
```

Figura 17. Documentación de una clase.

3.3.5. Buenas prácticas

Los valores lógicos y nulos siempre se escriben con mayúscula, para facilitar la legibilidad del código. Se debe usar un cambio de línea antes de las estructuras de control y definición de las funciones.

```
$archivo = NULL;
$existeArchivo = FALSE;
```

Figura 18. Buenas prácticas.

3.4. Estándares del diseño

El diseño del SGO se realizó siguiendo las pautas de diseño establecidas en el manual de directrices del SGU recogidas en el documento *SGU-NUC-010216_3a_AP* del expediente del proyecto, posibilitando la uniformidad en la estructura de todas las páginas web que componen el sistema.

3.5. Estrategias de validación de requisitos

La validación de requisitos examina las descripciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto (PRESSMAN, 2001).

3.5.1. Criterios de validación de requisitos

El proceso de desarrollo de software usando la metodología DAC propone criterios para la validación de los requisitos del cliente y del producto, donde se responden varias interrogantes que determinan si el requisito es aprobado o no contenidas en el documento *SGU-SGO-010105_CVRC* del expediente del proyecto.

Interrogantes para validar los requisitos del cliente

¿El proveedor del requisito es un proveedor válido?

¿El requisito está identificado como único?

¿El requisito es modificable?

¿El requisito no es ambiguo?

¿El requisito está completo?

¿El requisito es congruente con otros requisitos relacionados?

¿El requisito puede ser implementado?

¿El requisito puede ser probado?

¿El resultado de la evaluación de impacto es positivo?

¿El requisito está correcto?

3.5.2. Validación de requisitos

Para la validación de los requisitos se utilizaron las siguientes técnicas:

- **Revisión técnica formal de requisitos:** se realizan reuniones donde se examinan las descripciones del sistema buscando errores en el contenido o en la interpretación, ideas donde se necesitan aclaraciones, información incompleta, inconsistencias, requisitos contradictorios o requisitos imposibles o inalcanzables.
- **Construcción de prototipos:** la confección de los prototipos permite que el usuario pueda conocer cómo es la propuesta que el equipo de desarrollo implementa y puede aprobar la idea o corregir los elementos ajenos a los requisitos funcionales descritos.
- **Generación de casos de prueba:** la elaboración de casos de prueba permite comprobar el cumplimiento de los requisitos funcionales y que estos presentan la calidad requerida.

3.5.3. Resultado de aplicar los criterios de validación

Al concluir el proceso de revisión de requisitos se detectaron algunas inconsistencias que fueron erradicadas de inmediato. Entre las más comunes se pueden citar:

- Se interpretaron de forma incorrecta algunas de las funcionalidades y características solicitadas por el cliente.
- Errores ortográficos o tipográficos.

3.6. Validación de la aplicación

La prueba de software es una actividad donde un sistema o uno de sus componentes se ejecuta en circunstancias previamente descritas, registrándose los resultados obtenidos. El principal objetivo del diseño de casos de prueba es obtener un conjunto de pruebas que tengan la mayor probabilidad de descubrir los defectos del software. Las pruebas se ejecutan en la aplicación y mediante técnicas experimentales se trata de descubrir que errores presenta, para determinar el nivel de calidad y comprobar el grado de cumplimiento respecto de las especificaciones iniciales del sistema (PRESSMAN, 2001).

En la investigación se utilizan algunas de las pruebas definidas por Pressman[23] en su libro "Ingeniería de software: Un enfoque práctico", donde se pretende que el producto se pruebe desde la parte más pequeña del software hasta la más grande en forma de espiral usando los niveles, tipos, técnicas y herramientas de prueba.

Para comprobar que la solución implementada es válida y que cumple con los requisitos acordados con el cliente, se propone la siguiente estrategia de pruebas:

Tabla 3. Pruebas de software.

