Universidad de las Ciencias Informáticas
Facultad 3

Título: “Sistema para la Gestión de la Guardia Obrera-Estudiantil en la Facultad 3.”

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

Autores:
Dailín Mata Sánchez
Aroldo Navarro Quintero

Tutores:
Ing. Wendy Gracia Valdés
Ing. Didier Roque Ginebra

Ciudad de La Habana, Junio 2013
“Año 55 de la Revolución”

DECLARACIÓN DE AUDITORÍA

Declaramos ser autores de la presente tesis 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 a los ____ días del mes de _____ del año _____.

_____

Dailín Mata Sánchez

Autor

_____

Aroldo Navarro Quintero

Autor

_____

Ing. Didier Roque Ginebra

Tutor

_____

Ing. Wendy Gracia Valdés

Tutor

DATOS DEL CONTACTO

- ***Síntesis de los tutores:***

Ing. Wendy Gracia Valdés
Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en el año 2010 con Título de Oro. Se ha desempeñado como líder de gestión del dpto. Desarrollo de Productos del Centro CEIGE.
Correo electrónico: wgracia@uci.cu

Ing. Didier Roque Ginebra
Graduado de Ingeniero en Ciencias Informáticas en la UCI en el 2008. Actualmente se desempeña como desarrollador de la línea Contabilidad del proyecto ERP Cuba.
Correo electrónico: dginebra@uci.cu

- ***Síntesis de los autores:***

Dailín Mata Sánchez
Estudiante de la Universidad de las Ciencias Informáticas.
Correo electrónico: dmata@estudiantes.uci.cu.

Aroldo Navarro Quintero
Estudiante de la Universidad de las Ciencias Informáticas.
Correo electrónico: anquintero@estudiantes.uci.cu.

DEDICATORIA

Al mejor padre del mundo, por guiar cada uno de mis pasos, por brindarme siempre su amor a cambio de nada, sus enseñanzas, sus años de sacrificio, su confianza, sus consejos y apoyo en cada momento, por ser un padre especial, por confiar tanto en mí y estar seguro que no lo defraudaría. Te quiero mucho.

A mi eterno amor, la mujer más linda del mundo, mi mamá, por todo lo que has hecho en la vida por mí, solo de darme la posibilidad de venir al mundo fue lo más hermoso que cualquier ser humano puede pedir, por su amor, confianza, dedicación, apoyo, por ser guía ejemplar y constante, por ser una madre excepcional y no te comparo con nada en este mundo, sin ser perfecta me has guiado por el buen camino. Te quiero mucho.

A toda mi familia que siempre ha estado tan orgullosa de mí.

Aroldo

Les dedico este trabajo de diploma primeramente a mis padres que son las personas más importantes en mi vida, a mis sobrinitas que las quiero mucho, a mi tía Maritza y a mi novio José que tanto desearon este logro, los amo.

Dailín

AGRADECIMIENTOS

A mis padres por confiar en mí y brindarme su apoyo incondicionalmente durante estos 5 años de carrera, por darme ánimos aun cuando están lejos de mí físicamente, pero espiritualmente siempre los llevo conmigo, por sacrificarse por mí, les agradezco todo lo que soy hasta este momento y lo que puedo llegar a ser.

A mi mamá por brindarme siempre su amor y cariño y estar ahí siempre que la necesite, por darme la fuerza de seguir adelante. A mi papá, por ser un ejemplo en mi carrera, espero que estés orgulloso de mí. Los quiero infinitamente.

A mi hermano por las tantas broncas que hemos tenido, a pesar de eso te considero el mejor hermano del mundo y quiero que sepas que te quiero mucho.

A mis sobrinos a los cuales quiero con el alma, espero que en un futuro les sirva este trabajo como ejemplo de sacrificio profesional y que sepan que ellos lo pueden hacer mejor que yo.

A toda mi familia que de una forma u otra han contribuido a mi educación como persona y han incentivado en mí el deseo de ser alguien útil en la vida, en especial a mi Abuela por siempre preocuparse por mí, quiero que sepas que te quiero mucho.

Quiero agradecer a quienes también he considerado como mi familia, y de los cuales casi ya me tengo que despedir, pero esperemos que no sea por siempre. Ellos son mis mejores amigos, mis hermanos diría yo. Marcos, a quien desde los viejos tiempos de la primaria, he considerado un hermano que siempre me puede aconsejar. A Celso, Daimara, Wichy, Otero, Abdel ¨El Señor Cara de Papa¨, Radames, David ¨El Guata¨ con los cuales he compartido momentos que siempre perdurarán en la memoria.

A tres personas que las considero como una madre más, ella son: Sonia, Teresa y Sarahi, gracias por sus consejos nunca las olvidaré.

A mis amigos y amigas que me han acompañado durante estos años, gracias por ayudarme, soportarme, escucharme y estar allí para mí. A mis compañeros de aula por todos los momentos que pasamos juntos, en especial Javier Acebal y Juan Carlos con los cuales cogí mis primeras fiestas, quiero que sepan que pueden contar conmigo para lo que sea, a Yasmany, Cire, Anchel, Josbel, Hernán por las locuras formadas en el

apartamento. A Leo por aguantarme tantos carnavales en su casa y ser más que un amigo un hermano, a Lamothe, Perdomo, Daymi, Yiya, Alexei, Yasmin y su hermana por compartir buenos momentos juntos. A los Sensuales de la Pista y a los Guarapeños con los cuales pase momentos que nunca olvidaré. A Evelio el viejo más sabroso del edificio 7, a Correa y Cesar, mis discípulos del gimnasio, espero algún día los alumnos superen al maestro.

Al piquete la Onda Fashion: Luis Carlos, Ramón, Luis Angel, Cangri "El Cachalotero", Jorgito, Pompa.

A Duvergel y el Titi, que a pesar de conocerlos en 3er año han sido como unos hermanos para mí, nunca me olvidaré de ustedes.

A Lisbeth, Micha, el Chino por los buenos momentos que pasamos juntos.

A Jenny por aguantar mis lindos piropos todos los días, además de ser mi costurera personal y de sacarme de varios apuros.

A mis amigas las cocodrilas de la 7, las quiero mucho, espero nunca me olviden. A las Titis por pasar gratos momentos juntos, las quiero.

A mis hijos de la Universidad que sin un padre como yo, no fueran hoy lo que son: para Bárbaro y Vladimir sigan el ejemplo de su padre a ver si se gradúan.

A mi compañera de tesis por aguantar mis ataques de preocupación, de tener una paciencia enorme conmigo y ser más que compañera, una amiga.

A mis tutores, en especial a Wendy, a usted le agradezco con todo mi corazón y con profunda gratitud la inmensa ayuda brindada y consejos adquiridos en verdad muchísimas gracias.

A todas las personas que han confiado en mí y que de una forma u otra me han apoyado a lo largo de la carrera. Gracias.

Aroldo

A mi mamá y a mi papá por el apoyo incondicional que me dieron, por la confianza que siempre depositaron en mí, por darme los mejores consejos del mundo, gracias a los dos por ser mi razón de ser.

A mi tía Maritza por quererme siempre como una madre, por brindarme todo su apoyo, ayudarme durante los 5 años de mi carrera de forma incondicional y darme muy buenos consejos.

A mi novio José por quererme tanto, ayudarme cuando más lo necesitaba y cuando pensaba que no llegaría a la meta, te amo mi amor.

A toda mi familia que siempre ha estado al tanto de mi trayectoria y resultados.

A mi compañero de tesis por tantas horas que dedicó a la confección de este trabajo tan importante para los dos, a él mi más sincero agradecimiento.

A mis compañeros de grupo, que siempre estuvieron ayudándome durante toda mi trayectoria en la universidad, supieron ganarse mi cariño, por los buenos momentos que pasamos juntos que son inolvidables, por comportarse como una gran familia para mí, no quisiera mencionar nombres porque a todos los quiero mucho, y ha sido un grupo muy unido y compartidor, los que estamos desde primer año juntos y los que se incorporaron después que fueron el grupito de los cadetes a los cuales le cogimos mucho aprecio, y de ahí destacar a mi amigo Maikelito, a los que a medida que pasaban los años se fueron, de los cuales mencionaré dos que fueron mis mejores amigos Yandy y Alain, nunca los olvidaré.

Al testarudo de Maurice que aunque siempre me peleaba me dio muy buenos consejos y me ayudó mucho en los 5 años de mi carrera.

A mis amigas que siempre estuvieron conmigo en las buenas y en las malas, Lisbet, Male, Jenny, Janny, Maire, Yamilita, Isa, Elizabet, Rocio, Yissell que son mis compañeras del grupo también y las demás que se incorporaron después al piquete de las locotas como Maite, Sahily, Miladis, Betty y Maria Elena, ah y mencionar a Yessika que no es de mi grupo pero es mi amiga también y cuando la molesté siempre me ayudó con su mejor disposición.

A mis hermanos que además de amarlos son un ejemplo para mí.

A mis tutores y en especial a mi tutora Wendy que tanto nos ayudó y se dedicó por completo a nosotros sin esperar nada a cambio, para ella lo mejor de este mundo porque se lo merece.

A la familia de mi novio, mi super abue Tati que con tanto cariño y dedicación siempre me ha tratado, a mi tío Eri y a mi suegra Maige que de una forma u otra han ocupado un lugar muy especial en mi vida.

A las personas de la línea que siempre estuvieron al tanto del avance de nuestro trabajo y disponibles para cualquier duda.

En fin a todos los que nos brindaron su apoyo incondicional.

Dailín

RESUMEN

El impulso de los sistemas de gestión en estos tiempos promete estrategias encaminadas a lograr la optimización de los procesos de negocio; permitiendo un ascenso perdurable en el desarrollo de la gestión de políticas, procedimientos y procesos, los cuales son la principal meta de estos sistemas.

El objetivo del presente trabajo se basa en el desarrollo de un sistema de gestión para la planificación de la Guardia Obrera-Estudiantil (GOE) en la Facultad 3 de la Universidad de las Ciencias Informáticas, que permita la gestión de procesos que tributan a un adecuado desempeño, organización y funcionamiento que posibilitará realizar de forma más eficiente el manejo y gestión de la información de dicha actividad. Para dar cumplimiento al objetivo se realizó un estudio de sistemas informáticos que incluyen este proceso, concluyendo que era necesario construir una solución propia. Las herramientas y tecnologías utilizadas para su desarrollo son las definidas por el Centro de Informatización de la Gestión Entidades (GEIGE).

Como resultado final se propone un sistema que permite la planificación ágil y más organizada de la Guardia Obrera-Estudiantil, logrando una mayor rapidez en su ejecución.

Palabras Claves: Centro de Informatización de la Gestión Entidades (CEIGE), Guardia Obrera-Estudiantil (GOE), Sistemas de gestión.

ÍNDICE DE CONTENIDOS.

INTRODUCCIÓN.

Con el avance de las nuevas Tecnologías de la Información y las Comunicaciones (TIC) el ser humano ha tenido la posibilidad de mejorar, optimizar e informatizar los servicios que se brindan hoy en día, con el objetivo de ofrecerlos de la forma más sencilla, rápida y menos costosa posible, mediante software que posea una alta calidad. La informatización de procesos contribuye a la reducción de costos, permite racionalizar el trabajo, reduce el tiempo y los recursos dedicados al mantenimiento y asegura una mejor calidad del trabajo y el desarrollo del mismo.(1) Las personas se han convertido en consumidores directos e indirectos de las TIC, por esta razón el desarrollo y aplicación de éstas han creado un gran impacto y proyección en los individuos, sociedad y organizaciones.(2)

Los sistemas de gestión de la información han tenido un auge en los últimos años que ha provocado que la mayoría de las empresas tengan la necesidad de enfrentar este desafío cambiante y en aras del desarrollo, a tal punto que se les ha hecho imprescindible contar con estos sistemas para un mejor desarrollo.(3)

En la Universidad de las Ciencias Informáticas (UCI) existen muchos medios y recursos materiales que permiten a los estudiantes y trabajadores satisfacer sus necesidades, por lo que deben ser del cuidado y protección de los mismos. Desde sus inicios la UCI lleva a cabo una estrategia para velar por la protección y seguridad de estos medios, a través del Sistema de GOE, donde cada estudiante y trabajador de la universidad está involucrado. Este sistema es elaborado de acuerdo con las necesidades de cubrir las áreas de la residencia estudiantil, el área docente y el perímetro universitario por cada facultad, en dependencia de la cantidad de estudiantes y trabajadores con que cuentan dichas facultades y cubriendo las postas mediante turnos de 2 horas y media, con el objetivo de prevenir ilegalidades, acciones delictivas y pérdidas de objetos de valores cuantiosos puestos al servicio de la universidad.

A finales del curso pasado se realizaron análisis sistemáticos para realizar un nuevo proceso de restructuración, planificación y organización de la guardia, con el objetivo de realizar el procedimiento anteriormente mencionado más competente en la UCI. Con la nueva reestructuración, los cambios fueron: que las postas pasan a ser a nivel de universidad y el horario es de 4 horas.

A inicios del presente curso escolar 2012-2013 se pone en práctica dicha reestructuración, la cual era totalmente diferente a como se venían realizando hasta el momento, implicando cambios bruscos en la

organización y planificación de la misma. Actualmente los sistemas ya elaborados en la UCI, además de no cumplir con los nuevos parámetros para la guardia, no tienen en cuenta los siguientes requisitos para hacer una adecuada planificación de la GOE:

- si el trabajador está de vacaciones, o se encuentra en Venezuela o cualquier otro país cumpliendo misión; si está de certificado médico, o licencia, tanto de maternidad como estudiantil no se tiene en cuenta en la planificación de la guardia.
- si es trabajadora con niño pequeño solo se le planifica guardia de día.
- la fecha del cumpleaños del trabajador o el estudiante implicado no le puede coincidir con la planificación de su guardia.

El proceso de planificación y control de la GOE en la Facultad 3 se le hace difícil y engorroso a la Vicedecana de Administración y Servicios, que es la encargada de planificar y controlar este proceso ya que:

- Maneja gran número de información.
- Genera los reportes de forma manual, lo que implica la pérdida de información, la duplicación de la misma y que los reportes no sean los más exactos.
- En ocasiones no tiene en cuenta algunas de las restricciones anteriormente mencionadas en el momento de realizar la planificación debido a la cantidad de personal que maneja, además de emplear demasiado tiempo para realizar dichas planificaciones.

A pesar de que los estudiantes y trabajadores estén informados de la planificación de la guardia a través del correo, puede ocurrir el caso de que el correo llegue con demora y que el estudiante o trabajador involucrado en la guardia de ese día pierda la información, por lo que no se cuenta con un sistema capaz de realizar el aviso al implicado en dicha guardia, que le permita informarse sobre la cantidad y regularidad de sus guardias teniendo en cuenta todas las afectaciones y factores que influyen en dicha planificación.

Para solucionar estas dificultades, es necesario desarrollar una aplicación capaz de gestionar todas estas actividades y facilitar el monitoreo y control necesario sobre estos procesos, permitiendo así un trabajo más ágil sobre estas actividades.

Teniendo en cuenta lo antes planteado se identifica el siguiente **problema a resolver:** La actual gestión de información relacionada con la GOE en la Facultad 3 afecta su planificación y control. De esta forma

queda definido como **objeto de estudio:** Procesos de gestión de planificación y control, enmarcado en el **campo de acción:** Sistemas para la gestión de planificación y control. Para darle solución al problema se plantea el siguiente **objetivo general:** Desarrollar un sistema para la gestión de información relacionada con la GOE, que permita realizar una adecuada planificación y control de este proceso en la Facultad 3, el cual se desglosa en los siguientes **objetivos específicos:**

- Elaborar el marco teórico a partir del estudio y análisis crítico de las herramientas existentes.
- Realizar modelado del negocio para comprender como funciona el negocio que se desea automatizar.
- Realizar el análisis y diseño del sistema para la gestión de la guardia.
- Implementar el sistema para la gestión de la guardia.
- Validar el sistema implementado mediantes las pruebas de caja negra.

Para dar cumplimiento a los objetivos propuestos se diseñan las siguientes **tareas generales:**

- Caracterizar los procesos relacionados con la gestión de la guardia en la Facultad 3.
- Valorar las soluciones informáticas y experiencias asociadas a la gestión de la guardia.
- Identificar los requisitos funcionales a resolver.
- Realizar la especificación de requisitos funcionales.
- Realizar la administración de los requisitos funcionales.
- Realizar los prototipos de interfaz de usuario.
- Realizar las validaciones de los requisitos funcionales.
- Elaborar los diagramas de clases del análisis y diagramas de colaboración.
- Describir los elementos arquitectónicos.
- Caracterizar, identificar y aplicar los patrones.
- Elaborar los diagramas de clases del diseño y diagramas de secuencia correspondientes al diseño del sistema.
- Elaborar los diagramas: modelo de componentes, modelo de datos y diagrama de despliegue correspondientes al diseño del sistema.
- Validar el análisis y diseño.
- Realizar diseños de casos de prueba.
- Implementar las vistas y negocio.
- Programar consultas en la base de datos, así como procedimientos almacenados, configuraciones y mantenimientos necesarios.

- Validar la implementación del sistema. Aplicar los diseños de casos de prueba a la solución.

Se presenta como **Idea a defender:** Si se desarrolla un sistema para la gestión de la información relacionada con la GOE se realizará una adecuada planificación y control de este proceso en la Facultad 3.

Para llevar a cabo esta investigación se utilizaron los siguientes **métodos científicos**:

- **Métodos Teóricos:**
  - **Analítico-sintético:** posibilita extraer e identificar conceptos, características y otros elementos de la bibliografía consultada que posteriormente ayudan a establecer una propuesta adecuada a las necesidades del sistema. Luego del análisis de esta información es necesario organizarla y sintetizarla para la elaboración de una proposición que contenga una estructura apropiada con los elementos fundamentales del objeto estudiado.
  - **Histórico-lógico:** para realizar el estudio del estado del arte relacionado con el proceso de gestión de guardia.
  - **Modelación:** para elaborar los artefactos que se generan en las fases del desarrollo de la solución propuesta.
- **Métodos Empíricos:**
  - **Entrevista:** para obtener información valiosa sobre el proceso de gestión de guardia en nuestra entidad y poder identificar los requisitos funcionales y no funcionales que debe cumplir el sistema.

El trabajo de diploma se estructura en tres capítulos, los cuales se describirán a continuación:

- **CAPÍTULO 1: Fundamentación teórica.** Se realiza un estudio del estado del arte sobre los sistemas que gestionan la guardia. Además se analizarán los principales conceptos relacionados, así como las tendencias, técnicas, tecnologías, software usado en la actualidad, la fundamentación del uso de los lenguajes, herramientas, técnicas de captura y validación de requisitos y modelo de desarrollo del software.

- **CAPÍTULO 2: Análisis y diseño de la aplicación.** En este capítulo se obtendrán un conjunto de artefactos correspondientes a la modelación de los procesos del negocio como el mapa de proceso, el diagrama de proceso del negocio y el modelo conceptual. Se realiza el levantamiento de los requisitos funcionales y no funcionales, que sirven de entrada a la definición de los procesos que se

implementarán en el sistema, además de realizarse la descripción de las clases y los métodos más importantes. En consecuencia, se obtendrán un conjunto de artefactos que serán de gran valor para la fase de construcción como son: el diagrama de clases del diseño, el modelo físico de datos, diagramas de secuencia, diagrama de componentes y por último los prototipos de interfaz de usuario funcionales. Igualmente se especificará la utilización de un conjunto de patrones dentro del diseño de la aplicación.

