Universidad de las Ciencias Informáticas

Facultad 2

Aplicación informática para gestionar la Guardia Obrera Estudiantil en las facultades de la Universidad de las Ciencias Informáticas

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor(es): Raúl Enrique Liranza Chacón

Kenier Isidoro Cedré Gutiérrez

Tutor (es): Ing. Javier de León Barral

Ing. Odiel Estrada Molina

La Habana, junio de 2016

“Año 58 de la Revolución”

Aplicación informática para gestionar la Guardia Obrera Estudiantil en las facultades de la Universidad de las Ciencias Informáticas	Declaración de autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Raúl Enrique Liranza Chacón Kenier Isidoro Cedré Gutiérrez

_____ _____

Ing. Javier de León Barral Ing. Odiel Estrada Molina

Dedicatoria

Raúl:

Dedico este trabajo a mis padres Marcia y Raúl, a mi hermano Yunier y amigos.

Kenier:

A mi mamá María de los Angeles y mi papá Isidoro, esto también es por ustedes.

A mi esposa Haidi.

A mi familia y amigos.

Agradecimientos

Raúl:

Agradezco a mis vecinos que siempre estuvieron preocupados por mi trayectoria desde el comienzo de mis estudios, a mis amigos Yanet, Cifredo, Arletis, Isvany, Kenier que es mi compañero de tesis, a Daniel, a mis padres y mi hermano que siempre estuvieron al tanto de todo, a mis tías Mima y Ramona, a mi abuela Lucidia y a mis tutores Javier y Odiel y a Julia mi especial y querida amiga. También a las personas del Departamento de Economía como Claribel, Michel que siempre estuvieron apoyándome.

Kenier:

A mis padres: María de los Angeles e Isidoro, por darme el ser, el amor, el cariño, la confianza, los esfuerzos y el tiempo que se requiere para con un hijo(nunca me he sentido con falta de todo ello), agradezco porque me educaron y sirvieron de ejemplo en la constancia, el trabajo duro y el compromiso con uno mismo y para con los demás, por todo su apoyo y consejos para ser un hombre de bien, les aseguro que ha sido fructífera su labor de educar, por ser únicos, especiales y mi mayor tesoro. Los amo.

A mi esposa Haidi: por darme su amor, comprensión, su amistad, confianza, por ser mi compañera, por compenetrar tan bien conmigo que vamos codo a codo desdoblándonos día a día, como digo: "somos el duo dinámico", espero nunca cambies, te amo mi vida.

A toda mi familia, en especial a Miriam y Dunielbys por cuidar desde su pedacito tan bien a mis padres, no saben la tranquilidad que me dan, para seguir desde acá la batalla, a Tío Juan, Zoe, mis primas Yine y Sofía por la acogida y atenciones en los momentos necesarios.

A mi suegra Mayda, su esposo Rafael y su familia por sus consejos y acogida. A mis abuelos habaneros como les llamo: Nilo Y Aracely, me ha dado un inmenso gusto poder conocerlos y emprender junto a ustedes.

A mi amiga y compañera Yanet por su paciencia, ayuda y respuestas en las horas de estudio y trabajos realizados a lo largo de estos años, en esta carrera has sido mi coopiloto. De más está decir que siempre has podido y puedes contar conmigo.

Agradecimientos

A mis amigos a lo largo de mi vida: Ernesto (el gordo), Jorgito, Yender, Sumaily, Alejandro Silva, Eiquel, Arnaldo, Juan Daniel, Niurka Rodes y su mamá Caridad, Keniel. De una forma u otra han estado ahí para darme su apoyo y ayuda cuando la he necesitado, saben que siempre bajo cualquier circunstancia ahí estaré para ustedes.

A los tutores Javier y Odiel, por brindarnos sus conocimientos y tiempo, así como a mi compañero de tesis Raúl que ambos hemos cumplido la meta, gracias a los 3. A los miembros del tribunal: Rosa, Aurelio, Héctor y Leandro por sus consejos y su paciencia.

A todos los compañeros de aula y de la facultad, de los dos grupos por los que he transitado en mi carrera, especialmente: Mojena, Keiser, Yasser, Kathy, Andrés Bedoya, Jose Antonio Cascaret, Nurys, Eddry, Victor, El Mago, cuando hizo falta activar las neuronas en momentos de pruebas o difíciles ahí estábamos y salíamos victoriosos tanto en el día a día como en las exámenes.