Nivel de prueba	Tipo de prueba	Método	Técnica
Integración	Estructural	Caja negra	Incremental
Sistema	Rendimiento y resistencia	Caja negra	Automática
	Funcional	Caja negra	Particiones equivalentes
Aceptación	Funcional	Caja negra	Alfa y Beta

3.6.1. Métodos de pruebas

Pruebas de caja blanca: se centran en la estructura de control del programa. Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejercita por lo menos una vez todos los caminos independientes de cada componente, ejerciten todas las decisiones lógicas en sus vertientes verdadera y falsa, ejecuten todos los bucles en sus límites y con sus

[23] Roger Pressman, es una autoridad internacionalmente reconocida en la mejora de procesos de software y en tecnologías de ingeniería de software. Por más de tres décadas, ha trabajado como ingeniero, gerente, profesor, autor y consultor de software en temas de ingeniería de software. Actualmente es presidente de R.S. Pressman *and Associates*, Inc., una firma consultora especialista en métodos y entrenamiento en ingeniería de software.

límites operacionales y ejerciten las estructuras internas de datos para asegurar su validez (PRESSMAN, 2001).

Pruebas de caja negra: son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno de un programa. La prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca; se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los métodos de caja blanca como son errores de 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. Las técnicas de prueba de caja negra se centran en el ámbito de información de un programa, de forma que se proporcione una cobertura completa de la prueba (PRESSMAN, 2001).

3.6.2. Pruebas de aceptación

Cuando se construye software para un cliente, se llevan a cabo las pruebas de aceptación para permitir que el cliente valide todos los requisitos. Son ejecutadas antes de que la aplicación sea instalada dentro de un ambiente de producción. La prueba de aceptación es conducida a determinar como el sistema satisface sus criterios de aceptación validando los requisitos que han sido levantados para el desarrollo, incluyendo la documentación y procesos de negocio (PRESSMAN, 2001).

Prueba alfa

La prueba alfa se lleva a cabo, por un cliente, en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario y registrando los errores y los problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado (PRESSMAN, 2001).

Prueba beta

La prueba beta se lleva a cabo por los usuarios finales del software en los lugares de trabajo de los clientes. El cliente registra todos los problemas que encuentra durante la prueba beta e informa a intervalos regulares al desarrollador. Como resultado de los problemas informados durante la prueba beta, el desarrollador del software lleva a cabo modificaciones y así prepara una versión del producto de software para toda la clase de clientes (PRESSMAN, 2001).

Resultados de las pruebas de aceptación

Para la ejecución de las pruebas de aceptación se realizaron varias reuniones con el cliente donde se le explicó el funcionamiento del sistema y se observó su interacción con los componentes del sistema

identificando errores de validación, de interfaz y de funciones incorrectas. Luego se realizaron pruebas por el cliente en su entorno de trabajo y sin observadores, permitiéndole iniciar el trabajo con el sistema en una situación real donde identificó un conjunto de no conformidades. Estas no conformidades fueron erradicándose a medida que fueron informados al equipo de analistas y desarrolladores. La realización de las pruebas de aceptación a la solución utilizando las técnicas alfa y beta, permitió demostrar que el sistema cumple satisfactoriamente los requisitos del cliente, emitiéndose el acta de aceptación, garantizando evaluar el grado de calidad del software. Además, el sistema fue avalado por el cliente, documento que se encuentra vinculado en el **Anexo 4.**

Durante la realización de las pruebas de aceptación se pudo comprobar que se logró agilizar la ejecución de los procesos de la vida interna y que es posible acceder a la información concentrada en el sistema desde cualquier computadora de la universidad siempre que los usuarios tengan los privilegios necesarios. Un ejemplo de la rapidez de los procesos utilizando el sistema es la gestión de los archivos. Antes para conocer si había disponibilidad en una de las secciones el Secretario debía revisar un listado con los números de expediente ubicados en los archivos donde se anotaban los que se liberaban y se tachaban una vez que eran asignados a un expediente. Actualmente el sistema permite seleccionar el archivo y la sección donde se desea ubicar el expediente y automáticamente muestra el número que se le asigna a este en caso de haber uno disponible en esa selección. Usando el sistema este proceso se realiza en cuestiones de segundos mientras que con el antiguo sistema se debía revisar los listados de cada archivo en busca de un espacio disponible lo que ocasionaba que pudieran cometerse errores.

3.6.3. Pruebas de integración

La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción. El objetivo es coger los componentes probados anteriormente y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño (PRESSMAN, 2001).