- **CAPÍTULO 3: Implementación y prueba del sistema.** En este capítulo se realiza una breve descripción del marco de trabajo utilizado, así como los estándares a utilizar durante la implementación del sistema. También se describen las clases y funcionalidades de dicho sistema, además de realizar el diagrama de despliegue que va a visualizar los nodos físicos necesarios para el correcto funcionamiento de la aplicación. Por último se valida el diseño propuesto a través de las métricas Tamaño Operacional de Clases (TOC) y Relaciones entre Clases (RC), se describen las pruebas de caja negra realizadas al software en aras de detectar errores o defectos relacionados con su funcionalidad y se muestran los resultados obtenidos de la validación del sistema, de los requisitos y el diseño.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En el presente capítulo se dará una explicación acerca de los diferentes sistemas de guardias existentes tanto en Cuba como en el mundo, además de los ya implementados en la Universidad de las Ciencias Informáticas.

De estos sistemas se abordarán sus principales características para poder identificar los puntos reutilizables y basarse en ellos para ayudar en la solución del problema en cuestión. Por lo que uno de los objetivos que se persigue en el capítulo es la realización de un estudio referente al estado del arte, así como las tecnologías, sus tendencias, herramientas, metodologías aplicadas y lenguajes que serán utilizados para el desarrollo de la herramienta informática para facilitar una adecuada planificación y control de la guardia.

1.2. Conceptos generales

1.2.1 Gestión

Indica, que se trata de realización de diligencias enfocadas a la obtención de algún beneficio, tomando a las personas que trabajan en la compañía como recursos activos para el logro de los objetivos. Por otro lado también se debe tener en cuenta que la gestión en este caso requiere definir por su parte todas las políticas utilizadas por el personal para que así el mismo pueda ponerse en funcionamiento articulando las funciones sociales teniendo en cuenta las metas que posee la empresa.(4)

Importante subrayar que la gestión tiene como objetivo primordial conseguir aumentar los resultados óptimos de una industria o compañía y depende fundamentalmente de cuatro pilares básicos gracias a los cuales puede conseguir que se cumplan las metas marcadas. Dichos pilares anteriormente mencionados son:(4)

- ✓ **Estrategia:** es el conjunto de líneas y de trazados de los pasos que se deben llevar a cabo, teniendo en cuenta factores como el mercado o el consumidor, para consolidar las acciones y hacerlas efectivas.
- ✓ **Cultura:** es el grupo de acciones para promover los valores de la empresa en cuestión, para fortalecer la misma, para recompensar los logros alcanzados y para poder realizar las decisiones adecuadas.
- ✓ **Estructura:** son las actuaciones para promover la cooperación, para diseñar las formas para

compartir el conocimiento y para situar al frente de las iniciativas a las personas mejores cualificadas.

- ✓ **Ejecución:** consiste en tomar las decisiones adecuadas y oportunas, fomentar la mejora de la productividad y satisfacer las necesidades de los consumidores.

1.2.2 Sistema de gestión

"Un Sistema de Gestión es un conjunto de etapas unidas en un proceso continuo, que permite trabajar ordenadamente una idea hasta lograr mejoras y su continuidad. Se establecen cuatro etapas en este proceso, que hacen de este sistema, un proceso circular virtuoso, pues en la medida que el ciclo se repita recurrente y recursivamente, se logrará en cada ciclo, obtener una mejora."(5)

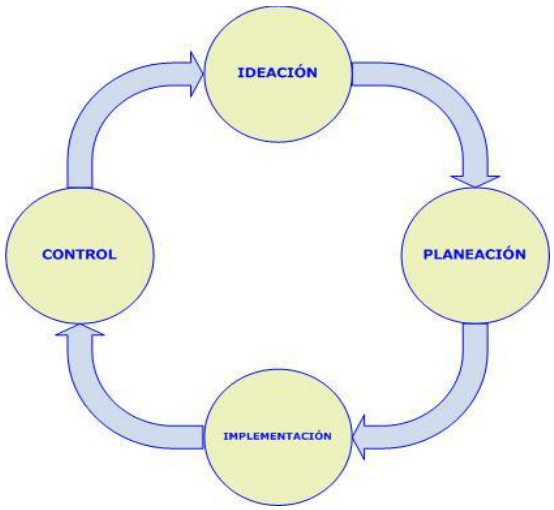

Figura 1. Etapas de Sistema de Gestión

Las cuatro etapas del **sistema de gestión** son:

1. Etapa de Ideación.
2. Etapa de Planeación.
3. Etapa de Implementación.
4. Etapa de Control.

- ✓ **Etapa de Ideación:** el objetivo de esta etapa es trabajar en la idea que guiará los primeros pasos del proceso de creación que se logra con el sistema de gestión propuesto.
- ✓ **Etapa de Planeación (Planificación):** en esta etapa se definen las estrategias que se utilizarán, la estructura organizacional que se requiere, el personal que se asigna, el tipo de tecnología que se necesita, el tipo de recurso que se utiliza y la clase de control que se aplica en todo el proceso.
- ✓ **Etapa de Implementación (Gestión):** se entiende por gestión, la acción y efecto de administrar. Pero, en un contexto empresarial, esto se refiere a la dirección que toman las decisiones y las acciones para alcanzar los objetivos trazados. Es importante destacar que las decisiones y acciones que se toman para llevar adelante un propósito, se sustentan en los mecanismos o instrumentos administrativos (estrategias, tácticas, procedimientos, presupuestos, etc.), que están sistémicamente relacionados y que se obtienen del proceso de planificación.
- ✓ **Etapa de Control:** el control es una función administrativa, esencialmente reguladora, que permite verificar (o también constatar, palpar, medir o evaluar), si el elemento seleccionado (es decir, la actividad, proceso, unidad, sistema, etc.), está cumpliendo sus objetivos o alcanzando los resultados que se esperan. Es importante destacar que la finalidad del control es la detección de errores, fallas o diferencias, en relación a un planteamiento inicial, para su corrección y/o prevención.(5)

1.3. Sistemas informáticos de gestión y planificación

Analizando la bibliografía consultada se pudo encontrar la propuesta de algunos productos Web que incorporan algunas de las funcionalidades que se propone para el producto final. Estas funcionalidades son:

- ➢ Notificar a los implicados de la planificación de la guardia.
- ➢ Configurar turnos y postas de la guardia.
- ➢ Generar la planificación de la guardia.
- ➢ Permitir escoger el período a planificar.
- ➢ Realizar permutas de guardia.
- ➢ Realizar control de la guardia.

1.3.1. Sistemas informáticos existentes en el mundo

Visual Time.

Es una completa solución 100% web, que permite obtener en tiempo real toda la información que necesita para la gestión de los recursos humanos con un gran ahorro de dedicación, reduce los costes y asegura el retorno de la inversión en un tiempo mínimo. Tiene como objetivo automatizar el control horario y flexibilizar los calendarios para poder optimizarlos en función de las necesidades reales de trabajo de la empresa.

Características:

- Gestiona todas las incidencias que se están produciendo sobre el plan horario; ausencias, retrasos, excesos y enlaza los datos con su programa de nómina.
- Estructura de forma gráfica la organización de la empresa, gestiona múltiples contratos por empleado, almacena la información que considera necesaria, define todo tipo de horarios, calendarios, etc.
- Permite crear calendarios anuales, mensuales, o del número de días que se desee.
- Incluye de forma estándar un completo módulo de planificación de calendarios, individuales o de grupo, optimizado para disponer de todos los datos visualmente y poder acceder a toda la información de forma rápida**.**(6)

Este sistema incluye dentro de sus funcionalidades la realización de la planificación de calendarios, permite realizar cambios de horarios y además la notificación de cualquier incumplimiento de convenio en el momento de la planificación, pero no realiza la configuración de los turnos y las postas, ni tampoco incluye la realización del control de la guardia.

Tecno hospital.

Esta herramienta gestiona las guardias, informa de las coberturas existentes para cada área de trabajo, controla las presencias en cada turno e informa puntualmente y al detalle a cada miembro del equipo del turno que le ha sido asignado en cualquier fecha del calendario. La aplicación es 100% Internet, por lo que permite a los responsables de personal manejar la información desde cualquier parte del mundo y a los trabajadores consultarla.

Es un sistema de gestión web automático que permite planificar los turnos de personal en cualquier empresa y que permite reducir un 70% el tiempo dedicado a esta tarea.

Características:

- Genera automáticamente la planificación en base a unos parámetros definidos inicialmente como: área, categoría, número de horas máximas según los convenios de enfermería, número de horas máximas de guardia, o número mínimo y máximo de enfermeros necesarios por turno.
- Ofrece un portal de usuarios con servicios para los enfermeros y enfermeras.
- Permite intercambiar de forma sencilla los turnos entre empleados, modificar horarios, establecer turnos personalizados o trabajar con turnos predefinidos.
- Permite consultar en todo momento si un turno está cubierto de acuerdo a los parámetros predefinidos y, en caso de faltar personal, propone una lista de trabajadores para cubrir dicho turno.(7)

Este sistema integra dentro de sus funcionalidades la generación automática del portal de consulta para los trabajadores de cada área, donde pueden consultar sus turnos, horarios, guardias o calendario laboral, permite realizar la configuración de los turnos y el intercambio de los mismos, informa en todo momento del número de horas mensuales y anuales que lleva acumuladas el trabajador de acuerdo a la programación que se le ha establecido, cuenta con un control de incidencias laborales diarias pero no permite escoger el periodo a planificar.

OPTIHPER.

Es un sistema para la asignación automática de horarios y tareas al personal de una empresa, teniendo en cuenta el conjunto de tareas a realizar, el personal disponible y su cualificación, restricciones y preferencias existentes.

Características:

- Permite adaptar, modificar y validar interactivamente una planificación previa ante incidencias o cambios posteriores (reactividad on-line).
- Incluye interfaces expresivas y amigables. Así mismo, puede adquirir los datos necesarios directamente desde Bases de Datos estándar.
- Genera automáticamente informes gráficos y textuales, con información general o particularizada.
- Facilita la reubicación y cambio de turnos, pudiendo ser auto-gestionado por los propios trabajadores, satisfaciendo las condiciones y restricciones existentes y bajo supervisión de la empresa. En caso de imposibilidad de atender los cambios propuestos, el sistema puede proveer de alternativas.
- Re-planifica la asignación de tareas u horarios en caso de incidencias.

- Está implementado en ANSI C, es multiplataforma (unix, linux, windows, etc.) y no hace uso de rutinas o código protegido por terceros.(8)

Este sistema incluye dentro de sus funcionalidades la planificación de horarios y/o turnos al personal, permitiendo escoger el periodo, ya sea semanal, mensual o anual. Realiza la reubicación y cambio de turnos, notifica con antelación los turnos de trabajo y las tareas concretas a realizar, pero no permite la configuración de los turnos y las postas y la realización del control de la guardia.

1.3.2 Sistemas informáticos existentes en Cuba

Sistema de Guardia Estudiantil de la Facultad 3.

Para el desarrollo de este sistema se utilizó como lenguaje de programación PHP y JQuery, como metodología XP, como IDE NetBeans y como Gestor de Base de Datos PostgreSQL.

Características.

- Permite generar la planificación de la guardia.
- Permite configurar las postas y los turnos.
- Realiza la planificación teniendo en cuenta criterios como: no repetir los turnos, planificar primero a los evaluados de M del mes anterior, planificar seguidamente a los que no hayan realizado la guardia y después al resto.
- Notifica la planificación por correo y por el Jabber.
- Permite evaluar, gestionar las guardias y los usuarios.
- Les facilita a los estudiantes visualizar cuándo les toca su guardia y sus evaluaciones.

Este sistema se centra solamente en el proceso de planificación de la guardia estudiantil. Dentro de sus funcionalidades incluye la generación de la guardia mediante la utilización de un método bayesiano, con un tiempo de respuesta del sistema de 5 minutos al realizar la planificación de un mes. No permite realizar permutas de guardia y tampoco tiene en cuenta la realización del control de la misma.

PortalFacultad7.

El sitio de la Facultad 7 de la Universidad de las Ciencias Informáticas tiene como objetivo proporcionarle a la dirección de la facultad la posibilidad de tener registrada toda la información necesaria referente a los procesos de la guardia estudiantil y cuartelería. Para el desarrollo de esta aplicación se utilizaron: la metodología RUP, como lenguaje de modelado se utilizó: UML y como herramienta de modelado visual:

Visual Paradim, como herramienta de diseño: Dreamweaver, como lenguaje de programación: PHP, como lenguaje del lado de servidor y como framework Symfony.

Características:

- Proporciona que se pueda llevar un control personalizado de la evaluación de los estudiantes en la residencia estudiantil.
- Genera un conjunto de reportes, en formato digital, acerca de estas dos actividades, que pueden ser impresos.
- Facilita la planificación y el control de la guardia estudiantil y la cuartelería a los responsables de estas tareas.
- Posibilita realizar de forma más eficiente el manejo y gestión de la información de estas actividades.(9)

Este sistema se centra en el proceso de guardia estudiantil, en el cual incluye varias de las funcionalidades necesarias como la generación de la planificación de la guardia estudiantil, permitiendo que las postas y los turnos sean configurables; pero no incluye las restantes funcionalidades importantes para una adecuada planificación de la misma, además, no integra las funcionalidades referentes al proceso de control de la guardia.

Gestor Web para el control de la Guardia Obrera de la Universidad de las Ciencias Informáticas (UCI).

Este sistema se desarrolla con el objetivo de mejorar el control y la obtención de reportes referente al proceso de control de la guardia obrera de la Universidad de las Ciencias Informáticas. Para el desarrollo de este sistema se utilizó como lenguaje de programación PHP y otros lenguajes como HTML, JavaScript y las Hojas de Estilo (CSS) para el diseño. Se utilizó la metodología XP, como herramienta de diseño Dreamweaver MX y como gestor de bases de datos PostgreSQL, se utilizó además la librería ExtJs y como servidor web Apache.

Características:

- Es independiente del sistema operativo donde se ejecute y presenta un requerimiento de hardware mínimo.
- Brinda una amplia información sobre todo el tema perteneciente al control de la guardia obrera.

- Disminuye el trabajo manual y agiliza el flujo de información, ahorrando así tiempo y recursos a la universidad.
- Ofrece la posibilidad de realizar reportes semanalmente relacionados con todo el proceso de control de la guardia.
- Cuenta con una interfaz web amigable y fácil de usar permitiendo que los usuarios puedan disfrutar de sus servicios.(10)

Este sistema, aunque incluye dentro de sus funcionalidades el control de la guardia obrera, se centra solamente en este proceso. Por lo tanto, no realiza las funcionalidades antes expuestas relacionadas con el proceso de planificación de la guardia.

En la actualidad los sistemas antes mencionados no están en funcionamiento por no cumplir con la nueva restructuración, planificación y organización de la guardia.

1.3.3 Valoración de los sistemas informáticos analizados.

Las investigaciones realizadas arrojaron como resultado que no existe ningún sistema que cumpla con todas las características definidas para la planificación de la GOE. Aunque cubran escenarios comunes, gestionan y planifican dependiendo de sus características propias; no obstante, pueden ser utilizadas algunas de las concepciones brindadas para la realización de sus funcionalidades, sirviendo de ayuda sus respectivas interfaces de usuario.

Los sistemas estudiados a nivel mundial incluyen funcionalidades necesarias para la realización del proceso de planificación de la guardia, pero tienen la desventaja que utilizan últimas tecnologías disponibles en el mercado, privan el acceso al código fuente del programa y niegan el derecho de poder copiar alguna porción de código o interfaz de usuario y modificarlo de acuerdo con las necesidades deseadas, por lo que la descripción de cómo realizan estos procesos y las características de dicho software, así como la documentación, inclúyase manuales de usuario y vistas de diseño son de carácter privativo, por lo que no cumplen con el paradigma de independencia tecnológica por el cual nuestro país está abogando.

Estos elementos evidencian la imperiosa necesidad de elaborar una solución que responda acertadamente a los procesos de planificación y control de la GOE en la Facultad 3.Con la nueva propuesta de sistema se pretende lograr que estos procesos se realicen de manera adecuada.

1.4 Técnicas de captura de requisitos

Existen varias técnicas para llevar a cabo el proceso de captura de requisitos. Las principales son:(11)

- **Entrevistas y cuestionarios:** permite al equipo de desarrollo interpretar las necesidades del cliente. Pueden ser lo mismo provechosos o no ya que dependen de las habilidades del entrevistador y los entrevistados para obtener información con la mayor calidad posible.
- **Desarrollo conjunto de aplicaciones:** proviene del inglés *JointApplicationDevelopment* (JAD) y es una técnica exploratoria que, aunque puede ser muy costosa por la cantidad de personal que involucra, incluye a los usuarios como participantes activos en el proceso de desarrollo de sistemas. De esta técnica puede surgir una declaración bastante fiel de los requisitos.
- **Tormenta de ideas:** consiste en la acumulación de ideas sin prejuicios y valoraciones que puedan descartarlas y aunque no evidencia los detalles concretos que se pueden necesitar del sistema, es muy común en los comienzos del proceso de ingeniería de requisitos.
- **Mapas conceptuales:** es otra técnica bastante común, usada para la representación gráfica de las ideas y sus relaciones.
- **Escenarios:** valiosos medios para proporcionar contexto a las exigencias del consumidor. Proporcionan un marco para preguntas sobre tareas del usuario permitiendo preguntas "*y si"* y *"cómo se hace esto".*
- **Comparación de terminología:** se utiliza para complementar otras técnicas, pero consiste en identificar todos los términos con los que se trabajará durante el desarrollo del sistema. Para su correcta utilización, es necesario identificar el uso de términos diferentes para los mismos conceptos (correspondencia), misma terminología para diferentes conceptos (conflictos) o cuando no hay concordancia ni en el vocabulario ni en los conceptos (contraste).
- **Reuniones:** el propósito de éstas es intentar alcanzar un efecto aditivo por el que un grupo de gente puede obtener más penetración en los requisitos del software que trabajando individualmente. Ellos pueden inspirarse y refinar las ideas que pueden ser difíciles de traer a la superficie usando entrevistas. Otra ventaja es que dejan a los stakeholders[1] reconocer donde hay requisitos en conflicto. Cuando se aplica bien, esta técnica puede resultar en un rico y constante sistema de requisitos que difícilmente sería realizable de otro modo.
- **Observación:** los ingenieros de software aprenden sobre las tareas del usuario sumergiéndose en la observación de cómo los usuarios obran recíprocamente con su software. Estas técnicas son relativamente costosas, pero son instructivas porque ilustran que muchas tareas del usuario y

[1] Persona o entidad interesada en la realización de un proyecto o tarea.

procesos del negocio son demasiado sutiles y complejos para que sus agentes los describan fácilmente.