A todos los profesores a lo largo de mi vida como estudiante, especialmente a los profes y directivos de la facultad 2 por sus buenas clases, ejemplos y constancia para con los estudiantes, también a la UCI en general, que más que una universidad ha sido una escuela: docente y para la vida.

Resumen

La UCI cuenta con una gran cantidad de recursos, sobre todo de carácter tecnológico, los cuales por su importancia se hace necesario preservar y cuidar. Es por ello que la organización y realización de la Guardia Obrera Estudiantil debe hacerse por sus propios dueños y no solo por los cuerpos de vigilancia y protección. Actualmente el proceso de planificación y control de la guardia en la UCI se hace difícil y engorroso pues se maneja gran número de información y se generan los reportes de forma manual, lo que implica la pérdida de información y en muchos casos su duplicación. Tampoco se generan estadísticas para apoyar el proceso.

La solución que se propone en el presente trabajo pretende dar respuesta a esta problemática, mediante la creación de una Aplicación informática para gestionar la Guardia Obrera Estudiantil en las facultades de la UCI. Para desarrollar el trabajo se estudiaron sistemas informáticos similares al que se deseaba implementar. Se describieron las principales tecnologías, herramientas y la metodología de desarrollo. Se generaron los artefactos de acuerdo a la metodología seleccionada, se implementaron las funcionalidades definidas y se realizaron pruebas para verificar la calidad del sistema implementado.

El resultado del trabajo es una aplicación informática que de forma automatizada realiza la planificación de la guardia, permitiendo una mejor planificación y control de la guardia en la UCI y la obtención de reportes que apoyan el proceso. Además permite que esta planificación sea consultada por cualquier persona desde cualquier área de la universidad.

Palabras clave: Aplicación informática, Guardia, Planificación

Índice General

Índice de Figuras

Índice de Tablas

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. “La sociedad contemporánea sin la informatización no tiene sentido… Hoy no podemos hablar de ningún proceso productivo o de servicios si no se informatiza”. (1)

El objetivo fundamental que se persigue con la informatización es que los servicios se realicen de la forma más sencilla, rápida y menos costosa posible, mediante un software que posea una alta calidad. La práctica ha demostrado que la informatización de procesos permite racionalizar el trabajo, reducir costos y recursos, además asegurar una mejor calidad en el trabajo que se realiza.

La informatización de procesos depende en gran medida de los sistemas de gestión de la información, los cuales han tenido un auge en los últimos años. Estos han provocado que la mayoría de las empresas tengan la necesidad de enfrentar desafíos cambiantes en aras de obtener beneficios, a tal punto que se les ha hecho imprescindible contar con estos sistemas para un mejor desarrollo.

Cuba no queda atrás en su deseo de informatizar los servicios y utilizar los sistemas de gestión de la información. En el año 2002 se crea, como una idea del Comandante en Jefe Fidel Castro Ruz, la Universidad de las Ciencias Informáticas (UCI). La UCI surge como un centro docente-productor que desarrolla aplicaciones y servicios informáticos orientados a diversos sectores de la economía y los servicios, dentro y fuera de Cuba.

La UCI cuenta con una gran cantidad de recursos, sobre todo de carácter tecnológico, lo cuales por su importancia se hace necesario preservar y cuidar. La organización y realización de la Guardia Obrera Estudiantil (GOE) se ha consolidado, basándose en el contenido ideológico que entraña el cuidado de los centros por sus propios dueños y no solo de los cuerpos de vigilancia y protección. A partir del curso 2012-2013 hasta el 2015-2016 la GOE como proceso fundamental de la institución para el cuidado y custodia de sus recursos y bienes materiales, ha transitado por varios cambios en su concesión.