Integración incremental

A menudo al culminar el desarrollo de un sistema se tiende a una integración no incremental; es decir, a combinar todos los componentes y probar todo el programa en su conjunto. El resultado puede ser caótico con un gran conjunto de fallos y la consiguiente dificultad para identificar el componente (o componentes) que los provocó. En contra, se puede aplicar la integración incremental en la que el programa se construye y se prueba en pequeños segmentos en los que los errores son más fáciles de aislar y de

corregir, es más probable que se puedan probar completamente las interfaces y se puede aplicar un enfoque de prueba sistemática (PRESSMAN, 2001). Por esta razón se escogió el enfoque incremental para la realización de las pruebas de integración de la solución.

Resultados de las pruebas de integración

En la validación de la solución se probó la integración del sistema con los componentes del SGU, en este caso con los componentes Seguridad, Personal y Estructura y composición. La Tabla 4 muestra el caso de prueba de integración realizado al componente de Seguridad, el resto aparecen relacionadas en el **Anexo 5**. Al finalizar las pruebas de integración no se detectaron errores asociados a la interacción entre el SGO con los componentes del SGU.

También se realizaron pruebas de integración durante el desarrollo entre los componentes propios del sistema probando que estos se interrelacionan correctamente. El primer componente desarrollado fue Estructura y las funcionalidades Categoría de estructura y Responsabilidades del componente Configuración, necesarias para el funcionamiento del mismo. Luego se implementó la agrupación funcional Trámites de Configuración para su uso en el componente Movimientos. Este componente a su vez se relaciona con Estructura pues utiliza para su funcionamiento las estructuras existentes con las áreas y responsabilidades asociadas para la gestión de la información de los militantes y los trámites realizados a estos.

Tabla 4. Prueba de integración componente Seguridad.

Caso de Prueba: INT-S	
Componente al que se integra	Seguridad.
Condiciones de ejecución	El componente de Seguridad haya introducido los datos en la base de datos central y exista conexión con la misma.
Descripción de la prueba	Comprobar que el Sistema de Gestión de Organizaciones es capaz de realizar el acceso a funcionalidades y seguridad de negocio a partir de la información gestionada por el componente Seguridad.
Entradas/Pasos de ejecución	El componente de Seguridad introduce en la base de datos central los datos y el componente de Sistema de Gestión de Organizaciones consulta estos datos y define la seguridad del sistema.
Resultado esperado	Los usuarios tienen acceso a las funcionalidades de acuerdo al rol y sus responsabilidades.
Evaluación	Prueba satisfactoria.

3.6.4. Pruebas de sistema

Las pruebas de sistema están constituidas por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema basado en computadora. Aunque cada prueba tiene un propósito diferente, todas trabajan para verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las funciones apropiadas (PRESSMAN, 2001). Entre las pruebas de sistema que se realizan están las pruebas de rendimiento, resistencia y funcionalidad.

- **Pruebas de resistencia:** están diseñadas para enfrentar a los programas con situaciones anormales. La prueba de resistencia ejecuta un sistema de forma que demande recursos en cantidad, frecuencia o volúmenes anormales (PRESSMAN, 2001).
- **Pruebas de rendimiento:** están diseñadas para probar el rendimiento del software en tiempo de ejecución dentro del contexto de un sistema integrado. La prueba de rendimiento se da durante todos los pasos del proceso de la prueba (PRESSMAN, 2001).

Resultado de las pruebas de rendimiento y resistencia

Una prueba de rendimiento se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. El sistema se prueba con un número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la prueba. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación. Para llevar a cabo las pruebas de rendimiento y resistencia se utilizó la herramienta JMeter 2.3.1 descrita en el acápite 1.5.2 y se utilizó una copia local del sistema en una computadora con un microprocesador Intel Core i3, con 4GB de RAM a 3.30 GHz de velocidad . La prueba consistió en realizar a cuatro funcionalidades tres pruebas de 25, 50 y 75 hilos cada una, los cuales simulan 25, 50 y 75 accesos de usuarios respectivamente. Se definieron una lista de enlaces a los que se simuló el acceso aleatorio y a partir de ahí, se recolectaron los datos necesarios para su interpretación. A continuación se muestran en la Tabla 7 los resultados obtenidos con la herramienta y que fueron extraídos de las imágenes relacionadas en el **Anexo 5**.