1.4.1. Técnicas de validación de requisitos

Las técnicas de validación de requisitos descritas a continuación son las que se encuentran en el libro "*Ingeniería de Software*" v.7 de Ian Sommerville:(11)

- **Revisiones técnicas formales:** los requisitos son analizados sistemáticamente por un equipo de revisores, en busca de anomalías y/u omisiones.
- **Prototipos de interfaz de usuario**: se muestra un modelo ejecutable del sistema a los usuarios finales y los clientes, para que puedan ver si dicho modelo cumple con sus necesidades reales.
- **Generación de casos de prueba:** la elaboración de casos de prueba puede revelar los problemas con que puede contar un requisito y es parte fundamental de la programación externa.

1.5. Modelo de desarrollo de software

Para la construcción del sistema se utilizará el modelo de desarrollo de software propuesto por el centro CEIGE. Dicho modelo está basado en componentes y en él se muestran los diferentes artefactos que se deben generar en la medida que se construye el software.

Desarrollo basado en componentes: lleva a alcanzar un mayor nivel de reutilización de software, aún en contextos distintos a aquellos para los que fue diseñado. Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes integrados. Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema. Dado que un componente puede ser construido y luego mejorado continuamente, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.(12)

1.6. Tecnologías y herramientas para el desarrollo

Como parte del proceso de investigación se estudiaron las tecnologías y herramientas definidas por el centro CEIGE, las cuales se utilizarán en el desarrollo del sistema, éstas son:

1.6.1. Marco de trabajo

Las tecnologías con las que cuenta el marco de trabajo Sauxe son:

ExtJS 2.2: es una librería JavaScript que permite construir aplicaciones complejas en internet. Permite realizar completas interfaces de usuario, fáciles de usar, muy parecidas a las conocidas aplicaciones de escritorio, posibilitando a los desarrolladores concentrarse en la funcionalidad de las aplicaciones en vez de en las advertencias técnicas.

Ventajas:

- ✓ Permite crear aplicaciones complejas utilizando componentes predefinidos.
- ✓ Relación entre Cliente-Servidor balanceado: se distribuye la carga de procesamiento, permitiendo que el servidor pueda atender más clientes al mismo tiempo.
- ✓ Eficiencia de la red: disminuye el tráfico en la red, pues las aplicaciones cuentan con la posibilidad de elegir qué datos desea trasmitir al servidor y viceversa (este criterio puede variar con el uso de aplicaciones de pre-carga).
- ✓ Comunicación asíncrona: en este tipo de aplicación el motor de hacer puede comunicarse con el servidor sin necesidad de estar sujeto a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta.(13)

Doctrine 1.2.1: "Potente y completo sistema de mapeo relacional de objetos (por sus siglas en inglés ORM, Object Relation Mapper). Permite convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, es decir, las tablas de la base de datos pasan a ser clases y los registros objetos se pueden manejar con facilidad.

Utilizar un ORM tiene una serie de ventajas que facilitan tareas comunes y de mantenimiento:(14)

- ✓ **Reutilización:** permite llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.
- ✓ **Encapsulación:** la capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.
- ✓ **Portabilidad:** utilizar una capa de abstracción permite cambiar en mitad de un proyecto de una base de datos MySQL a una Oracle sin ningún tipo de complicación. Esto es debido a que no se utiliza una sintaxis MySQL, Oracle o SQLite para acceder al modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de bases de datos.
- ✓ **Seguridad:** suelen implementar mecanismos de seguridad que protegen nuestra aplicación de los ataques más comunes como SQL Injections.

- ✓ **Mantenimiento del código:** gracias a la correcta ordenación de la capa de datos, modificar y mantener el código es una tarea sencilla."(14)

Zend Framework 1.7: es un framework de código abierto para desarrollar aplicaciones web y servicios web con PHP 5. La estructura de los componentes de Zend Framework es algo único; cada componente está construido con una baja dependencia de otros componentes. Como objetivo, Zend Framework persigue una meta muy clara: simplicidad, ante todo. Busca tener una API muy fácil de aprender, de manera que como desarrolladores se pueda comenzar a escribir aplicaciones rápidamente. Esta simplicidad se ve representada en tres puntos:

- ✓ Simplicidad en el uso. Menos código y más simple de leer.
- ✓ Simplicidad representa código más estable y con menos probabilidad de error.
- ✓ Simplicidad para mantener luego el código.

Ofrece un gran rendimiento y una robusta implementación MVC[2], una abstracción de base de datos fácil de usar, y un componente de formularios que implementa la prestación de formularios HTML, validación y filtrado para que los desarrolladores puedan consolidar todas las operaciones usando de una manera sencilla la interfaz orientada a objetos.(15)

Zend Framework permite una aplicación sencilla, rápida y ágil proceso de desarrollo web, ya que cuenta con las siguientes características:

- ✓ Tiene una arquitectura flexible, en el que todos los componentes están débilmente atados. Además ofrece a los desarrolladores de PHP una arquitectura de aplicaciones que le posibilita utilizar cualquiera de los dos componentes del marco por separado o combinarlos según sus preferencias y necesidades.
- ✓ Proporciona Modelo-Vista-Controlador (MVC), que permite dividir el desarrollo de aplicaciones web en la presentación, lógica de negocio, y las capas de acceso a datos. Tal patrón MVC es útil cuando varios desarrolladores de PHP y diseñadores web tienen como tarea de proyecto trabajar sobre una misma aplicación.

[2] Modelo-Vista-Controlador

- ✓ Ofrece soporte AJAX[3]. Esta característica permite que con la utilización de Zend Framework los desarrolladores de aplicaciones web puedan convertir los datos XML en formato JSON, que es apoyado por AJAX basadas en aplicaciones de front-end.(16)

Sauxe 2.2: posee una estructura que facilita la reutilización de los diferentes componentes y es de fácil entendimiento para quienes desarrollen sobre el mismo. Sauxe utiliza ExtJS para implementar la capa de presentación, se apoya en Zend-Ext, una extensión de Zend Framework para el desarrollo de la lógica del negocio y para la gestión de los datos que maneja Doctrine. Utiliza como estilo arquitectónico el Modelo-Vista-Controlador. También tiene como propósito insertar la programación orientada a aspectos así como la inversión de controles.(17)

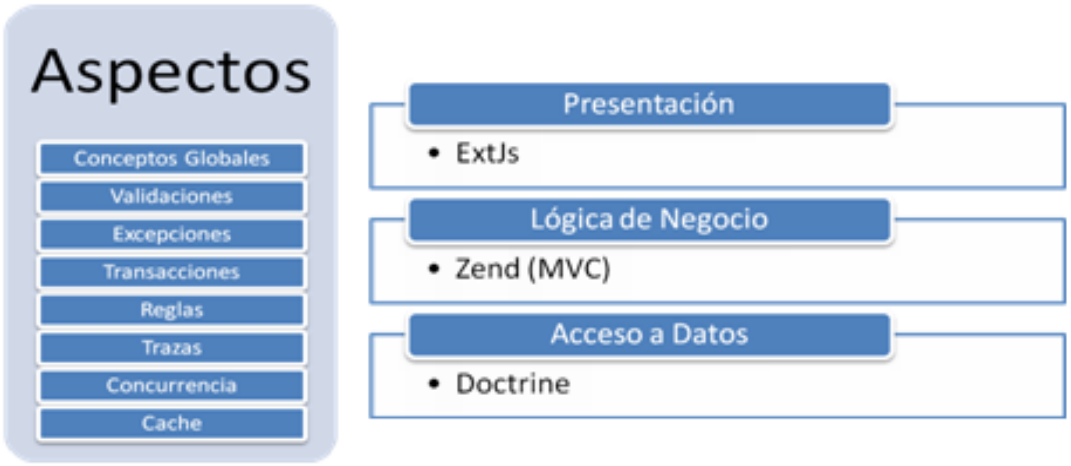

Figura 2. Arquitectura de Sauxe.

El marco de trabajo Sauxe, fusionado bajo tecnología totalmente libre (PHP, postgresql, apache, entre otras). Posee el desarrollo de tecnologías propias basadas en otros frameworks como ZendFramework, Doctrine, y que unifica temas que actualmente están en desarrollo y que son totalmente novedosos en las aplicaciones de gestión similares. Por otra parte este marco tecnológico soporta actualmente una aplicación de más de 15 componentes (los cuales responden a más de 1000 requisitos de negocio), garantizando las transacciones de negocio, la integración de los componentes de negocio, la exportación e importación de información, el tratamiento de las excepciones a nivel de negocio, las validaciones de negocio, la trazabilidad de las operaciones, el uso de los aspectos generales, la colaboración entre subsistemas y la integración con una aplicación externa (Reporteador dinámico). Sauxe se desarrolla en

[3] Es una técnica de desarrollo web para crear aplicaciones interactivas.

función de cumplir con todos los escenarios arquitectónicos que pueden encontrarse en la implementación de un sistema ERP[4].(17)

1.6.2. Lenguajes de programación

"Un lenguaje de programación puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente. Aunque muchas veces se usa lenguaje de programación y lenguaje informático como si fuesen sinónimos, no tiene por qué ser así, ya que los lenguajes informáticos engloban a los lenguajes de programación y a otros más, como, por ejemplo, el HTML**.**"(18)

PHP 5.2: es el lenguaje que se empleará para programar del lado del servidor. Es un lenguaje de programación interpretado, completamente orientado al desarrollo de aplicaciones web dinámicas. Es gratuito, fácil de usar y aprender, portable, de código abierto y multiplataforma. Presenta interfaces para una gran cantidad de sistemas de base de datos diferentes, así como bibliotecas incorporadas para muchas tareas web habituales. Sin embargo este lenguaje debido a su flexibilidad presenta como principal inconveniente, que mal utilizado, puede convertir al sitio en un punto de fácil de acceso a piratas informáticos. Por otra parte con respecto a otros lenguajes de programación como Java o C++ es mucho menos robusto y la programación orientada a objetos es aún muy deficiente para aplicaciones grandes.(19)

JavaScript 1.6: "es un lenguaje de programación que se puede utilizar para construir sitios web y para hacerlos más interactivos. Aunque comparte muchas de las características y de las estructuras del lenguaje Java, fue desarrollado independientemente. El lenguaje JavaScript puede interactuar con el código HTML, permitiendo a los programadores web utilizar contenido dinámico. Por ejemplo, hace fácil responder a los acontecimientos iniciados por usuarios (como introducción de datos en formularios) sin tener que utilizar interfaz de entrada común (CGI[5]). El lenguaje JavaScript es de código abierto, por lo que cualquier persona puede utilizarlo sin comprar una licencia."(20)
Es utilizado para crear pequeños algoritmos y funciones, encargados de realizar acciones dentro del ámbito de una página web. Es simple y permite concretar funcionalidades con alto grado de rapidez. Su

[4] Planificación de Recursos Empresariales (Enterprise Resource Planning).
[5] Interfaz de entrada común (Common Gateway Interface).

empleo es recomendable aún para personas con poca experiencia en programación, pues su sencillez permite asimilarlo y practicarlo con facilidad.(20)

HTML: es el lenguaje que se utilizará para definir páginas clientes de la aplicación. Se compone por un conjunto de etiquetas utilizadas para definir y ubicar los distintos elementos que componen una página web. Como un lenguaje de marcación de elementos para la creación de documentos hipertexto, HTML puede describir hasta un cierto punto la apariencia de un documento. Puede incluir uno o varios scripts, como por ejemplo: JavaScript o PHP, los cuales pueden afectar el comportamiento del HTML. Su principal desventaja es que todos los navegadores no interpretan el código HTML de la misma manera.(21)

Hojas de Estilo en Cascada (CSS): las hojas de estilo en cascada serán utilizadas para representar todo lo referente a los estilos (dígase tamaños, colores, iconos, imágenes, tipografías, espacios y bordes). Constituyen el estándar para la inserción de estilos a documentos estructurados, como por ejemplo, páginas HTML o XML. El objetivo de la definición de este estándar del W3C[6] es permitir la separación entre las normas de presentación y el propio contenido a mostrar.(21)

1.6.3. Lenguajes de modelado

Se denomina lenguaje de modelado de objetos al conjunto estandarizado de símbolos y de modos de disponerlos para modelar un diseño de software.(21) Para la modelación de la solución que se propone se emplearán los siguientes lenguajes de modelado:

UML: el Lenguaje de Modelado Unificado (Unified Modeling Language) como notación orientada a objetos, se empleará con el fin de especificar y documentar un sistema de software, de un modo estándar incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema. Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representa la arquitectura del proyecto. UML implementa un lenguaje de modelado común para todos los desarrollos por lo que se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.(21)

[6] El World Wide Web Consortium, abreviado W3C, es un consorcio internacional que produce recomendaciones para la World Wide Web.

BPMN: BPMN (Business Process Modeling Notation) es empleado en el desarrollo de la solución que se propone para presentar gráficamente las diferentes etapas del proceso. Este estándar de modelado de procesos de negocio ha sido diseñado específicamente para coordinar la secuencia de procesos y los mensajes que fluyen entre los diferentes procesos participantes. BPMN a diferencia del lenguaje de modelado unificado (UML) toma un perfil orientado a procesos en el modelado de sistemas, al ser compatibles entre sí, ayudan a modelar con mayor precisión la situación actual y deseada en los procesos de negocio del cliente.(21)

1.6.4. Gestor de base de datos

PostgreSQL 9.1: es un servidor de base de datos relacional libre, bajo licencia BSD[7]. La licencia BSD al contrario que la GPL[8] permite el uso del código fuente en software no libre. Es el sistema de gestión de bases de datos de código abierto más avanzado del mundo y en sus últimas versiones posee muchas características que solo se podían ver en productos comerciales de alto calibre. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.
Cuenta con una gran comunidad de desarrollo en Internet, su código fuente está disponible sin costo alguno y es multiplataforma.(22)

1.6.5 Navegador Web

Mozilla Firefox 3.6.24 o superior: es un navegador web libre y de código abierto, lo que quiere decir que todo desarrollador puede modificar el código y así poder mejorarlo. Se trata de un navegador multiplataforma, es decir, que se encuentra disponible en diferentes versiones y así es posible utilizarlo con Microsoft Windows, Mac OS X, y también con GNU/Linux, coordinado por la Corporación Mozilla y la Fundación Mozilla. Usa el motor Gecko para renderizar páginas webs, el cual implementa actuales y futuros estándares web. A partir de agosto de 2012 firefox tiene aproximadamente un 23% de la cuota de mercado, convirtiéndose en el tercer navegador web más usado.(23)

- Características del **Mozilla Firefox:**

Sus características incluyen navegación por pestañas, corrector ortográfico, búsqueda progresiva, marcadores dinámicos, un administrador de descargas, bloqueo de ventanas emergentes no deseadas

[7] Distribución de Software Berkeley (Berkeley Software Distribution).
[8] Licencia Pública General

incorporado (antipop-up's), buscadores en la barra de herramientas, navegación privada, es un software libre, con licencia GPL e integración del motor de búsqueda que desee el usuario. No existen virus elaborados para demostrar la fragilidad e inestabilidad de Mozilla Firefox ni de ninguna otra aplicación libre. Se puede utilizar Firefox bajo cualquier sistema operativo, ya se trate de Linux o de Windows.

Proporciona un entorno para los desarrolladores web en el que se pueden utilizar herramientas incorporadas, como la Consola de errores, *Scratchpad* (para probar código JavaScript), el Inspector DOM[9], o extensiones como Firebug.(24)

1.6.6. Servidor Web

Apache 2.2: "es un servidor web de software libre desarrollado por la Apache Software Foundation cuyo objetivo es servir o suministrar páginas web (en general, hipertextos) a los clientes web o navegadores que las solicitan."

La arquitectura utilizada es cliente/servidor, es decir, el equipo cliente hace una solicitud o petición al equipo servidor y éste la atiende.

"El servidor web **Apache 2.2** proporciona contenidos al cliente web o navegador como:

- ✓ **Páginas estáticas**: es el uso más generalizado que se hace de un servidor web. De esta forma se transfieren archivos HTML[10], imágenes, etc. y no se requiere un servidor muy potente en lo que al hardware se refiere.
- ✓ **Páginas dinámicas**: la información que muestran las páginas que sirve apache cambia y se obtiene a partir de consultas a bases de datos u otras fuentes de datos. Son, por tanto, páginas con contenido dinámico, cambiante."(25)

- ➢ Características del **Apache:**

Existen una serie de características que convierten a Apache en uno de los servidores web más utilizados, como son el tener el código fuente abierto, mantener una evolución rápida y continuada de versiones, poder ser utilizado por desarrolladores de cualquier plataforma, y además, es gratuito. Una característica importante a señalar es que Apache permite trabajar con servidores virtuales tanto con direcciones IP así como con nombres virtuales por lo que se podría convertir en un servidor proxy. Apache puede trabajar

[9] Modelo de Objetos del Documento
[10] Protocolo de transferencia de hipertexto.

con otros lenguajes de respuesta del servidor como Perl y Java (servlets) siempre y cuando se añadan los módulos necesarios en el fichero de configuración.(26)

1.7 Herramientas para el modelado

Visual Paradigm 6.4: es una herramienta que facilita y permite a las organizaciones visualizar, diseñar, integrar y distribuir sus aplicaciones empresariales de misión crítica. Es una herramienta de UML CASE y Business Process Modeling Notation (BPMN). Además de soporte de modelado, ofrece generación de informes y capacidades de ingeniería de código incluyendo la generación de código, puede crear ingeniería inversa del código de diagramas y ofrecer ida y vuelta de ingeniería para diversos lenguajes de programación. Esta herramienta ayuda a los desarrolladores de software a crear un modelo de excelencia durante la creación y distribución del proceso de desarrollo de aplicaciones, esto es porque maximiza y acelera al equipo de desarrollo en conjunto con las contribuciones de cada programador.

Entre sus principales características se destacan:

- ✓ Modelado de base de datos: proporciona una mayor documentación de la base de datos y diagramas de mapeo de relación de objetos.
- ✓ Interoperabilidad: intercambio de diagramas UML y modelos con otras herramientas, usando representaciones industriales comunes.
- ✓ Integración IDE.
- ✓ Modelado de requisitos.
- ✓ Modelo de procesos de negocios: visualización, improvisación y entendimiento de procesos con la herramienta BPMN.
- ✓ Generador de código.
- ✓ Generador de documentación.(27)

1.7.1. IDE de desarrollo

Los Entornos de Desarrollo Integrado, IDE por sus siglas en inglés (Integrated Development Environment), son un entorno de programación que ha sido empaquetado como un programa de aplicación, o sea, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. Los IDE ofrecen un marco de trabajo amigable para la mayoría de los lenguajes de programación, tales como: C++, PHP, Python, Java, C#, Delphi, Visual Basic, etc. El IDE seleccionado para el trabajo con los lenguajes de programación para la realización del sistema es:(28)

Netbeans 6.9: "es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extenderlo. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso. "(29)

Características de **Netbeans 6.9:**

- ✓ Mejoras en el formateo de código.
- ✓ Refactor y búsqueda de usos para CSS y lenguajes tipo HTML.
- ✓ Auto-completado de código y links para atributos de CSS.
- ✓ Soporte para PHP y Zend Framework.(30)

1.8 Arquitectura de Software

Para esta investigación se adoptará la definición brindada por la Arquitectura - IEEE 1471-2000 la cual plantea lo siguiente: "la arquitectura del software es la organización fundamental de un sistema, formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución".

1.8.1 Estilo arquitectónico Modelo-Vista-Controlador (MVC)

MVC es un estilo arquitectónico usado principalmente en aplicaciones que manejan gran cantidad de datos y transacciones complejas donde se requiere una mejor separación de los conceptos para que el desarrollo esté estructurado de una mejor manera. Facilita la programación en diferentes capas de forma paralela e independiente. MVC sugiere la separación del software en 3 estratos.(31)

Modelo: esta capa es la representación específica de la información con la cual el sistema opera. Se limita a lo relativo de la vista y su controlador facilitando las presentaciones visuales complejas.

El Modelo es el responsable de:

- ➢ Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
- ➢ Definir las reglas de negocio.
- ➢ Si se encuentra ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

Vista: esta capa presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Las Vistas son responsables de:

- Recibir datos del modelo y mostrarlos al usuario.
- Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
- Pueden dar el servicio de actualización, para que sea invocado por el controlador o por el modelo.

Controlador: este responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y notifica a la vista.

El Controlador se encarga de:

- Recibir los eventos de entrada.
- Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo o a las vistas.(31)

En la **figura 3** se muestra el funcionamiento del estilo arquitectónico Modelo-Vista-Controlador.

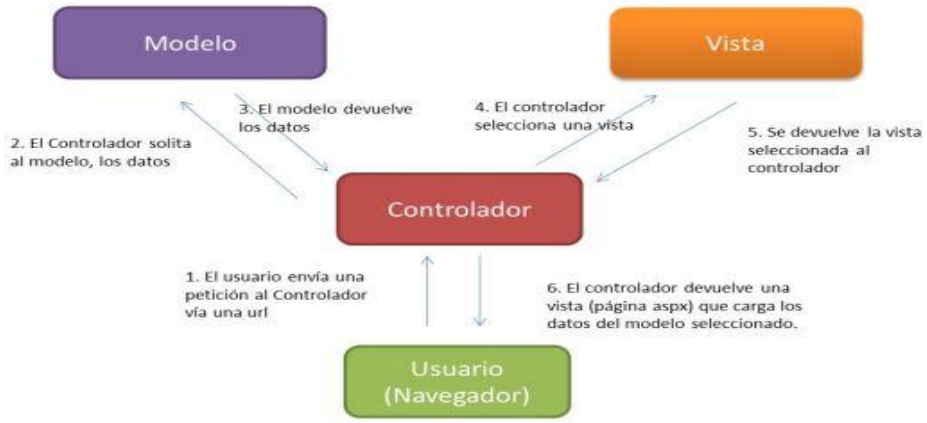

Figura 3. Patrón Modelo-Vista-Controlador.

1.9. Patrones de Diseño

Un patrón de diseño es una solución a un problema de diseño que se presenta en el desarrollo de un software. Entre sus características se debe mencionar que es reutilizable y aplicable a diferentes problemas de diseño en distintas circunstancias.

1.9.1. Patrones GRASP

General Responsibility Assignment Software Patterns (GRASP), es un acrónimo que en español significa Patrones Generales de Software para asignar Responsabilidades; describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones.(32) Según Craig Larman en su libro "*UML y patrones. Introducción al análisis y diseño orientado a objetos.*", existen 9 patrones generales de software para asignar responsabilidades; ellos son:(11)

Tabla 1. Patrones de diseño GRASP.

Patrón	Descripción
Experto	Asigna una responsabilidad al experto en información: la clase que posee la información necesaria para cumplir con la responsabilidad.
Creador	Asigna a la clase B la responsabilidad de crear una instancia de clase A.
Controlador	Asigna la responsabilidad de administrar un mensaje de eventos del sistema a una clase.
Bajo acoplamiento	Asigna responsabilidades de modo que se mantenga bajo acoplamiento.
Alta cohesión	Asigna responsabilidad de modo que se mantenga alta cohesión.
Polimorfismo	Cuando varía el tipo de alternativas o comportamientos relacionados, asigna la responsabilidad del comportamiento, mediante operaciones polimórficas, a los tipos en que varía el comportamiento.
Fabricación pura	Asigna un conjunto muy alto de cohesión de responsabilidades a una clase artificial que no represente nada en el dominio del problema, a fin de brindar soporte a una alta cohesión, a bajo acoplamiento y a la reutilización.
Indirección	Asigna la responsabilidad a un objeto intermedio para que medie entre otros componentes o servicios, de modo que no se acoplen directamente.
No hables con extraños (Ley de Demeter)	Asigna la responsabilidad al objeto directo del cliente para colaborar con el objeto indirecto, de modo que el cliente no necesita conocer el objeto indirecto.

1.9.2 Patrones GOF

Los patrones de diseño GOF[11] se dieron a conocer a principios de los años 90 con el libro “Design Patterns. Elements of Reusable Object-Oriented Software.”. En dicho libro se hace una recopilación de 23 patrones de diseño comunes, clasificados en tres grupos de acuerdo a su naturaleza:

- Creacionales: concierne al proceso de creación de objetos.
- Estructurales: tratan la composición de clases y/o objetos.
- De comportamiento: caracterizan las formas en la que interactúan y reparten responsabilidades las distintas clases u objetos.(11)

1.10 Herramientas para el control de versiones

SVN (Subversión) 1.6.6: es un sistema de control de versiones usado para que varios desarrolladores puedan trabajar en un mismo proyecto en forma más o menos ordenada. Tiene una arquitectura cliente-

[11] Gang of Four (Banda de los cuatro): Nombre con el que se conoce comúnmente a los autores del libro Design Patterns.

servidor con controles de concurrencia para cuando varios desarrolladores están trabajando en el mismo archivo y funciona más o menos así. En algún servidor se monta un repositorio SVN. En este lugar se van a registrar los cambios (revisiones) y los logs[12] que se vayan generando. El cliente de SVN se baja una copia local de alguna revisión (generalmente la última), el desarrollador hace los cambios y los sube al servidor para que estén disponibles para los otros desarrolladores (además de generar un log con un comentario de que cosa modificó, para qué, etc.).(33)

1.11. Conclusiones del capítulo

Con la realización del presente capítulo se concluye lo siguiente:

- Los conceptos utilizados ayudan a lograr un mejor entendimiento de la investigación científica en cuestión, conformando la fundamentación teórica que sustenta el desarrollo de la investigación y propiciando a que el lector adquiera conocimientos acerca del tema a tratar.
- La realización del estudio del estado del arte acerca de los sistemas de gestión y los procesos de planificación y control de la guardia, existentes tanto en Cuba como en el mundo, permite arribar a la conclusión de que no existe un sistema de gestión que satisfaga las necesidades planteadas por la Facultad 3 en aras de una mejor organización y concepción de estos procesos.
- La descripción de las tecnologías, lenguajes y herramientas con las cuales se va a desarrollar la solución propuesta permite para lograr un mejor entendimiento de la misma.

[12] Es un registro de actividad de un sistema, que generalmente se guarda en un fichero de texto, al que se le van añadiendo líneas a medida que se realizan acciones sobre el sistema.

CAPÍTULO II: ANÁLISIS Y DISEÑO DE LA APLICACIÓN

2.1. Introducción

En el presente capítulo se realizará la descripción de la propuesta para dar solución a la problemática existente. Para ello se formularán todas las características que tendrá el sistema, partiendo del estudio de los procesos actuales del negocio donde se van a identificar los actores, el rol y la responsabilidad que tiene cada uno con el sistema. Además se lleva a cabo el modelado de negocio de los procesos presentes en el sistema, además de realizarse una descripción de dichos procesos. Se identifican los requisitos funcionales y no funcionales, a partir de los cuales se definen diagramas de procesos, lo que permite dar paso al diseño donde se realizan los diagramas de clases del diseño, diagrama de componentes, diagramas de secuencias, modelo conceptual, modelo físico de datos y además, se especificará la utilización de un conjunto de patrones dentro del diseño del sistema.

2.2. Objeto de informatización

Se define como objeto de informatización el proceso que se realiza para obtener la planificación de la guardia en la Facultad 3. Este proceso se denomina: Realizar planificación de la Guardia Obrera-Estudiantil: el mismo se lleva a cabo cuando la Vicedecana de Administración y Servicios solicita la información necesaria a la Secretaría y a los Jefes de áreas de los trabajadores para realizar la planificación de la guardia.

El negocio que requiere la solución informática a desarrollar en este trabajo comprende el desempeño de procesos relacionados con la planificación de la GOE en el área de la Facultad 3. En este sentido la Vicedecana de Administración y Servicios de la facultad es la encargada de la ejecución y planificación de la guardia en cada una de las áreas en las que está comprendida. Básicamente el desarrollo del negocio del sistema involucra un proceso fundamental: Planificación de la GOE. Seguidamente se especifica con más detalles la secuencia de actividades que se realizan en dicho proceso.

La Vicedecana de Administración y Servicios es la encargada de llevar a cabo la planificación de la GOE en la Facultad 3, la misma la realiza en un modelo que se tiene confeccionado en Microsoft Excel y se le asignan los días, postas y turnos correspondientes a ejecutar la GOE. Posteriormente este modelo es enviado mediante el correo a toda la facultad con el objetivo de dar a conocer los días de guardia de cada uno de los implicados. En caso de existir algún inconveniente con dicha guardia tanto por parte de los trabajadores, como de los estudiantes, la Vicedecana de Administración y Servicios se encarga de

ejecutar los cambios pertinentes en dicha planificación de manera iterativa, hasta lograr la conformidad de todos los necesitados.

2.3. Propuesta del sistema

Debido a las dificultades existentes en la gestión de los problemas relacionados con la Guardia Obrera-Estudiantil de la Facultad 3, se decidió desarrollar un sistema informático que dé solución a estas dificultades. El mismo brindará la posibilidad de gestionar la planificación de la guardia en la Facultad 3, donde el sistema deberá ser capaz de generar el proceso de planificación de la Guardia Obrera-Estudiantil a través de una serie de requisitos y restricciones que debe cumplir para lograr una buena planificación. Además, se tendrán en cuenta algunas limitantes que son de suma importancia para la confección de dicha planificación. Se realizarán cambios o permutas de guardias entre estudiantes y trabajadores, siempre notificándose a través del correo, logrando así conformidad de acuerdo a las necesidades de los estudiantes. En dicho sistema se registrarán las incidencias ocurridas en el transcurso de la guardia día por día.

2.4. Modelado del negocio

La modelación del proceso de negocio permite comprender al grupo de desarrollo los procesos de negocio que se realizan en la entidad actualmente, así como la relación que existe entre ellos; y cómo funciona el negocio que se desea automatizar para tener garantías de que el software desarrollado va a cumplir su propósito.

2.4.1 Mapa de procesos del negocio

El mapa de procesos de negocio ofrece una visión general del sistema de gestión. En él se representan los procesos que componen el sistema así como sus relaciones principales. Dichas relaciones se indican mediante flechas y registros que representan los flujos de información.(34) El mapa de proceso para el sistema de gestión encargado de la planificación de la guardia en la Facultad 3 abarca dos procesos: Realizar Planificación de la Guardia-Obrera-Estudiantil y Realizar Control de la Guardia-Obrera-Estudiantil, donde este primer proceso mencionado necesita del listado de estudiantes y trabajadores y el segundo del documento de planificación de la guardia que genera el proceso Realizar Planificación de la Guardia-Obrera-Estudiantil. Ver la **figura 4.**

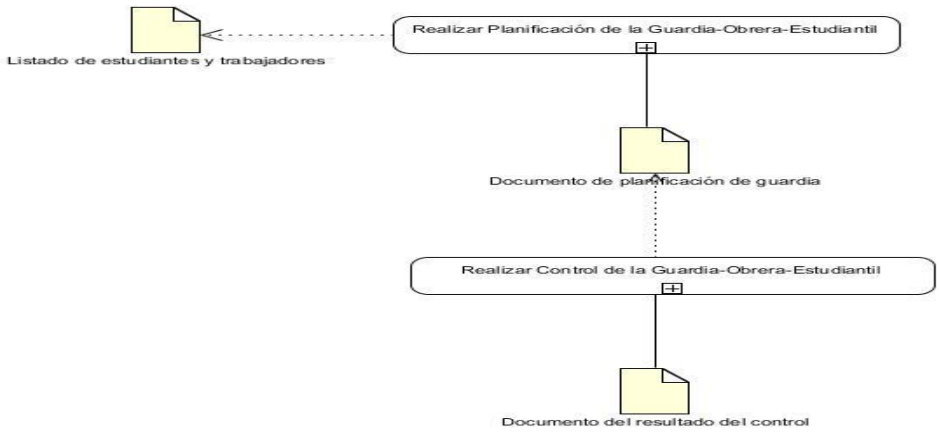

Figura 4. Mapa de proceso de negocio.

2.4.2. Diagrama del proceso de negocio

Un proceso de negocio es un conjunto de actividades relacionadas que permiten crear un producto o servicio final y tiene como objetivo fundamental satisfacer al cliente.(35)

El negocio cuenta con dos procesos fundamentales los cuales serán descritos a continuación para un mejor entendimiento de los mismos.

Realizar Planificación de la Guardia-Obrera-Estudiantil: en este proceso se observa cómo se lleva a cabo la planificación, paso a paso. Primeramente la Vicedecana de Administración y Servicios es la encargada de gestionar una serie de listados con los estudiantes y trabajadores habilitados para realizar la guardia y de esta manera llegar a conformar la planificación de la misma en la Facultad 3. Estos listados son solicitados a la Secretaria Docente y a los Jefes de las distintas áreas respectivamente, donde son enviados a la Vicedecana de Administración y Servicios para tener una noción de las personas que están en condiciones de realizar la guardia. Después de realizada dicha planificación es enviada vía correo electrónico a toda la facultad, ver **figura 5.**

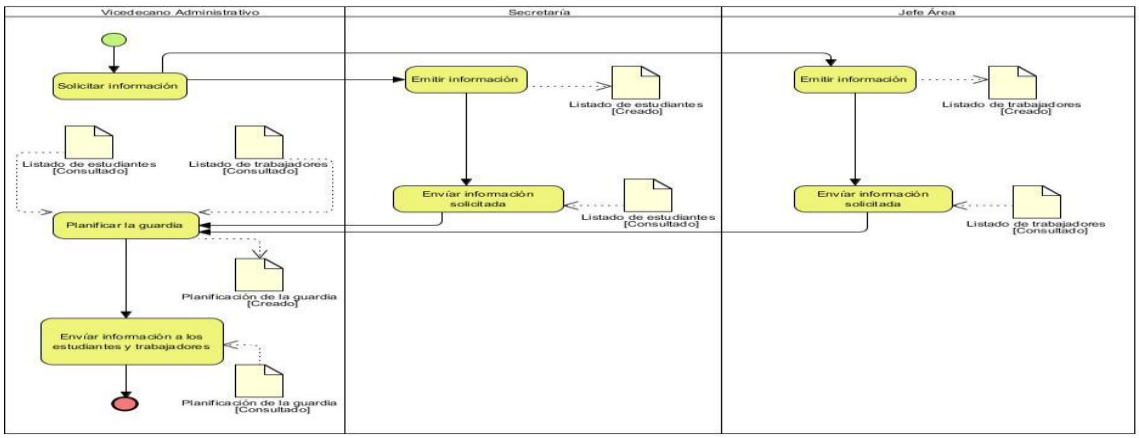

Figura 5. Proceso de negocio: Realizar Planificación de la Guardia-Obrera-Estudiantil.

Tabla 2. Descripción del proceso de negocio: Realizar Planificación de la Guardia-Obrera-Estudiantil.

Objetivo	Realizar la planificación de la guardia de los estudiantes y trabajadores de la Facultad 3.
Evento(s) que lo genera(n)	N/A.
Pre condiciones	
Marco legal	N/A.
Clientes externos	N/A.
Entradas	Listado de estudiantes y trabajadores.
Flujo de eventos	
Flujo básico Análisis Financiero	
1.	Solicitar información: la Vicedecana de Administración y Servicios solicita la información necesaria de los estudiantes a secretaría y de los trabajadores a los Jefes de Áreas para poder visualizar quién no tiene problemas y está apto para realizar la guardia.
2.	Emitir información: secretaría y los Jefes de Áreas organizan los listados.
3.	Enviar información: secretaría y los Jefes de Áreas mandan los listados actualizados solicitados por la Vicedecana de Administración y Servicios.
4.	Planificar la guardia: la Vicedecana de Administración y Servicios realiza la planificación de la guardia de los estudiantes y trabajadores de la Facultad 3.
5.	Enviar información a los estudiantes y trabajadores: la Vicedecana de Administración y Servicios envía el Excel con la planificación de la guardia a través del correo a toda la facultad.
Pos-condiciones	
1.	Se obtiene un Excel con la planificación de la guardia de los trabajadores y estudiantes de la Facultad 3.
Salidas	Reporte impreso con la planificación de la guardia de los trabajadores y estudiantes de la Facultad 3.
1.	
Flujos paralelos	
Salidas	
Flujos alternos	
N/A	
Salidas	
N/A	
Asuntos pendientes	
N/A.	

Realizar Control de la Guardia-Obrera-Estudiantil: proceso que tomando como entrada el documento de planificación de guardia, se realiza el control por todas las postas en cada uno de los turnos, y al culminar el horario de guardia por día, se elabora un informe con los incumplimientos de la misma y se toman medidas con los implicados.

El diagrama del modelado del proceso de negocio Realizar Control de la Guardia-Obrera-Estudiantil y la descripción del mismo se encuentran en los Anexo 1 y Anexo 2 respectivamente.