Tabla 5. Resultados de las pruebas de rendimiento.

Aplicado a	Cantidad de Hilos	Muestras	Tiempo de ejecución (ms)				Rendimiento		
			Mín.	Max.	Media	Mediana	% Error	Pet/seg	Kb/seg
Listar archivos	25	1750	0	6513	301	8	0.0%	72.1/seg	1115.7
	50	3500	0	11898	589	13	0.0%	73.6/seg	1139.4
	75	5250	0	17909	899	21	0.0%	73.9/seg	1144.1
Configurar niveles escolares	25	2375	0	6010	314	8	0.0%	72.7/seg	987.4
	50	9500	0	11674	628	13	0.0%	58.6/seg	795.8
	75	7125	0	17193	913	14	0.0%	72.4/seg	983.5
Listar ingresos	25	2575	0	4971	228	6	0.0%	96.4/seg	1396.0
	50	5150	0	10567	514	13	0.0%	88.9/seg	1286.9
	75	7725	0	14698	834	21	0.0%	85.4/seg	1237.0
Modificar tipo de alta	25	2375	0	6117	298	8	0.0%	76.2/seg	1035.9
	50	4750	0	11310	615	14	0.0%	74.6/seg	1014.0
	75	7125	0	14450	932	20	0.0%	75.1/seg	1021.1

- **Cantidad de hilos:** cantidad de páginas o hilos que simulan la cantidad de usuarios interactuando con el sistema desde la misma URL.
- **Muestras:** cantidad de peticiones atendidas por el servidor.
- **Mínimo (Mín.):** indica el mínimo de tiempo de ejecución invertido para una petición con *n* usuarios haciendo peticiones de manera concurrente.
- **Máximo (Máx.):** indica el máximo de tiempo de ejecución invertido para una petición con *n* usuarios haciendo peticiones de manera concurrente.
- **Media:** representa el tiempo de ejecución promedio de una petición con *n* usuarios.
- **Mediana:** significa que el 50% de las peticiones realizadas por *n* usuarios tardaron menos del valor reflejado.
- **%Error:** indica la relación entre el total de peticiones y el número de peticiones que originaron errores.
- **Rendimiento (Pet. /seg):** hace referencia al número de peticiones que el servidor puede procesar en un segundo.

- **Rendimiento (Kb. /seg):** hace referencia a la cantidad de datos que el servidor puede procesar en un segundo.

Análisis de los resultados de las pruebas de rendimiento y resistencia

De manera general se puede observar que las peticiones son ejecutadas en tiempos relativamente rápidos. Para todas las muestras de usuarios la ocurrencia de errores se mantiene en 0.0%, lo que quiere decir que todas las peticiones hechas se ejecutan satisfactoriamente. Si se tiene en cuenta que el sistema está pensado para determinados usuarios, se puede asumir entonces que este resultado es satisfactorio, ya que el sistema es capaz de soportar la carga de entre 25 y 75 usuarios concurrentes sin que ocurran errores en el sistema.

Para comprobar la resistencia del sistema se seleccionó una de las funcionalidades probadas y se le realizaron pruebas para 100, 300 y 600 hilos o usuarios concurrentes, de las cuales se pudo concluir que al existir una cantidad elevada de usuarios haciendo peticiones concurrentes comienzan a aparecer algunos errores a la hora de cargar los elementos necesarios como las imágenes y los archivos css o js y el sistema pierde la capacidad de procesar todas las peticiones. Al analizar el porcentaje de estos errores se puede observar que en los casos más críticos, únicamente ocurre algún tipo de error en menos del 7% del total de las peticiones realizadas, lo que indica que en consecuencia con la cantidad de usuarios conectados no es un porciento elevado. Si se tiene en cuenta que en la universidad existen cerca de 600 militantes del PCC y no todos accederán al sistema, se puede asumir que el resultado obtenido sigue siendo satisfactorio, lo que significa que el sistema es capaz de soportar mucho más carga de la que se deberá enfrentar en el peor de los casos.