2.5. Modelo conceptual

En el modelo conceptual se trata de obtener el esquema conceptual de la base de datos a partir de la lista descriptiva de objetos y asociaciones identificadas en la organización durante el análisis.(36) Ver **figura 6.**

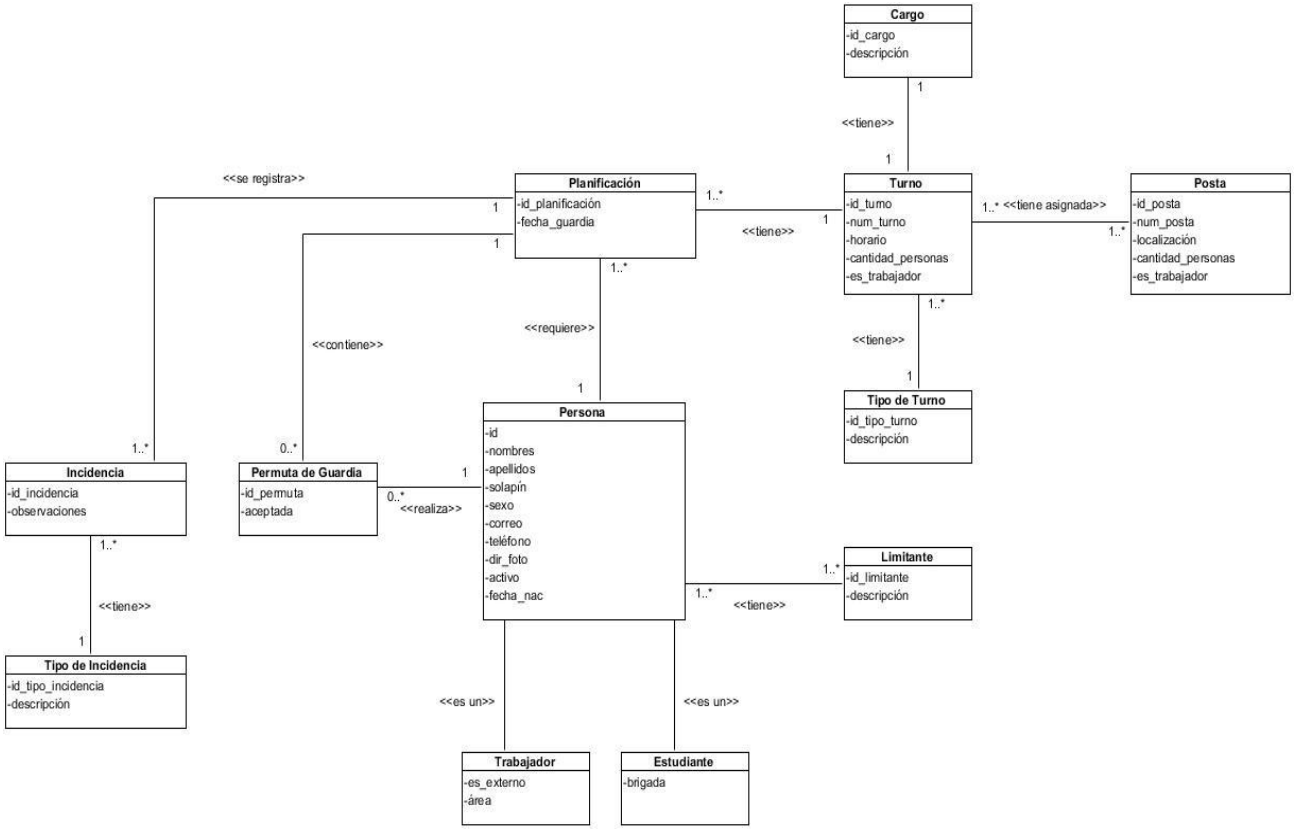

Figura 6. Modelo Conceptual.

1.6. Requisitos

Los requisitos son la base para un desarrollo exitoso así como para una plena conformidad con el entregable final.(37) A continuación se muestra el listado de requisitos definidos para el sistema informático propuesto por la investigación.

Para realizar la captura de los requisitos se hizo necesaria la utilización de algunas técnicas de captura de requisitos. Entre las técnicas para la captura de requisitos utilizadas se encuentra la de realizar varias **Reuniones**, en las que mediante **Mapas conceptuales** y **Tormenta de ideas** se pueda llegar a la

conclusión de cuáles serían los requisitos funcionales de software para el Sistema de GOE a modelar. Teniendo en cuenta la importancia del proceso de validación de requisitos se propone realizar varias **Revisiones técnicas formales** en las que estén presentes la gran mayoría del personal involucrado en la solución y se propone la creación de los **Prototipos de interfaz de usuario** para que el cliente visualice y compruebe si dicho modelo cumple con sus necesidades reales y la **Generación de casos de prueba** para probar cada requisito identificado.

1.6.1. Requisitos funcionales

Los requisitos funcionales muestran las características requeridas por el sistema informático propuesto, que expresan una capacidad de acción del mismo, una funcionalidad o comportamiento interno; generalmente expresada en una declaración en forma verbal.(37)

A continuación se listan los requisitos funcionales identificados para el sistema:

RF-1 Generar guardia.

RF-2 Notificar planificación de la guardia.

RF-3 Gestionar grupos de turnos.

RF-4 Gestionar postas.

RF-5 Gestionar turnos.

RF-6 Imprimir planificación de la guardia.

RF-7 Enviar limitantes para la guardia.

RF-8 Activar o desactivar personas con limitantes.

RF-9 Mostrar normativas de la guardia e imagen de las postas.

RF-10 Eliminar incidencias del cumplimiento de la guardia.

RF-11 Imprimir incidencias del cumplimiento de la guardia.

RF-12 Eliminar planificación de la guardia.

RF-13 Realizar cambios de guardia.

RF-14 Notificar solicitud de cambio de guardia.

RF-15 Aceptar o rechazar solicitud de cambio de guardia.

RF-16 Notificar solicitud aceptada de cambio de guardia.

RF-17 Confirmar solicitud de cambio de guardia.

RF-18 Registrar incidencia de cumplimiento de la guardia.

RF-19 Eliminar incidencias del cumplimiento de la guardia.

RF-20 Imprimir incidencias del cumplimiento de la guardia.

RF-21 Actualizar plantilla para la guardia.

RF-22 Adicionar tipos de turno para la guardia.

RF-23 Eliminar tipos de turno para la guardia.

RF-24 Adicionar cargos para la guardia.

RF-25 Eliminar cargos para la guardia.

RF-26 Adicionar tipos de incidencias para la guardia.

RF-27 Eliminar tipos de incidencias para la guardia.

RF-28 Adicionar limitantes para la guardia.

RF-29 Eliminar limitantes para la guardia.

Como ejemplo ilustrativo de la descripción de los requisitos funcionales se encuentra en los Anexo 3, Anexo 4 y Anexo 5 la descripción de los requisitos Generar guardia, Registrar incidencia de cumplimiento de la guardia y Realizar cambios de guardia respectivamente.

1.6.2. Requisitos no funcionales

Los requisitos no funcionales definen propiedades del sistema informático que se desea como producto final. Estos requisitos son normalmente a los que debe apuntar la arquitectura y si estos no son cumplidos, el sistema informático propuesto puede no funcionar o el cliente simplemente no aceptar el producto.(37)

Apariencia o interfaz externa.

- ✓ El sistema tiene que ofrecer una interfaz amigable, fácil de operar.
- ✓ Debe ser interactivo con el usuario.
- ✓ La comunicación entre el servidor Web y las máquinas clientes será mediante el HTTP y entre el servidor y el directorio activo mediante LDAP.

Usabilidad.

- ✓ Las etiquetas de cada funcionalidad y los campos de cada interfaz tendrán títulos asociados a su función de negocio.
- ✓ Tiene que poseer una interfaz agradable para el cliente.

Rendimiento.

- ✓ La eficiencia del producto estará determinada en gran medida por el aprovechamiento de los recursos y la velocidad de la consultas de la base de datos.
- ✓ El sistema debe ser capaz de funcionar al ser instalados en una misma PC todos sus componentes tales como Gestor de bases de Datos, Aplicación Web, etc.

Disponibilidad:

- ✓ El sistema debe estar disponible las 24 horas del día sin ninguna interrupción.

Mantenibilidad:

- ✓ El sistema debe estar en capacidad de permitir en el futuro su fácil mantenimiento con respecto a los posibles errores que se puedan presentar durante la operación del sistema.

Soporte.

- ✓ Tiene que brindar soporte para grandes volúmenes de datos y velocidad de procesamiento.
- ✓ El sistema tiene que ser multiplataforma.

Confiabilidad y seguridad:

- ✓ Garantizar que las funcionalidades del sistema se muestren de acuerdo al rol del usuario que esté activo.
- ✓ Tiene que brindar una protección contra acciones no autorizadas o que puedan afectar la integridad de los datos.

- ✓ Permitir autenticarse a través de un Servidor de Dominio.
- ✓ El sistema tiene que mantener en todo momento la seguridad de la información asegurando la autenticidad de la misma.

Hardware.

Cliente:

- ✓ Requerimientos mínimos: procesador Pentium IV a 800 MHz, 128mb de memoria RAM y un navegador web.
- ✓ Las computadoras clientes deben estar conectadas en red.

Servidor:

- ✓ Requerimientos mínimos: procesador Dual Core a 3.00 GHz, 1gb de memoria RAM y una capacidad de 40gb de disco duro.
- ✓ El servidor debe tener al menos 1 tarjeta de red para establecer la conexión.

Software.

Cliente:

- ✓ Sistema operativo con interfaz gráfica y soporte para red.
- ✓ Las interfaces deben ser compatibles con Mozilla Firefox 3.0 o superior.

Servidor:

- ✓ Servidor web Apache 2.2.6 o superior.
- ✓ Gestor de base de datos PostgreSQL 8.3 o superior.

2.7 Patrones utilizados en el diseño de la aplicación

Dentro de los patrones GRASP se utilizaron:

- **Experto:** se emplea al trabajar con el marco de trabajo Sauxe y un ejemplo de ello es la inclusión de Doctrine Generator para mapear la base de datos, el cual es el encargado de generar las clases para la gestión de las tablas en dicha base de datos con las responsabilidades debidamente asignadas. Cada clase cuenta con un grupo de funcionalidades que las convierte en experta de la información de la tabla a la que representa. Esto se puede observar en la clase (class DatPersona extends BaseDatPersona) la cual es la encargada de realizar las acciones directamente a la base de datos.

- **Creador:** este patrón se evidencia en la clase (class SistemadeguardiaController extends ZendExt_Controller_Secure), la cual contiene las acciones y es la encargada de ejecutarlas. Dentro de esta existen varias funcionalidades en las cuales se crean varios objetos o instancias de las clases que representan cada una de las tablas existentes en la base de datos.

- ✓ **Controlador:** la clase controladora *SistemadeguardiaController* se encarga de llevar el control de todos los eventos relacionados con el negocio. Implementa las funcionalidades que dan respuesta a las peticiones del usuario.

- ✓ **Alta Cohesión:** Sauxe presenta entre sus principales características la organización del trabajo en cuanto a estructura y responsabilidades bien definidas, esto permite que se trabaje con las clases con una alta cohesión. Un ejemplo de esto es la clase (class SistemadeguardiaController extends ZendExt_Controller_Secure), la cual delega funciones específicas a otras clases, encargándose solo de definir las acciones a realizar.

- ✓ **Bajo Acoplamiento:** este patrón se evidencia en la aplicación ya que el sistema presenta poca dependencia entre las clases. Es en el modelo donde únicamente se encuentran algunas relaciones de asociación entre las clases, pero no representa una gran jerarquía.

Dentro de los patrones GOF se utilizaron:

- ✓ **Mediador:** la comunicación en la base de datos se puede tornar compleja debido a las dependencias marcadas entre las tablas que la componen, esto resulta engorroso a la hora de acceder a un determinado valor. La solución a este inconveniente viene dada por la utilización de este patrón, específicamente, se encuentra reflejado en la creación de una clase para la relación entre las clases (dat_persona) y (nom_limitante), en este caso dat_persona_limitante.

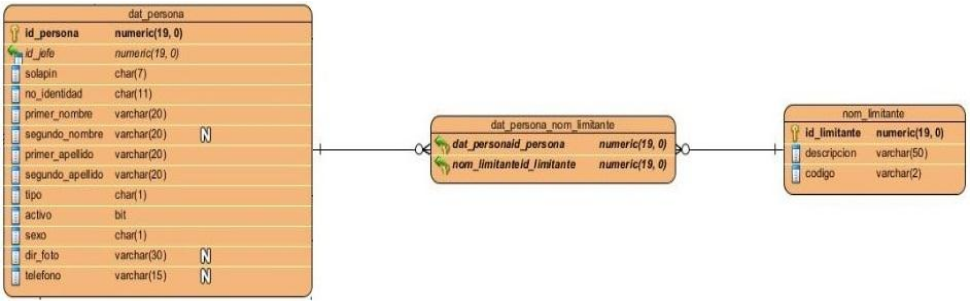

Figura 7. Aplicación del patrón Mediador

- ✓ **Cadena de Responsabilidad:** está concebido que ante la ocurrencia de un error al realizarse una determinada consulta a la base de datos, el mismo sea manejado por el modelo, creando una nueva

excepción de tipo ZendExt_Exception. Dicha excepción debe ser propagada al controlador, el cual será el encargado de capturarla y enviarla a la vista ya traducida, esta última por su parte mostrará un mensaje al usuario en un lenguaje entendible notificando el error y sin especificar detalles del mismo. De esta manera se distribuyen las responsabilidades entre los diferentes componentes, evidenciándose por lo tanto el empleo de este patrón.

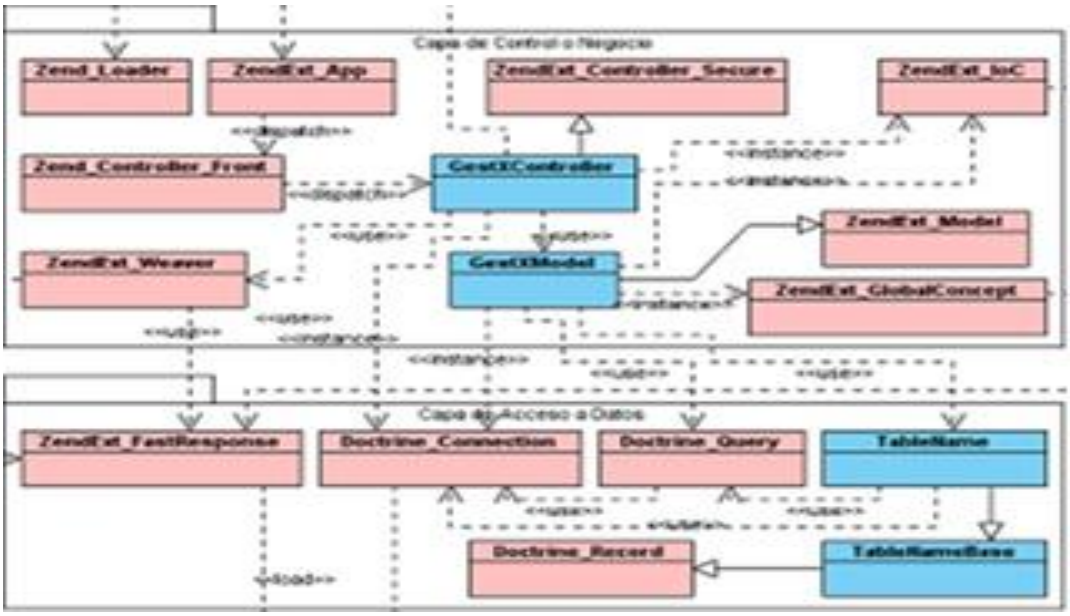

Figura 8. Aplicación del patrón Cadena de Responsabilidad

2.8. Modelo de datos del sistema.

Un modelo físico de los datos incluirá todos los artefactos de la base de datos requeridos para crear relaciones entre las tablas o para alcanzar metas del funcionamiento, tales como índices; ligando las tablas, donde las mismas son repartidas. El modelo físico de los datos se puede utilizar generalmente para calcular estimaciones del almacenaje y puede incluir los detalles específicos de la asignación de almacenaje para un sistema dado de la base de datos.(38)

A continuación se presenta el modelo de datos del sistema el cual cuenta con 14 tablas.

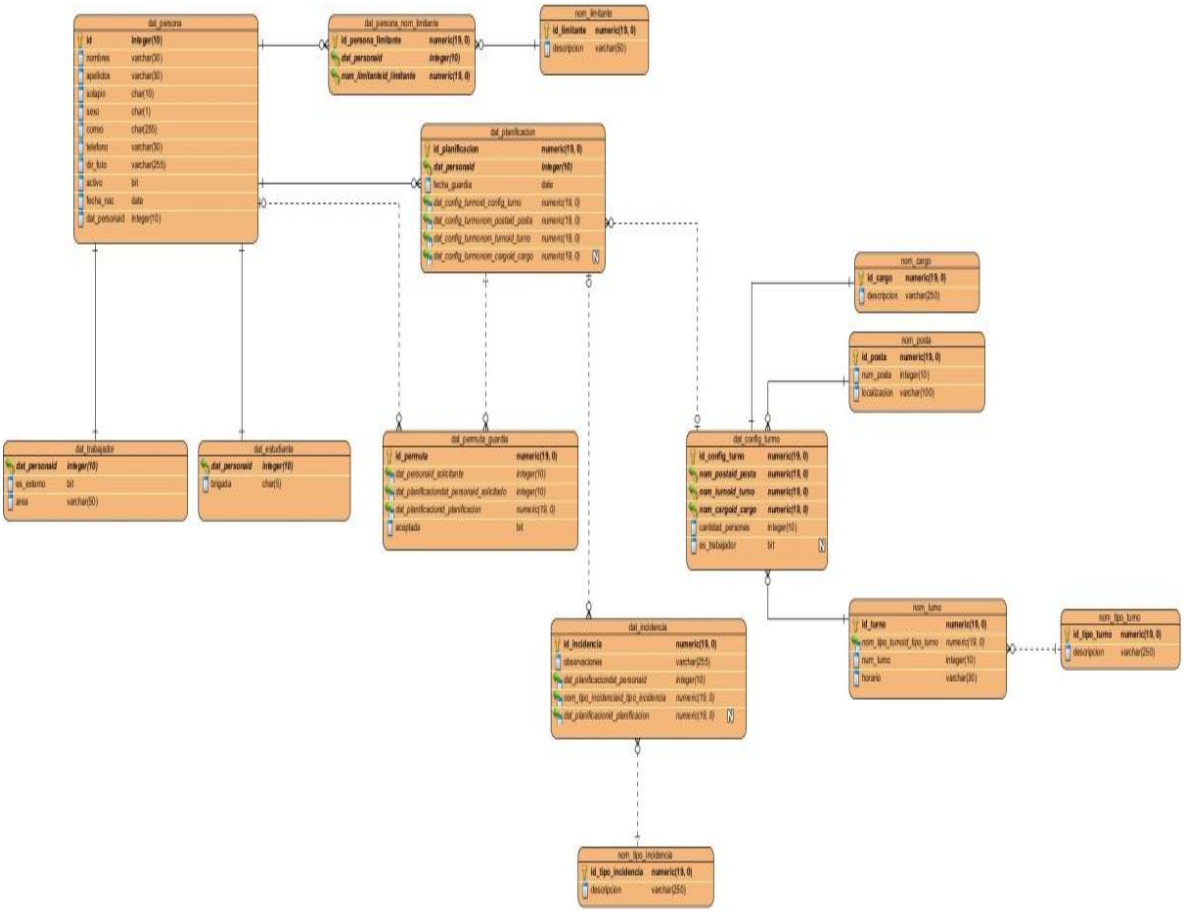

Figura 9. Modelo de datos del sistema.

2.8.1. Descripción de las tablas

A continuación se muestran las descripciones de algunas de las tablas del modelo de datos del sistema generadas, las restantes pueden verse en el Anexo 6.

Tabla 3. Descripción de la tabla "dat_persona".

Nombre: "dat_persona"		
Tipo: Entidad		
Descripción: Almacena los datos correspondientes de cada una de las personas.		
Atributo	**Tipo**	**Descripción**
id	serial	Identificador de la tabla.
solapin	char	Número de solapín de la persona.
nombres	varchar	Nombres de la persona.
apellidos	varchar	Apellidos de la persona.

fecha_nac	numeric	Fecha de nacimiento de la persona.
correo	varchar	Correo de la persona.
activo	boolean	Indica si la persona está activa o no para la guardia.
sexo	char	Sexo de la persona.
dir_foto	varchar	Dirección de la foto del solapín de la persona.
telefono	varchar	Teléfono de la persona.