3.6.5. Pruebas funcionales

Tienen por objetivo probar que los sistemas desarrollados cumplen con los requisitos funcionales del software. Los probadores o analistas de pruebas no enfocan su atención en cómo se generan las respuestas del sistema, sino en el funcionamiento de la interfaz del sistema (PRESSMAN, 2001).

Partición equivalente

La partición equivalente es una técnica de prueba de caja negra que divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar. Un caso de prueba ideal descubre de forma inmediata una clase de errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico (PRESSMAN, 2001).

Capítulo 3: Implementación y evaluación de la solución

Resultados de las pruebas funcionales

Para la realización de las pruebas funcionales se generaron los artefactos “Diseño de casos de pruebas basado en requisitos”. Por cada requisito funcional del sistema se generó un documento en donde se recogen todos los datos necesarios para probar la interfaz. A continuación se muestra el diseño de caso de prueba que responde al requisito funcional “Listar estructuras”. Podrán encontrarse en su totalidad los restantes diseños de casos de pruebas en el expediente del proyecto.

Tabla 6.Diseño de caso de prueba “Listar estructuras”.

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Mostrar datos correctamente.	Mediante este escenario se muestra un árbol jerárquico de estructuras de una organización con las opciones: Asociar áreas administrativas, Asociar responsabilidades, Desactivar, Ver detalles y Modificar en el área de íconos internos, así como las opciones: Crear, Actualizar y Ayuda en el área de íconos flotantes.	El sistema muestra un listado de estructuras.	1. Al autenticarse el usuario, el sistema muestra por defecto todos los sistemas que están contenidos. 2. Una vez autenticado el usuario en el sistema, selecciona el componente "Estructura". 3. Selecciona en la agrupación funcional "Estructura" de la derecha, la funcionalidad “Estructura”. 4. El sistema muestra el árbol jerárquico de estructuras existentes hasta la fecha.
EC 1.2 No existen elementos creados.	Mediante este escenario en caso de que no exista creado ningún elemento se muestra el listado vacío.	El sistema muestra el listado vacío.	1. Al autenticarse el usuario, el sistema muestra por defecto todos los sistemas que están contenidos. 2. Una vez autenticado el usuario en el sistema, selecciona el componente "Estructura". 3. Selecciona en la agrupación funcional "Estructura" de la derecha, la funcionalidad “Estructura”. 4. El sistema muestra el listado de elementos vacío.
EC 1.3 Ordenar	Mediante este escenario se puede ordenar	El sistema permite ordenar ascendente (a-z) o	1. Al autenticarse el usuario, el sistema muestra por defecto todos

alfabéticament e	alfabéticamente.	descendentemente (z-a).	los sistemas que están contenidos. 2. Una vez autenticado el usuario en el sistema, selecciona el componente "Estructura". 3. Selecciona en la agrupación funcional "Estructura" de la derecha, la funcionalidad “Estructura”. 4. El sistema muestra el árbol jerárquico de estructuras existentes hasta la fecha. 5. Se selecciona en las flechas que están a la derecha de Plantillas, para ordenar alfabéticamente.

Resultado de las pruebas funcionales

A continuación se muestra la relación de no conformidades por iteración encontradas durante la realización de las pruebas funcionales a los 134 requisitos implementados en la primera versión del sistema.

Tabla 7. Relación de no conformidades por iteraciones.

Iteraciones	Cantidad de no conformidades	Asociadas a
1ra	66	Errores de interfaz, funciones incorrectas o ausentes.
2da	48	Errores de interfaz, de validación y ortografía.