Tabla 4. Descripción de la tabla "dat_persona_limitante".

Nombre: "dat_persona_limitante"		
Tipo: Entidad		
Descripción: Relación entre las tablas "dat_persona" y "nom_limitante".		
Atributo	**Tipo**	**Descripción**
id_persona	numeric	Identificador de la tabla dat_persona (Llave foránea).
id_limitante	numeric	Identificador de la tabla nom_limitante.
id_persona_limitante	serial	Identificador de la tabla.

Tabla 5. Descripción de la tabla "nom_limitante''.

Nombre: "nom_limitante"		
Tipo: Entidad		
Descripción: Almacena los datos de las limitantes por las que una persona no puede realizar la guardia.		
Atributo	**Tipo**	**Descripción**
id_limitante	numeric	Identificador de la tabla.
descripcion	varchar	Descripción de la limitante.

Tabla 6. Descripción de la tabla "dat_planificacion''.

Nombre: "dat_planificacion**"**		
Tipo: Entidad		
Descripción: Almacena los datos de la planificación de la guardia.		
Atributo	**Tipo**	**Descripción**
fecha_guardia	date	Fecha de la planificación de la guardia.
id_persona	numeric	Identificador de la tabla dat_persona (Llave foránea).
id_planificacion	serial	Identificador de la tabla.
id_config_turno	numeric	Identicador de la tabla dat_config_turno (Llave foránea).

Tabla 7. Descripción de la tabla "dat_permuta_guardia''.

Nombre: "dat_permuta_guardia"		
Tipo: Entidad		
Descripción: Almacena las solicitudes de permutas que se pueden realizar en la planificación de la guardia.		
Atributo	**Tipo**	**Descripción**
id_permuta	serial	Identificador de la tabla.
id_solicitante	numeric	Identificador de la tabla dat_planificacion (Llave foránea).
id_solicitado	numeric	Indicador de la tabla dat_planificacion (Llave foránea).
aceptada	boolean	Indica si es aceptada o no la permuta.

2.9. Diagrama de componentes

Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre los mismos. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.(39)

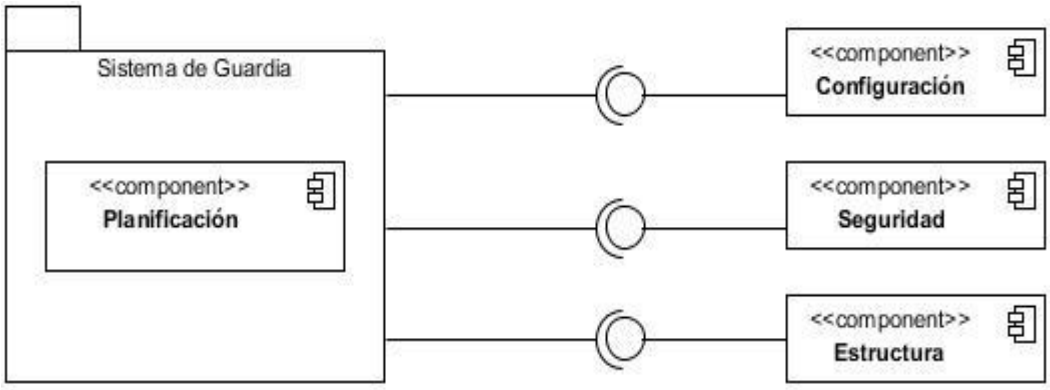

Figura 10. Diagrama de componentes

2.10. Diagrama de clases del diseño

El diagrama de clases es el diagrama principal para el análisis y diseño de un software. Estos diagramas estáticos además de mostrar las clases junto con sus métodos y atributos también sirven para visualizar las relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia, de uso y de contenido.(40) A continuación se muestra el diagrama de clases del diseño identificado para el desarrollo del sistema de gestión de la GOE, ver **figura 11**.

Diagrama de clases del diseño del proceso Planificación de la Guardia-Obrera-Estudiantil.

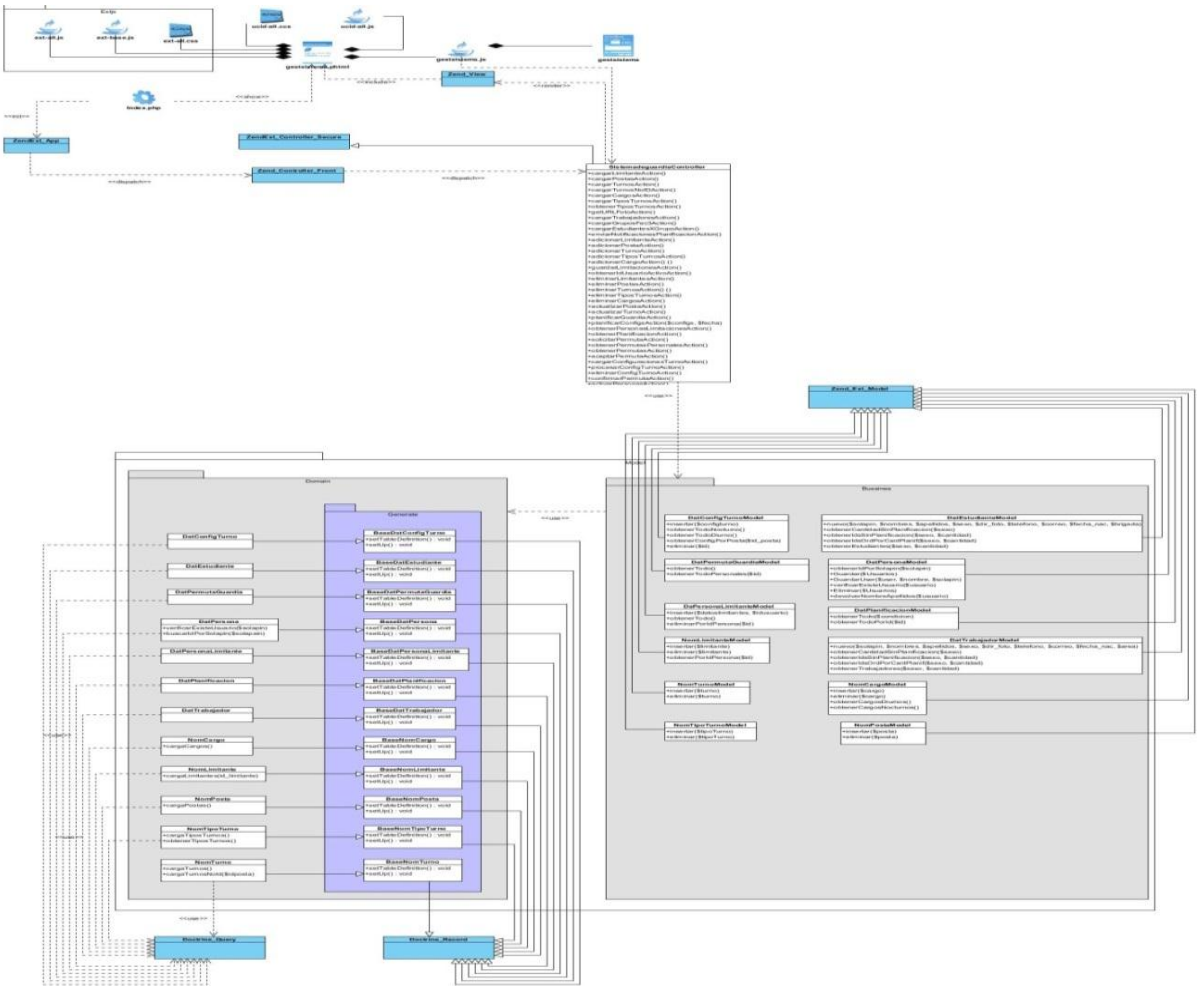

Figura 11. Diagrama de clases del diseño del proceso: Planificación de la Guardia-Obrera-Estudiantil.

Tabla 8. Descripción de las clases del diseño del proceso Planificación de la GOE.

Clases	Descripción
Extjs	Contiene los componentes generados a través de la librería Java Script Extjs.
getsistema.phtml	Responsable de visualizar a través de los js la información necesaria del proceso planificación de la Guardia Obrera-Estudiantil.
getsistema.js	Debe generar de forma dinámica a través del DOM y utilizando la librería Extjs componentes a través de los cuales se adicionen y se modifiquen la planificación de la guardia. Responsable de enviar y recibir los datos de la controladora utilizando tecnología AJAX.

ZendExt_Model	Obtiene las conexiones activas en el sistema cuando se trata de acceder a una clase determinada del modelo.
ZendExt_Controller_From	Inicializa el entorno de la solicitud, enruta la solicitud entrante, y luego envía las acciones descubiertas; le agrega las respuestas y los devuelve cuando el proceso se haya completado.
SistemaGuardiaController	Clase controladora que sirve de puente entre la vista y el modelo. Contiene los métodos necesarios para adicionar, modificar o eliminar las planificaciones.
ZendExt_Controller_Secure	Encargada de gestionar acciones personalizadas y está integrada a la seguridad.
Paquete models	Encargado de manejar los datos persistentes dentro del componente. Contiene el Bussines y el Domain.
ZendExt_Model	Modelo gestor de negocio que permite entre otras funcionalidades iniciar la conexión a la base de datos.

Además, se puede apreciar el diagrama de clases del diseño para el Control de la Guardia-Obrera-Estudiantil necesario en la realización del análisis en el Anexo 7.

2.11. Diagramas de secuencias

El diagrama de secuencia muestra una interacción, que representa la secuencia de mensajes entre las instancias de clases, componentes, subsistemas o actores. El tiempo fluye hacia abajo en el diagrama y muestra el flujo de control de un participante a otro.(41) Se realizan diagramas de secuencia para definir acciones que se pueden realizar en la aplicación en cuestión. Se cuenta con el diagrama de secuencia correspondiente a la funcionalidad Generar planificación de la Guardia-Obrera-Estudiantil, ver **figura 12**.

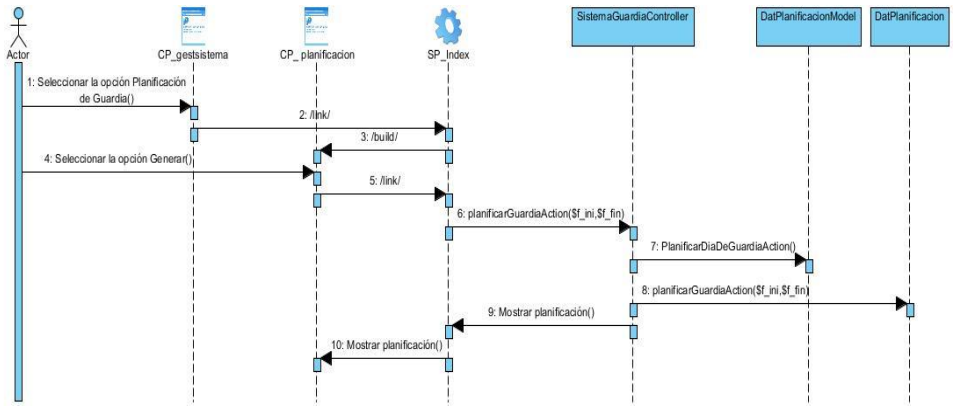

Figura 12. Diagrama de secuencia: Generar planificación de la Guardia-Obrera-Estudiantil

Una muestra de los diagramas de secuencia puede ser apreciada en el Anexo 8.

2.12 Prototipo de interfaz de usuario funcional

Los prototipos de interfaz de usuario son una representación parcial de la interfaz de usuario que tendrá el software, que muestra la forma en que el usuario interaccionará con él. Este prototipo es un elemento muy importante para la comunicación con el usuario en la definición de los requisitos, pues al revisar la interfaz, el usuario puede refinar sus necesidades y comunicarlas al desarrollador.(42)

A continuación se muestra el prototipo de interfaz de usuario de la ventana principal del sistema que se obtuvo como resultado del diseño del mismo. Ver **figura 13**. El resto de los prototipos de interfaz de usuario se encuentran en el Anexo 9.

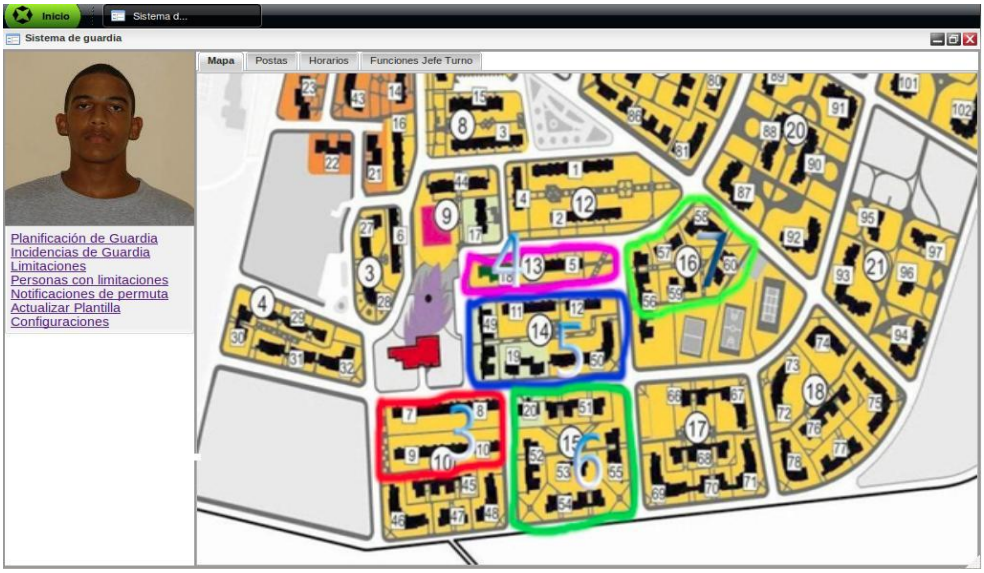

Figura 13. Prototipo de interfaz de usuario: interfaz principal.

2.13 Conclusiones del capítulo

Con la realización de este capítulo se concluye lo siguiente:

- La definición de la propuesta de sistema, la obtención del mapa de proceso de negocio y diagrama de procesos, posibilita la comprensión del funcionamiento de cada proceso que será informatizado.
- La obtención de los requisitos funcionales y no funcionales del sistema permite adquirir una adecuada comprensión de las necesidades existentes y obtener la información necesaria para un posterior desarrollo de la aplicación.
- La fundamentación de los patrones de diseño a utilizar permite que exista una baja dependencia y una alta reutilización entre las clases.

Por consiguiente, una vez precisadas las funcionalidades y la estructura de la solución es posible el comienzo de la implementación del sistema.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBA

3.1 Introducción

En este capítulo se realiza una breve descripción del marco de trabajo a utilizar, así como los estándares a utilizar durante la implementación del sistema. También se describen las clases y funcionalidades de dicho sistema, además de realizar el diagrama de despliegue para obtener una visión más clara sobre el mismo. Por último, se validará el diseño propuesto a través de las métricas TOC y RC, se describirán las pruebas de caja negra realizadas al software y los resultados obtenidos de estás.

3.2 Estructura del marco de trabajo Sauxe

A continuación se presenta la estructura del marco de trabajo Sauxe, mostrando cómo será organizada la implementación del sistema a desarrollar, de manera que facilite la organización y claridad durante el desarrollo. Además, se explica cómo va a ser garantizada la seguridad del Sistema para la gestión de la GOE en la Facultad 3.

Contenido dentro de la carpeta apps:

En la carpeta denominada apps se almacenan los controladores y el modelo de cada una de las funcionalidades a desarrollar.

Figura 14. Contenido de la carpeta de aplicación apps.

- ✓ **comun:** contiene la carpeta recursos y dentro de esta una denominada xml. Esta última incluye a los ficheros: ioc, validation, exception que serán explicados a continuación.
- ✓ **ioc:** es donde se publican los servicios que brindan cada uno de los componentes en cuanto a nombre de clases, funciones y tipo de resultado.
- ✓ **validation:** chequea las precondiciones antes de ejecutar una determinada función en el servidor según el tipo de parámetros, la acción y el usuario que la realice.
- ✓ **exception:** se define el tipo, idioma y la descripción del mensaje que va a ser mostrado cuando se lance una excepción en el servidor.

El componente **Sistema de Guardia** va a contener un conjunto de carpetas que serán especificadas a continuación:

Figura 15. Contenido del componente Sistema de Guardia dentro de la carpeta apps.

En la carpeta **controllers** se encontrarán las clases controladoras encargadas de gestionar las funcionalidades del sistema.

La carpeta **models** se estructura como se muestra en la **figura 16**:

Figura 16. Contenido de la carpeta models.

Esta carpeta contiene dos carpetas, las cuales a su vez agrupan clases y otras carpetas. Las mismas serán explicadas a continuación:

- ✓ **bussines:** contiene las clases necesarias para acceder a los datos que persisten en la base de datos.
- ✓ **domain:** contiene las clases generadas por el ORM Doctrine Generator a partir de cada una de las tablas existentes en la base de datos.

Cada una de estas clases mencionadas anteriormente heredará de una clase generada igualmente por el Doctrine Generator las cuales se ubican en otra carpeta dentro de esta llamada generated.

La carpeta **services** incluye todas las clases y funcionalidades de los servicios que va a ofrecer el sistema. Para solicitar un servicio que está en otro dominio, se accede a través de la carpeta services, quien analizará la solicitud e irá a la clase que tiene dicha funcionalidad y la devolverá a services, que a su vez se la entregará al dominio solicitante.

La carpeta **views** contiene las carpetas idioma y scripts, que se encargan de contener el idioma en que se va a mostrar la aplicación y las páginas clientes respectivamente.

Figura 17. Contenido de la carpeta views.

Contenido dentro de la carpeta web:

Al mismo nivel de la carpeta apps mencionada anteriormente debe existir además una carpeta que contiene las vistas de los subsistemas y componentes. Dicha carpeta se denomina web.

Figura 18. Carpeta de diseño correspondiente al componente Sistema de Guardia.

- ✓ **index:** este fichero incluye la dirección del archivo de configuración y a través de este inicializa la aplicación para que se carguen en la misma un conjunto de componentes necesarios para su funcionamiento. Su código permanece igual para todos los componentes.

La carpeta **views** contendrá los js que se explican a continuación.

Figura 19. Contenido dentro de la carpeta views.

- ✓ **js:** comprenderá las clases Java Script necesarias para que el usuario interactúe con el sistema y obtenga los resultados necesarios. Está compuesta por carpetas con los nombres de las funcionalidades del componente.

3.2.1 Seguridad del sistema

La seguridad del sistema va a ser garantizada a través del sistema de gestión Integral de Seguridad (ACAXIA), el cual tiene como objetivo garantizar la administración de la seguridad en sistemas de gestión. Este brinda sus servicios a todos los sistemas que se suscriban a él. Para ello se gestionan las conexiones a la base de datos, las funcionalidades asociadas y las acciones que realizan las mismas. Una vez registrada esta información se procede a la creación de roles a los cuales se les dan los permisos dentro de cada sistema. Luego se crean los usuarios con el perfil seleccionado y se le asignan uno o muchos roles en una o muchas entidades respectivamente.