3ra	0	-

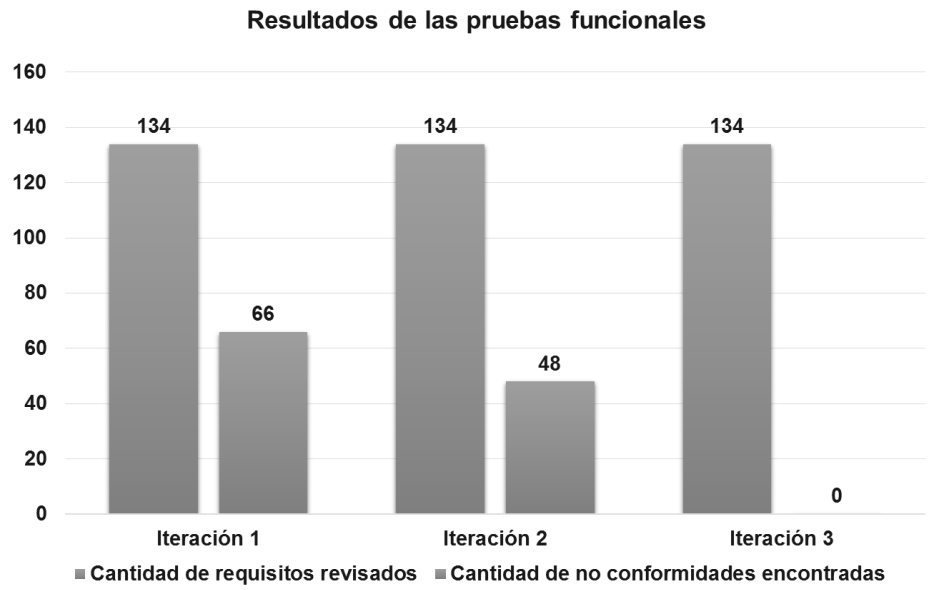


Figura 19. Relación de no conformidades por iteración.

3.6. Conclusiones parciales

En este capítulo se detallaron los estándares de diseño y codificación empleados en la implementación del Sistema de Gestión de Organizaciones permitiendo una mejor organización y comprensión del código. Con los resultados de la validación de los requisitos identificados se logró obtener un listado de requisitos correctamente redactados y descritos. La descripción de las pruebas utilizadas para asegurar la calidad del software, permitió obtener resultados satisfactorios, asegurando que el sistema implementado no contiene errores y tiene la aceptación requerida.

Conclusiones generales

Al término de la presente investigación para la implementación del Sistema de Gestión de Organizaciones se dio respuesta al problema de investigación planteado, resultando las siguientes conclusiones:

- El estudio de los sistemas similares permitió identificar características que sirvieron como base para elaborar la propuesta de solución.
- El estudio del reglamento y los estatutos del PCC propició un mejor dominio del funcionamiento de la organización, lográndose caracterizar los procesos de la vida interna de la militancia.
- La obtención, descripción y validación de los requisitos definieron las características técnicas y funcionales de la propuesta de solución.
- La aplicación de pruebas de software permitió detectar y corregir no conformidades identificadas durante la etapa de desarrollo, garantizando el cumplimiento de los requisitos de software.
- La solución agiliza y centraliza el trabajo con la información asociada a los procesos de la vida interna del PCC en la UCI así como su consulta rápida y segura a partir de la integridad y confiabilidad de los datos, además comparte y aprovecha la información como recurso colectivo respetando jerarquías y niveles de acceso de acuerdo a la función de cada persona dentro del proceso.

Recomendaciones

Para garantizar el perfeccionamiento progresivo de la solución propuesta se proponen las siguientes recomendaciones que tributarán a un producto de mejor calidad. Las funcionalidades que se recomiendan son:

- Implementar las funcionalidades correspondientes a: plan de trabajo, registro de actas y cotización.
- Desplegar la herramienta desarrollada en otras instituciones.
- Configurar las notificaciones y alertas para mantener informados a los usuarios de las acciones que ocurren en el sistema.
- Confeccionar los manuales de usuario para lograr un mejor entendimiento por parte del usuario sobre las funcionalidades de la aplicación.

Bibliografía referenciada

- **MIRANDA ESCALONA, Osmani A. 2004.** Revista Granma Ciencia. [En línea] Enero-Abril de 2004. http://www.grciencia.granma.inf.cu/2004_08_n1_a2.html.
- **GRANMA. 2014.** Periódico Granma. *La informatización de la sociedad, una prioridad para Cuba.* [En línea] 12 de Diciembre de 2014. [Citado el: 2015 de Febrero de 18.] http://www.granma.cu/cuba/2014-12-12/la-informatizacion-de-la-sociedad-una-prioridad-para-cuba.
- **RAE.** Diccionario de la Real Academia Española. [En línea] http://lema.rae.es/drae/?val=organizaci%C3%B3n.
- **PCC. 1998.** Estatutos del Partido Comunista de Cuba. [En línea] 1998. [Citado el: 2015 de Febrero de 10.] http://www.pcc.cu/pdf/documentos/estatutos/estatutos6c.pdf.
- **PCC. 2013.** Reglamento de las organizaciones de base del Partido Comunista de Cuba. [En línea] abril de 2013. [Citado el: 2015 de Febrero de 10.] http://www.pcc.cu/pdf/documentos/reglamento/base.pdf.
- **GARCÍA CHACÓN, Yunaisy.** *INFOEST. [en persona].* La Habana, Marzo de 2015.
- **GONZÁLEZ CARDOSO, Victor Gabriel y FIGUEREDO BUSTAMANTE, Eiler.** *Sistema para la gestión de los procesos de la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas.* Ciudad de la Habana: s.n., 2014.
- **CRESPO ESTÉVEZ, Mayté. 2012.** *Sistema gestor de información para las organizaciones: Módulo UJC.* La Habana: Universidad de las Ciencias Informáticas, 2012.
- **CONCEPCIÓN SINGLER, Mayté y ESPIÑEIRA ROBAINA, Yaimara. 2007.** *Análisis de un sistema para el control*. La Habana: Universidad de las Ciencias Informáticas, 2007
- **W3C.** *Introduction to HTML 4.* [En línea] [Citado el: 25 de mayo de 2015.] http://www.w3.org/TR/REC-html40/intro/intro.html
- **W3C. 2008.** *Extensible Markup Language (XML) 1.0 (Fifth Edition).* [En línea] [Citado el: 25 de mayo de 2015.] http://www.w3.org/TR/REC-xml/.
- **PHP.** [En línea] [Citado el: 18 de Febrero de 2015.] http://www.php.net/.
- **EGUILUZ, Javier.** [En línea] [Citado el: 18 de Febrero de 2015.] http://librosweb.es/libro/javascript/capitulo_1.html.
- **W3C. 2011.** *Cascading Style Sheets (CSS) Snapshot 2010.* [En línea] [Citado el: 18 de Febrero de 2015.] http://www.w3.org/TR/CSS/.

Bibliografía referenciada

- **OMG. 2006**. *UML 2.0*. [En línea] [Citado el: 18 de Febrero de 2015.] http://doc.omg.org/formal/2005-07-05.pdf.
- **BPMN.** *Object Management Group Business Process Model and Notation.* [En línea] [Citado el: 18 de Febrero de 2015.] http://www.bpmn.org/.
- **NETBEANS.** *NetBeans IDE - The Smarter and Faster Way to Code*. [En línea] [Citado el: 18 de Febrero de 2015.] https://netbeans.org/features/index.html.
- **PENCIL PROJECT**. [En línea] [Citado el: 18 de Febrero de 2015.] http://pencil.evolus.vn/Features.html.
- **VISUAL PARADIGM**. *Visual Paradigm suite 5.0 Lanzamiento.* [En línea] [Citado el: 18 de Febrero de 2015.] http://www.visual-paradigm.com/aboutus/newsreleases/vpsuite50.jsp.
- **POSTGRESQL.** *PostgreSQL*. [En línea] [Citado el: 18 de Febrero de 2015.] http://www.postgresql.org.es/sobre_postgresql#intro.
- **PGADMIN**. *pgAdmin: PostgreSQL administration and management tools*. [En línea] [Citado el: 18 de Febrero de 2015.] http://www.pgadmin.org/.
- **OPENSUSE**. *Apache.* [En línea] [Citado el: 19 de Febrero de 2015.] https://es.opensuse.org/Apache.
- **ÁLVAREZ, Miguel Ángel.** *CodeIgniter*. [En línea] [Citado el: 19 de Febrero de 2015.] http://www.desarrolloweb.com/articulos/codeigniter.html.
- **TELLEZ COLLAZO, Yasmany. 2013.** *10216_8 Arquitectura de Software Vista de tecnología e infraestructura.* s.l.: UCI, 2013.
- **JMETER.** *Apache JMeter***.** [En línea] [Citado el: 12 de Marzo de 2015.]. http://jmeter.apache.org/.
- **JQUERY.** *jQuery. [*En línea] [Citado el: 19 de Febrero de 2015.] http://jquery.com/.