3.3 Estándares de código

Un estándar de código se basa en la estructura y apariencia física de un programa, con el fin de facilitar la lectura, comprensión, mantenimiento del código y reutilización a lo largo del proceso de desarrollo de un software y no en la lógica del programa. Este no solo busca definir la nomenclatura de las variables, objetos, métodos y funciones, sino que también tiene que ver con el orden y la legibilidad del código, aspecto crucial a la hora de darle mantenimiento y mejorar las funcionalidades de un software.(43) Partiendo de lo dicho anteriormente, se definen 3 partes principales dentro de un estándar de programación:

- ✓ Convención de nomenclatura: es la forma de nombrar las variables, funciones, métodos.
- ✓ Convenciones de legibilidad de código: es la forma de organizar el código.
- ✓ Convenciones de documentación: es la manera de establecer los comentarios, la ayuda.

3.3.1 Nomenclatura de las clases

Los nombres de las clases comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación **PascalCasing**, la cual define que los identificadores y nombres de variables, métodos y funciones están compuestos por múltiples palabras juntas, iniciando cada palabra con letra mayúscula y con solo leerlo se reconoce el propósito de la misma.(44)

Ejemplo: DatPlanificacionModel. En este caso el nombre de clase está compuesto por 3 palabras iniciadas cada una con letra mayúscula.

Nomenclatura según el tipo de clases

- ✓ **Clases controladoras:** las clases controladoras después del nombre llevan la palabra:”Controller”. Ejemplo: SistemadeguardiaController
- ✓ **Clases de modelo de Negocio:** las clases que se encuentran dentro de Business después del nombre llevan la palabra: “Model”. Ejemplo: DatPlanificacionModel.
- ✓ **Domain (Dominio):** las clases que se encuentran dentro de Domain, el nombre que reciben es el de la tabla en la base de datos. Ejemplo: DatPlanificacion.
- ✓ **Generated (Dominio base):** las clases que se encuentran dentro de Generated, el nombre que reciben comienza con la palabra: “Base” y seguido el nombre de la tabla en la base de datos. Ejemplo: BaseDatPlanificacion.

3.3.2 Nomenclatura de las funcionalidades y atributos

El nombre a emplear para las funcionalidades y los atributos se escribe con la inicial del identificador en minúscula, en caso de que sea un nombre compuesto se empleará la notación **CamelCasing** que es similar a la antes mencionada **PascalCasing,** con la excepción de la primera letra.

- ✓ Ejemplo de un método: personasLimitaciones. El nombre del método está compuesto por 2 palabras, la primera en minúsculas y la segunda iniciando con letra mayúscula.
- ✓ Las principales funcionalidades de las clases controladoras se les escribe el nombre y seguida la palabra “Action” Ejemplo: generarPlanificacionAction ().
- ✓ Ejemplo de un atributo: cantidadPersonas. El nombre del atributo está compuesto por dos palabras, la primera en minúsculas y la segunda iniciando con letra mayúscula.

3.3.3. Nomenclatura de los comentarios

Los comentarios deben ser claros y precisos de forma tal que se entienda el propósito de lo que se está desarrollando. En la realización de un software se debe tener comentarios de todo lo que se haga dentro del desarrollo, para lograr un código más legible y reutilizable y así se pueda aumentar su mantenibilidad a lo largo del tiempo.

3.4 Descripción de las clases y funcionalidades del sistema

Clase Controladora:

Tabla 9. Descripción de la clase sistemadeGuardiaController.

Nombre: sistemadeGuardiaController	
Tipo de clase: Controladora	
Para cada funcionalidad:	
Nombre	cargarLimitanteAction()
Descripción	Permite cargar todas las limitaciones de la Base de Datos.
Nombre	cargarPostasAction()
Descripción	Permite cargar todas las postas de la Base de Datos.
Nombre	cargarTurnosAction()
Descripción	Permite cargar todos los turnos de la Base de Datos.
Nombre	cargarTurnosNoIDAction()

Descripción	Permite cargar todos los turnos por los Ids de la base de datos.
Nombre	cargarCargosAction()
Descripción	Permite cargar todos los cargos por los Ids de la base de datos.
Nombre	cargarIncidenciasAction()
Descripción	Permite cargar todos los tipos de incidencias de la base de datos.
Nombre	cargarTiposTurnosAction()
Descripción	Permite cargar todos los tipos de turnos de la base de datos.
Nombre	obtenerTiposTurnosAction()
Descripción	Devuelve todos los tipos de turnos de la planificación.
Nombre	cargarTrabajadoresAction()
Descripción	Permite cargar todos los trabajadores de la Facultad 3.
Nombre	activarPersonasAction()
Descripción	Permite activar una persona para la planificación de la guardia.
Nombre	desactivarPersonasAction()
Descripción	Permite desactivar una persona para la planificación de la guardia.

Clase del Modelo:

Tabla 10. Descripción de la clase DatPlanificacionModel.

Nombre: DatPlanificacionModel	
Tipo de clase: Modelo	
Para cada funcionalidad:	
Nombre	obtenerTodo($condicion)
Descripción	Devuelve todas las planificaciones realizadas.
Nombre	obtenerTodoPorId($id)
Descripción	Devuelve todas las planificaciones realizadas mediante el Ids.
Nombre	eliminar()

Descripción	Permite eliminar toda la planificación de guardia.

Clase del Dominio:

Tabla 11. Descripción de la clase NomTipoTurno.

Nombre: NomTipoTurno	
Tipo de clase: Dominio	
Para cada funcionalidad:	
Nombre	cargaTiposTurnos()
Descripción	Permite cargar todos los tipos de turnos existentes en la Base de Datos.
Nombre	obtenerTiposTurnos()
Descripción	Devuelve todos los tipos de turnos de la planificación.

3.5 Diagrama de despliegue

El diagrama de despliegue muestra la configuración de los nodos de procesamiento en tiempo de ejecución, los vínculos de comunicación entre ellos, y las instancias de los componentes y objetos que residen en ellos. Está compuesto por nodos, dispositivos y conectores. El propósito del modelo de despliegue es capturar la configuración de los elementos de procesamiento y las conexiones entre estos elementos en el sistema. Permite el mapeo de procesos dentro de los nodos, asegurando la distribución del comportamiento a través de aquellos nodos que son representados.(45) Se definirá una arquitectura cliente-servidor detallada como se muestra en la **figura 20.**

A continuación se muestra el diagrama de despliegue del sistema:

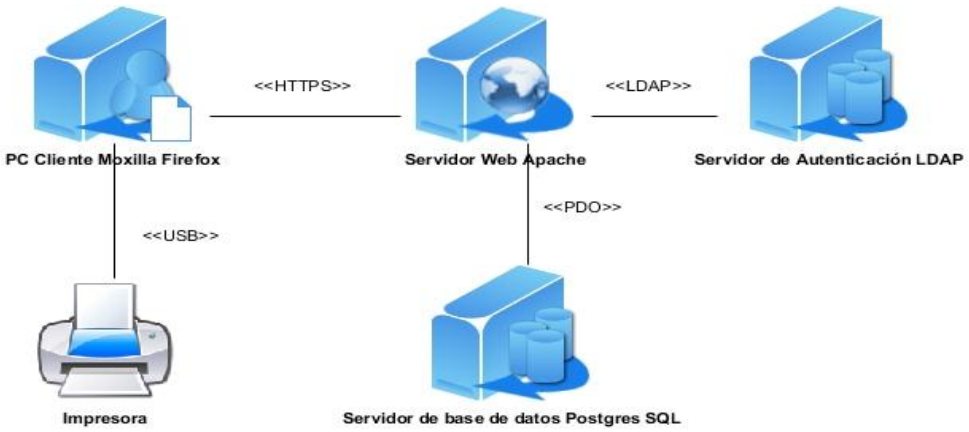

Figura 20. Diagrama de despliegue del sistema.

PC Cliente: computadora en la cual la aplicación se ejecutará a través de un navegador, en este caso se debe usar el mozilla firefox.
Servidor Web: radica la lógica de negocio de la aplicación. Servidor web Apache 2.2.4, utilizando el lenguaje de programación del lado del servidor PHP 5.
Servidor de Autenticación: servidor a través del cual se realiza la autenticación de los usuarios del sistema pertenecientes al dominio uci.cu, en aras de lograr que no accedan al sistema usuarios no autorizados.
Servidor de base de datos: servidor de datos PostgreSQL 9.1, donde se encuentra la base de datos que utiliza el sistema, este puede estar instalado en la misma computadora donde se encuentra el servidor web.
Impresora: utilizada para imprimir los reportes del sistema en caso de ser necesario.

3.6. Validación del diseño propuesto

Las métricas van a ayudar a la evaluación de los modelos de análisis y de diseño, en donde proporcionarán una indicación de la complejidad de diseños procedimentales y de código fuente y ayudarán en el diseño de pruebas más efectivas.(46)

Para la evaluación de la calidad del diseño propuesto para el sistema se hizo un estudio de las métricas básicas inspiradas en la calidad del diseño orientado a objeto, en el mismo se abarcan atributos de calidad que permiten medir la calidad del diseño propuesto. Dentro de estos se encuentran:(46)

- ✓ **Responsabilidad:** consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- ✓ **Complejidad de implementación:** consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
- ✓ **Reutilización:** consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- ✓ **Acoplamiento:** consiste en el grado de dependencia o interconexión de una clase o estructura de clase con otras, está muy ligada a la característica de reutilización.
- ✓ **Complejidad del mantenimiento:** consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costos y la planificación del proyecto.

- ✓ **Cantidad de pruebas:** consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad del producto diseñado.

Las métricas concebidas como instrumento para evaluar la calidad del diseño y su relación con los atributos de calidad definidos son las siguientes:

- **Tamaño Operacional de Clase (TOC):** se refiere al número de métodos pertenecientes a una clase. Está determinada por los atributos: Responsabilidad, Complejidad de implementación y la Reutilización, existiendo una relación directa con los dos primeros e inversa con el último antes mencionado.

Tabla 12. Tamaño operacional de clase (TOC).

Atributo que afecta	Modo en que lo afecta
Responsabilidad	Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
Complejidad de implementación	Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
Reutilización	Un aumento del TOC implica una disminución en el grado de reutilización de la clase.

Tabla 13. Rango de valores para la evaluación técnica de los atributos de calidad (Responsabilidad, Complejidad de Implementación y Reutilización) relacionados con la métrica TOC.

	Categoría	Criterio
Responsabilidad	Baja	< =Prom. (5.27)
	Media	Entre Prom. Y 2* Prom
	Alta	> 2* Prom
Complejidad de implementación	Baja	< =Prom
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom
Reutilización	Baja	> 2*Prom
	Media	Entre Prom. y 2* Prom
	Alta	<= Prom

Tabla 14. Umbrales para el TOC.

Tamaño operacional de la clase	Criterio
Pequeña	<= Prom.(5.27)
Media	Entre Prom. y 2 * Prom.
Grande	> 2* Prom.

Resultados del instrumento de evaluación de la métrica Tamaño Operacional de Clase (TOC)

Representación en % de los resultados obtenidos en el instrumento, agrupados en los intervalos definidos, ver **Figura 21.**

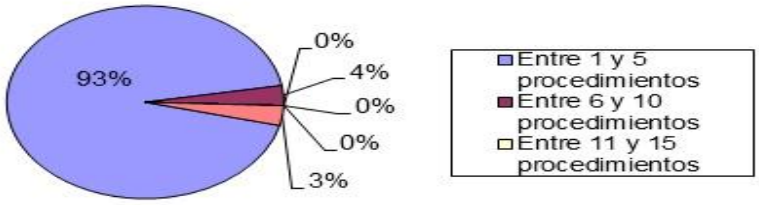

Figura 21. Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos.

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad, ver **Figura 22.**

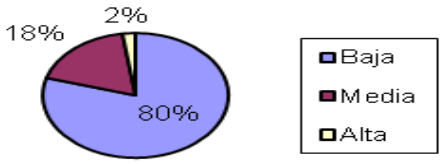

Figura 22. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Responsabilidad.

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Complejidad de implementación, ver **Figura 23**:

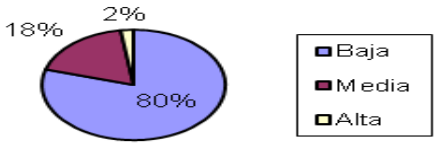

Figura 23. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Complejidad de implementación.

Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Reutilización, ver **Figura 24:**

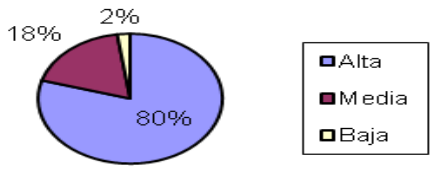

Figura 24. Representación de la incidencia de los resultados de la evaluación de la métrica TOC en el atributo Reutilización.

Al analizar los resultados obtenidos luego de aplicar el instrumento de medición de la métrica TOC, se puede concluir que el diseño propuesto para el sistema es simple y tiene una calidad aceptable, teniendo en cuenta que la mayoría de las clases (93%) se distribuyen las operaciones. Los atributos de calidad se encuentran en un nivel satisfactorio en el 80% de las clases, de manera que se puede observar cómo se fomenta la Reutilización (elemento clave en el proceso de desarrollo de software) y cómo están reducidas en menor grado la Responsabilidad y la Complejidad de implementación.

- **Relaciones entre Clases (RC):** dado por el número de relaciones de uso de una clase. Está determinada por los atributos: Acoplamiento, Complejidad de mantenimiento, Reutilización y Cantidad de pruebas, existiendo una relación directa con los tres primeros e inversa con el último antes mencionado.

Tabla 15. Relaciones entre clases (RC).

Atributo que afecta	Modo en que lo afecta
Acoplamiento	Un aumento del RC implica un aumento del Acoplamiento de la clase.
Complejidad de mantenimiento	Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
Reutilización	Un aumento del RC implica una disminución en el grado de reutilización de la clase.
Cantidad de pruebas	Un aumento del RC implica un aumento de la cantidad de pruebas de unidad necesarias para probar una clase.

Tabla 16. Rango de valores para la evaluación técnica de los atributos de calidad (Acoplamiento, Complejidad de mantenimiento, Reutilización y Cantidad de Pruebas) relacionados con la métrica RC.

	Categoría	Criterio
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	2
	Alto	>2
Complejidad de mantenimiento	Baja	< =Prom (1.31)
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom
Reutilización	Baja	> 2*Prom
	Media	Entre Prom. y 2* Prom
	Alta	<= Prom
Cantidad de Pruebas	Baja	< =Prom
	Media	Entre Prom. y 2* Prom
	Alta	> 2* Prom

Resultados del instrumento de evaluación de la métrica Relaciones entre Clases (RC).

Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos, ver **Figura 25:**

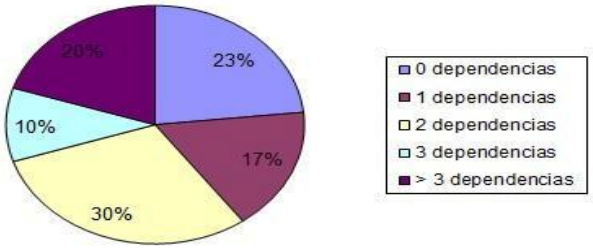

Figura 25. Representación en % de los resultados obtenidos en el instrumento agrupados en los intervalos definidos.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Acoplamiento, ver **Figura 26:**

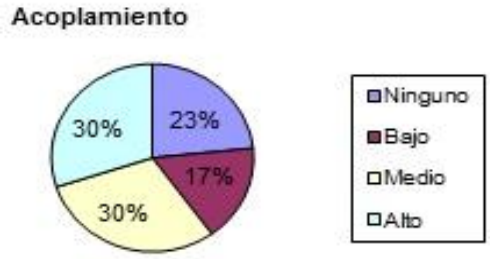

Figura 26. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Acoplamiento.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Complejidad de mantenimiento, ver **Figura 27:**

Figura 27. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Complejidad de mantenimiento.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Cantidad de pruebas, ver **Figura 28:**

Figura 28. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Cantidad de pruebas.

Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Reutilización, ver **Figura 29:**

Figura 29. Representación de la incidencia de los resultados de la evaluación de la métrica RC en el atributo Reutilización.

Al analizar los resultados obtenidos luego de aplicar el instrumento de medición de la métrica RC, se puede concluir que el diseño propuesto para el sistema es simple y tiene una calidad aceptable, teniendo en cuenta que la mayoría de las clases (70%) poseen 2 ó menos dependencias respecto a otras. Los atributos de calidad se encuentran en un nivel satisfactorio, en el 40% de las clases el grado de acoplamiento es mínimo, la Complejidad de mantenimiento, la Cantidad de pruebas y la Reutilización se comportan favorablemente para un 70% de las clases.

3.7. Pruebas de software al sistema

La prueba de software involucra las operaciones del sistema bajo condiciones controladas y evaluando los resultados. Las condiciones controladas pueden ser normales o anormales. La prueba puede intencionalmente esforzar al programa y producir errores en las respuestas para determinar si los sucesos ocurren cuando no tendrían que ocurrir o cuando los hechos no suceden cuando deberían suceder. Además son una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad de un programa u ordenador.(47)

La prueba de software es un elemento crítico para la garantía del correcto funcionamiento del software. Entre sus objetivos están:

- ✓ Detectar defectos en el software.
- ✓ Verificar la integración adecuada de los componentes.
- ✓ Verificar que todos los requisitos se han implementado correctamente.
- ✓ Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.
- ✓ Diseñar casos de prueba que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.(48)

3.7.1. Niveles de prueba

A la hora de evaluar dinámicamente un sistema se debe comenzar por los componentes más simples y pequeños e ir avanzando progresivamente hasta probar todo el software en su conjunto. Las pruebas se aplican en distintos niveles de trabajo, dentro de estos se distinguen:

1. Pruebas de unidad: prueba individual a las unidades separadas de un sistema de software.

2. Pruebas de integración: los componentes individuales son combinados con otros componentes para asegurar que la comunicación, enlaces y los datos compartidos ocurran apropiadamente.

3. Pruebas del sistema: son usualmente conducidas para asegurar que todos los módulos trabajan como sistema sin error. Es similar a la prueba de integración pero con un alcance mucho más amplio.

4. Pruebas de aceptación: son realizadas principalmente por los usuarios con el apoyo del equipo del proyecto. El propósito es confirmar que el sistema está terminado, que desarrolla puntualmente las necesidades de la organización y que es aceptado por los usuarios finales.(49)

3.7.2. Métodos de prueba

Un método de prueba es un enfoque sistemático, independiente del nivel en que se enmarque la prueba, que ayuda a encontrar buenos conjuntos de casos de prueba para detectar diferentes tipos de errores.

1. Pruebas de caja blanca: Se comprueban los componentes internos. Comprueba los caminos lógicos del software proponiendo casos de prueba que ejerciten conjuntos específicos de condiciones. Se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado o mencionado.

2. Pruebas de caja negra: Se comprueban las funcionalidades sin tener en cuenta la estructura interna. Se refiere a las pruebas que se llevan a cabo sobre la interfaz del software. O sea, a través de los casos de prueba se demuestra que las funciones del software son operativas, que la entrada se acepta de

forma adecuada y que se produce un resultado correcto, así como que la integridad de la información externa se mantiene.