- **SÁNCHEZ MÉNDEZ, Alelí Sánchez. 2013.** *Proceso de desarrollo de software DAC.* s.l.: Universidad de las Ciencias Informáticas, 2013.
- **SPARKS, Geoffrey.** Craftware. *Introducción al modelado de sistemas de software.* [En línea] 10 de Marzo de 2015. http://www.craftware.net/es/descargas/modelo_de_proceso_de_negocio.pdf.
- **PRESSMAN, Roger S**. *Ingeniería de software: Un enfoque práctico.* Madrid: s.n., 2001. Quinta edición.
- **SOMMERVILLE, I.** *Cuerpo del conocimiento de la Ingeniería de Software.* 17 de agosto de 2006.

Bibliografía referenciada

- **KIOSKEA.** *Entorno cliente/servidor.* [En línea] 12 de Marzo de 2015. http://es.kioskea.net/contents/148-entorno-cliente-servidor.
- **CODEIGNITER.** [En línea] [Citado el: 12 de Marzo de 2015.] http://escodeigniter.com/guia_usuario/overview/mvc.html.
- **MARTIN SOLER, Nayilet y SANTIESTEBAN ROJAS, Yander.2013**. *Diseñador de plantillas para el módulo Reportes del Sistema de Gestión Académica de Pregrado.* La Habana: Universidad de las Ciencias Informáticas, 2013.
- **SAAVEDRA, Jorge A. 2007.** [En línea] [Citado el: 12 de Marzo de 2015.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/.
- **MARQUEZ, Javier.** *Patrones de diseño GOF*. [En línea] [Citado el: 12 de Marzo de 2015.] https://infow.wordpress.com/category/patrones-de-disenogof/.
- **LARMAN, Craig. 1999.** *UML Y PATRONES. Introducción al análisis y diseño orientado a objetos.* s.l.: PRENTICE HAL, 1999.

Bibliografía consultada

- **PCC. 1998.** Estatutos del Partido Comunista de Cuba. [En línea] 1998. [Citado el: 2015 de Febrero de 10.] http://www.pcc.cu/pdf/documentos/estatutos/estatutos6c.pdf.
- **PCC. 2013.** Reglamento de las organizaciones de base del Partido Comunista de Cuba. [En línea] abril de 2013. [Citado el: 2015 de Febrero de 10.] http://www.pcc.cu/pdf/documentos/reglamento/base.pdf.
- **PRESSMAN, Roger S**. *Ingeniería de software: Un enfoque práctico.* Madrid: s.n., 2001. Quinta edición.
- **SÁNCHEZ MÉNDEZ, Alelí Sánchez. 2013.** *Proceso de desarrollo de software DAC.* s.l.: Universidad de las Ciencias Informáticas, 2013.
- **SOMMERVILLE, I.** *Cuerpo del conocimiento de la Ingeniería de Software.* 17 de agosto de 2006.
- **GONZÁLEZ CARDOSO, Victor Gabriel y FIGUEREDO BUSTAMANTE, Eiler.** *Sistema para la gestión de los procesos de la Federación Estudiantil Universitaria en la Universidad de las Ciencias Informáticas.* Ciudad de la Habana: s.n., 2014.
- **CRESPO ESTÉVEZ, Mayté. 2012.** *Sistema gestor de información para las organizaciones: Módulo UJC.* La Habana: Universidad de las Ciencias Informáticas, 2012.
- **CONCEPCIÓN SINGLER, Mayté y ESPIÑEIRA ROBAINA, Yaimara. 2007.** *Análisis de un sistema para el control*. La Habana: Universidad de las Ciencias Informáticas, 2007
- **TELLEZ COLLAZO, Yasmany. 2013.** *10216_8 Arquitectura de Software Vista de tecnología e infraestructura.* s.l.: UCI, 2013.
- **MARTIN SOLER, Nayilet y SANTIESTEBAN ROJAS, Yander.2013**. *Diseñador de plantillas para el módulo Reportes del Sistema de Gestión Académica de Pregrado.* La Habana: Universidad de las Ciencias Informáticas, 2013.
- **LARMAN, Craig. 1999.** *UML Y PATRONES. Introducción al análisis y diseño orientado a objetos.* s.l.: PRENTICE HAL, 1999.