Este tipo de pruebas permite encontrar:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a las bases de datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y terminación.

Dentro de la prueba de caja negra se incluyen varias técnicas de pruebas tales como:

Partición de equivalencia: divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.

Análisis de valores límites: prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.

Grafos de causa-efecto: permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.(21)

3.7.3. Pruebas realizadas al sistema

Para verificar el correcto funcionamiento del sistema se realizaron pruebas funcionales aplicando el método de caja negra y específicamente la técnica de partición de equivalencia. Esta última divide el dominio de entrada de un programa en un número finito de variables de equivalencia. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa. Se definen dos tipos de variables de equivalencia: las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real de dato.

Para cada uno de los requisitos funcionales fueron diseñados los casos de prueba, de los cuales se muestra uno a continuación:

Caso de prueba para el requisito Generar guardia.

Condiciones de ejecución

- ✓ Se debe identificar y autenticar ante el sistema y además debe tener los permisos para ejecutar esta acción.

- ✓ Se debe seleccionar el subsistema Sistema de Guardia/sistema de guardia.

Tabla 17. Caso de prueba para el requisito: Generar guardia.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Generar Guardia.	El sistema genera la planificación de la guardia de los estudiantes y trabajadores.	EP 1.1: Generar la planificación de la guardia satisfactoriamente presionando el botón **Generar.**	– Se selecciona la opción **Personas con Limitaciones.** – Se muestra la interfaz **Personas con Limitaciones.** – Se muestra las personas con sus limitaciones. – Selecciona el botón **Activar** y **Desactivar** en dependencia de las limitaciones insertadas por la persona. – Muestra las personas activas y desactivadas para la guardia. – Se selecciona la opción **Planificación de Guardia.** – Se muestra la interfaz **Planificación de Guardia**. – Se selecciona la opción **Generar.** – Se muestra la ventana **Generar guardia.** – Se selecciona el periodo en que desea planificar la guardia. – Se selecciona la opción **Generar.** – Se muestra el mensaje de información. Se presiona el botón **Aceptar** del mensaje. – Se muestra la planificación de

			la guardia en el período seleccionado. – Notifica a los implicados de la planificación de la guardia.
		EP 1.2: Generar la planificación de la guardia dejando campos requeridos en blanco presionando el botón **Generar.**	– Se selecciona la opción **Personas con Limitaciones.** – Se muestra la interfaz **Personas con Limitaciones.** – Se muestra las personas con sus limitaciones. – Selecciona el botón **Activar** y **Desactivar** en dependencia de las limitaciones insertadas por la persona. – Muestra las personas activas y desactivadas para la guardia. – Se selecciona la opción **Planificación de Guardia.** – Se muestra la interfaz **Planificación de Guardia**. – Se selecciona la opción **Generar.** – Se muestra la ventana **Generar guardia.** – Se selecciona el periodo en que desea planificar la guardia. – Se selecciona la opción **Generar.** – Se muestra el mensaje de información. Se presiona el botón **Aceptar** del mensaje.

El sistema fue sometido a tres iteraciones de pruebas de caja negra. En la primera iteración se detectaron 9 no conformidades, de las cuales 7 fueron de aplicación y 2 de documentación. Dentro de las no

conformidades referentes a las de aplicación se encuentran 6 de validación y 1 de funcionalidad. En cuanto a las de documentación se encontraron 1 no conformidad de redacción y 1 de correspondencia. En la segunda iteración se detectaron 1 no conformidad, la cual fue de aplicación. Dentro de dicha no conformidad de aplicación se encontró 1 de funcionalidad. Todas estas no conformidades fueron resueltas seguidamente de haberlas detectado, lo que permitió que el sistema pasara a una tercera iteración en la que no se detectaron no conformidades, obteniendo resultados satisfactorios.

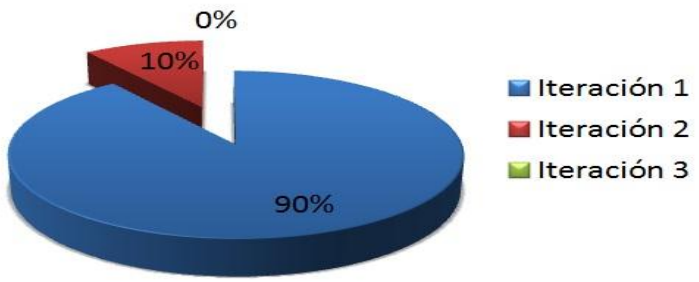

Figura 30. No conformidades por iteración detectadas durante las pruebas funcionales.

La aplicación desarrollada fue presentada ante la Vicedecana de Administración y Servicios, la cual luego de haberla probado estuvo satisfecha con el producto entregado. Prueba de esto lo constituye el acta de aceptación emitida por esta, en la cual consta que el sistema realizado cumple con las condiciones de calidad necesarias y satisface las necesidades plasmadas por la Vicedecana de Administración y Servicios, además contribuye a mejorar la gestión de la planificación de la Guardia-Obrera-Estudiantil en la Facultad 3. Para más información acerca del acta de aceptación emitida, ver Anexo 10.

3.8. Conclusiones del capítulo

Una vez finalizado el capítulo se concluye que:

- Los resultados arrojados por las métricas TOC y RC permiten valorar principalmente la complejidad en la implementación de la solución que se presenta con la actual investigación.
- La realización de las pruebas de caja negra al sistema para verificar su adecuado funcionamiento, aplicándose pruebas de partición de equivalencia, permite validar el cumplimiento de los requisitos capturados en la segunda disciplina de la fase de desarrollo.
- Las pruebas de aceptación realizadas hasta el momento por la Vicedecana de Administración y Servicios han permitido verificar y validar los requisitos identificados, arrojando que la solución propuesta proporciona un avance en la ejecución de los procesos de planificación y control de la guardia.

CONCLUSIONES GENERALES

Una vez terminado el presente trabajo de diploma se puede concluir que se desarrollaron todas las tareas a fin de cumplir los objetivos propuestos, resaltando que:

- ✓ El análisis de sistemas informáticos, tanto nacionales como internacionales, vinculados a la gestión de la planificación evidencian la no existencia de una solución informática capaz de cubrir todas las funcionalidades requeridas para realizar una adecuada gestión de la planificación de la guardia, lo cual hizo necesario el desarrollo de un sistema informático que permita realizar una adecuada planificación y control.
- ✓ El uso de las técnicas para la captura de los requisitos permitió identificar los requisitos funcionales del sistema, que de conjunto a la aplicación de las etapas de la Ingeniería de Requisitos, desarrollaron y administraron los mismos durante la investigación.
- ✓ El análisis y el diseño del sistema de gestión para la planificación de la guardia de la Facultad 3, permitió recoger todas las necesidades del cliente, además de lograr un mejor entendimiento de los procesos que fueron informatizados, garantizando una comprensión exacta sobre las funcionalidades a informatizar.
- ✓ Las pruebas de caja negra permitieron encontrar y corregir los errores no detectados durante la implementación posibilitando cumplir con las especificaciones requeridas y la validación del sistema implementado, las mismas arrojaron resultados satisfactorios demostrando la calidad del sistema desarrollado.

RECOMENDACIONES

Teniendo como base los resultados de este trabajo y la experiencia adquirida durante el desarrollo del mismo, se recomienda:

- ✓ Realizar mejoras al consumir servicios del LDAP, de forma tal que la base de datos del sistema siempre esté actualizada con las personas que van a estar involucradas en la guardia.
- ✓ Agregar funcionalidades al sistema que permitan realizar la planificación de la guardia del personal externo en los períodos vacacional y de fin de año.
- ✓ Realizar un estudio a nivel de universidad para incorporarle nuevas funcionalidades al sistema de forma tal que pueda ser usado en todas las áreas de la misma.

BIBLIOGRAFÍA

1. SUSANA BEATRIZ LLANUSA RUIZ, NEREIDA ROJO PÉREZ, MAGALI CARABALLOSO HERNÁNDEZ,ROBERTO CAPOTE MIR Y JULIA PÉREZ PIÑERO. TIC. In: [online]. [Accessed 5 junio 2013]. Available from: http://bvs.sld.cu/revistas/spu/vol31_3_05/spu08305.htm.

2. Impacto TIC. In: [online]. [Accessed 5 junio 2013]. Available from: http://www.utvm.edu.mx/OrganoInformativo/OrgFeb07/paginas/impacto.htm.

3. DAILÉ ALVAREZ HERNÁNDEZ Y ELIZETH MARTÍNEZ VAZQUEZ. *Sistema de Gestión para la Planificación Docente en la Facultad 5.* [online]. S.l.: Universidad de las Ciencias Informáticas, 2009. [Accessed 15 noviembre 2012].

4. Concepto de gestión — Definicion.de. In: *Definición.de* [online]. [Accessed 15 noviembre 2012]. Available from: http://definicion.de/gestion/.

5. ¿Que es un SISTEMA de GESTION? | Blog de MejoraTuGestion. In: *¿Que es un SISTEMA de GESTION?* [online]. [Accessed 15 noviembre 2012]. Available from: http://mejoratugestion.com/mejora-tu-gestion/que-es-un-sistema-de-gestion/.

6. VisualTime Gestión y Control Horario de personal. In: *VisualTime* [online]. [Accessed 9 abril 2013]. Available from: http://www.robotics.es/control-horario-personal/.

7. *TecnoHospital* [online]. S.l.: s.n. [Accessed 11 abril 2013]. Available from: http://www.coprava.com/prensa/Nota%20de%20prensa%20Gestion%20Turnos%20Abril08.pdf.

8. OPTIHPER. Asignación Optimizada de Personal y Tareas. In: [online]. [Accessed 18 mayo 2013]. Available from: http://users.dsic.upv.es/grupos/gps/optihper/aplicacion.html.

9. KAREL BEJERANO GONZÁLEZ Y FÉLIX MIGUEL MONTES DE OCA BARRIOS. *Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7*. S.l.: Universidad de las Ciencias Informáticas, 2009.

10. GRETTELL TORRES AGUILERA Y HUMBERTO ALMEIDA OQUENDO. *Gestor Web para el control de la Guardia Obrera de la Universidad de las Ciencias Informáticas (UCI).* S.l.: Universidad de las Ciencias Informáticas, 2009.

11. THAIS FIGUERAS GARCIA Y MIGUEL LA LLAVE IGLESIAS. *Solución informática para la Compartimentación de la información en los sistemas que utilizan el Sistema de Gestión Integral de Seguridad ACAXIA.* S.l.: Universidad de las Ciencias Informáticas, 2012.

12. SUBDIRECCIÓN DE PRODUCCIÓN. *Modelo de Desarrollo de Software v1.1* [online]. S.l.: Centro de Informatización de Gestión de Entidades. Available from: CEIGE-Modelo de Desarrollo de Software v1.1.

13. Sencha Ext JS - EcuRed. In: *Sencha Ext JS* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.ecured.cu/index.php/Sencha_Ext_JS.

14. Qué es Doctrine ORM? In: *Qué es Doctrine ORM?* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.tecnoretales.com/programacion/que-es-doctrine-orm/.

15. Zend Framework » ¿Qué es Zend Framework? In: *Zend Framework » ¿Qué es Zend Framework?* [online]. 2009. [Accessed 15 noviembre 2012]. Available from: http://spanish.zendfw.com/que-es-zend-framework/.

16. Zend Framework. In: *Zend Framework* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.oxagile.com/article/212-zend-framework-php-web-application-development.

17. ING. OINER GÓMEZ BARYOLO, ING. YOANDRY MOREJÓN BORBÓN, ING. DARIEN GARCÍA TEJO. *Trabajo Arquitectura Tecnológica para el Desarrollo de Software (Sauxe).* [online]. S.l.: s.n. [Accessed 21 enero 2013]. Available from: http://www.google.com.cu/url?sa=t&rct=j&q=sauxe&source=web&cd=1&cad=rja&ved=0CCwQFjAA&url=http%3A%2F%2Fwww.gestec.disaic.cu%2Fponencias%25202010%2FCUBA%2F(55)Oiner%2520GomezTRABAJO%2520ARQUITECTURA%2520TECNOL%25C3%2593GICA%2520PARA%2520EL%2520DESARROLLO%2520DE%2520SOFTWARE.doc&ei=Fpr9UKz4BJP08AS9-oC4CQ&usg=AFQjCNEKNZatiLOFoWFZwjG2sRGYhZEYAA&bvm=bv.41248874,d.eWU.

18. LENGUAJES DE PROGRAMACION. In: *El Mundo Informático* [online]. [Accessed 15 noviembre 2012]. Available from: http://jorgesaavedra.wordpress.com/2007/05/05/lenguajes-de-programacion/.

19. MIGUEL ANGEL ALVAREZ. Qué es PHP. In: *DesarrolloWeb.com* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.desarrolloweb.com/articulos/392.php.

20. ¿Qué es JavaScript? - Definición de Javascript. In: *¿Qué es JavaScript?* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.masadelante.com/faqs/javascript.

21. ARIADNA RENDÓN ARTOLA Y MANUEL ALEJANDRO CASTELLANOS PÉREZ. *Modelación y construcción de los componentes Gestor de actividades y Calendario para el Subsistema Planificación por Objetivos del Sistema Integral de Gestión de Entidades Cedrux.* Ciudad de la Habana: Universidad de las Ciencias Informáticas, 2010.

22. ¿Qué es PostgreSQL? In: *Microbuffer* [online]. [Accessed 15 noviembre 2012]. Available from: http://microbuffer.wordpress.com/2011/05/04/que-es-postgresql/.

23. ¿Qué es Firefox? In: *¿Qué es Firefox?* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.misrespuestas.com/que-es-firefox.html.

24. Características Mozilla Firefox. In: *Características Mozilla Firefox* [online]. [Accessed 15 noviembre 2012]. Available from: http://recursos.fundacionesplai.org/intranet/dinamizadores/linux/firefox/index.htm.

25. Apache 2.2. In: *Apache 2.2* [online]. [Accessed 15 noviembre 2012]. Available from: http://recursostic.educacion.es/observatorio/web/es/software/servidores/580-elvira-mifsud.

26. CARACTERÍSTICAS DE APACHE for Tesis Ing. Sistemas. In: *CARACTERÍSTICAS DE APACHE* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.scribd.com/doc/69446267/Tesis-Ing-Sistemas.

27. Herramienta Case Visual Paradigm. In: [online]. [Accessed 17 enero 2013]. Available from: http://dianbeel.blogspot.com/2012/06/segundo-trabajo-herramienta-case-visual.html.

28. IDE de Desarrollo - EcuRed. In: *IDE de Desarrollo* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n.

29. ¿Qué es NetBeans? In: *¿Qué es NetBeans?* [online]. [Accessed 15 noviembre 2012]. Available from: http://ayuda-java.blogspot.com/2007/07/qu-es-netbeans.html.

30. Caracteristicas NetBeans 6.9. In: *Caracteristicas NetBeans 6.9* [online]. [Accessed 15 noviembre 2012]. Available from: http://www.jujuyjug.com.ar/portal/index.php?option=com_content&view=article&id=74:netbeans-69-final&catid=3:newsflash&Itemid=50.

31. El Patrón MVC (Modelo Vista Controlador). In: *Prestashop 5 Estrellas* [online]. [Accessed 11 abril 2013]. Available from: http://prestashop5estrellas.wordpress.com/2010/03/29/el-patron-mvc-modelo-vista-controlador/.

32. ANDRÉS GROSSO. Patrones GRASP. In: *Prácticas de Software* [online]. 21 marzo 2011. [Accessed 11 abril 2013]. Available from: http://www.practicasdesoftware.com.ar/2011/03/patrones-grasp/.

33. Que es SVN. In: *Que es SVN* [online]. [Accessed 15 noviembre 2012]. Available from: http://lihuen.linti.unlp.edu.ar/index.php?title=C%C3%B3mo_usar_SVN.

34. SAMESPINOSA. MAPA DE PROCESOS. In: [online]. Business & Mgmt. S.l. 20 febrero 2009. [Accessed 10 abril 2013]. Available from: http://www.slideshare.net/samespinosa/mapa-de-procesos-1053479.

35. SEVILLANO, Fernando. redindustria: Definición de Proceso de Negocio (I). In: *redindustria* [online]. [Accessed 10 abril 2013]. Available from: http://redindustria.blogspot.com/2009/04/definicion-de-proceso-de-negocio-i.html.

36. INEI - 2.3 El Modelo conceptual. In: [online]. [Accessed 10 abril 2013]. Available from: http://www.ongei.gob.pe/publica/metodologias/Lib5011/cap2-3.htm.

37. FELIPE HERNÁNDEZ SALAZAR Y IVIS MARAY AMARÓ MORENO. *Componente para el control de acceso en el marco de trabajo Symfony 1.3 para el proyecto productivo Sistema de Informatización de la Gestión Fiscal II.* UCI: UCI, 2012.

38. Mod Datos. In: [online]. [Accessed 7 junio 2013]. Available from: http://labredes.itcolima.edu.mx/fundamentosbd/sd_u2_1.htm.

39. *Diagrama de componentes* [online]. S.l.: s.n., 2013. [Accessed 3 junio 2013]. Available from: http://es.wikipedia.org/w/index.php?title=Diagrama_de_componentes&oldid=65548694.

40. DIAGRAMA DE CLASES. In: [online]. [Accessed 11 abril 2013]. Available from: http://es.scribd.com/doc/19776998/DIAGRAMA-DE-CLASES.

41. Diagramas de secuencia. In: *Diagramas de secuencia UML: Referencia* [online]. [Accessed 11 abril 2013]. Available from: http://msdn.microsoft.com/es-es/library/dd409377.aspx.

42. Prototipo de interfaz de usuario. In: [online]. [Accessed 11 abril 2013]. Available from: http://es.scribd.com/doc/111600953/25/Prototipo-de-interfaz-de-usuario.

43. Estandares de Programacion? In: [online]. [Accessed 12 abril 2013]. Available from: http://espanol.answers.yahoo.com/question/index?qid=20070207200508AALC8lG

44. *PascalCasting* [online]. S.l.: s.n. [Accessed 12 abril 2013]. Available from: http://chitita.uta.cl/cursos/2011-1/0000966/recursos/r-5.pdf.

45. Diagramas de Despliegue. In: [online]. [Accessed 12 abril 2013]. Available from: http://es.scribd.com/doc/19611312/diagramas-de-despliegue-2222.

46. Métricas de Software. In: [online]. [Accessed 2 mayo 2013]. Available from: http://www.desarrolloweb.com/articulos-copyleft/articulo-metricas-de-software.html.

47. M. VASQUEZ C. FALCATO. Prueba de software. In: [online]. [Accessed 2 mayo 2013]. Available from: http://html.rincondelvago.com/prueba-de-software.html.

48. Pruebas de software (Objetivos). In: [online]. [Accessed 2 mayo 2013]. Available from: http://www.ecured.cu/index.php/Pruebas_de_software.

49. CARRILLO, Fernando. Pruebas del Software: Niveles de Prueba del Software. In: *Pruebas del Software* [online]. 3 enero 2011. [Accessed 3 mayo 2013]. Available from: http://ingenieriadepruebasdelsoftware.blogspot.com/2011/01/niveles-de-prueba-del-software.html.