Para comprender el proceso de la GOE fue realizada una entrevista al Vicedecano de Administración, Vicedecano de Extensión y Residencia y a las secretarias de la facultad 2,

máximos responsables de planificar y controlar el proceso de la guardia de los estudiantes y trabajadores. Como resultado de la entrevista se evidencia las siguientes limitaciones:

- Planificación ineficiente, ya que a consideración de los autores e involucrados en el proceso y atendiendo a la revisión de la documentación consultada, este se realiza de manera manual (solamente se utilizan documentos de Word y Excel para el almacenamiento de la información recogida sobre la guardia en el momento de planificarla y enviar reportes). La socialización de la información desde el planificador al usuario final no llega con el tiempo requerido y el margen de error de coincidencias en postas, turnos y horarios provoca inconformidades en los involucrados.
- A pesar que los estudiantes y trabajadores estén informados de la planificación de la guardia a través del correo, puede ocurrir el caso que el correo llegue con demora y que el 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. Actualmente se envía la planificación del mes posterior y las personas involucradas no se les avisa, o no se les notifica días antes de que le corresponde la guardia.
- Como se tienen tantas variantes a la hora de la planificación y esta es realizada por una sola persona, se da el caso que la planificación de los equipos de guardia tienen muy poco tiempo entre una guardia y otra; o por el contrario, están muy espaciadas la una de la otra; o repite el mismo equipo en varios fines de semanas mientras otros no se afectan en estos tipos de días.
- Limitado acceso a la información generada en el proceso de planificación.
- El proceso que se realiza en la actualidad no permite la generación automática de reportes estadísticos, lo cual repercute en la toma de decisiones.
- Si el trabajador está de vacaciones, o se encuentra cumpliendo misión; si está de certificado médico, o licencia, tanto de maternidad como estudiantil, o está de cumpleaños no se tiene en cuenta en la planificación de la guardia.

Lo antes señalado permitió constatar que la gestión de la guardia obrera en las facultades de la UCI no se realiza de forma efectiva ya que no permite la generación de reportes actuales ni su consulta en cualquier instante de tiempo ya sea por los estudiantes, directivos, planificadores y demás trabajadores de la universidad.
Lo antes expuesto constituye la problemática que origina esta investigación de la cual se define el siguiente **problema a resolver**: La actual gestión de la GOE en las facultades de la UCI, no garantiza su correcta planificación y control.

Teniendo en cuenta el problema identificado, **el objeto de estudio** es: Gestión de la planificación y control; mientras que el **objetivo general** que se persigue con el desarrollo del trabajo es desarrollar una aplicación informática para gestionar la GOE en las facultades de la UCI que permita la planificación y control.
El **campo de acción** lo constituye: Aplicaciones para la gestión de planificación y control.
Del objetivo general se desglosan los siguientes **objetivos específicos**:

- Elaborar los referentes teóricos, tecnológicos e ingenieriles fundamentales para el desarrollo de la aplicación informática.
- Caracterizar el modelado del negocio para comprender como funcionan los procesos que se desean automatizar.
- Elaborar la arquitectura y el diseño de la aplicación informática.
- Elaborar el modelo de implementación de la aplicación informática así como constatar su funcionalidad a partir del diseño, ejecución y valoración de las pruebas de software.

Tareas de la investigación:

- Revisión bibliográfica para conformar el estado del arte de la investigación mediante la búsqueda de información sobre soluciones similares aplicadas en diversos entornos.
- Realización de un estudio de los sistemas de guardia existentes, tanto a nivel nacional como internacional, para conocer características y funcionalidades que pudieran ser aprovechadas en la realización de la aplicación informática.
- Determinación de las herramientas, lenguajes de programación, metodología de desarrollo de software a emplear para la implementación de la aplicación informática.

- Elaboración de los artefactos asociados a la metodología de desarrollo de software escogida para la modelación del negocio de la aplicación informática.
- Elaboración de la arquitectura y diseño de la aplicación informática.
- Implementación de la aplicación informática para gestionar la Guardia Obrera Estudiantil de las facultades de la UCI que permita la planificación y control.
- Validación de los resultados obtenidos a través de pruebas de caja negra lo cual permitirá constatar la funcionalidad de la aplicación informática implementada.

Para dar cumplimiento a las tareas propuestas anteriormente, se utilizaron los siguientes métodos científicos de investigación teóricos y empíricos.

Métodos Teóricos:

- **Analítico-Sintético:** permitió analizar las características de los sistemas de gestión de guardia existentes y definir sus características fundamentales, ventajas y desventajas.
- **Modelación:** permite modelar mediante diagramas los procesos referentes a la planificación y control de la GOE.

Métodos Empíricos:

- **Observación:** permitió obtener la información necesaria para el desarrollo de los servicios de guardia y las características de los procesos que tienen lugar el mismo.
- **Entrevista:** permitió conocer las necesidades del cliente y obtener los requisitos funcionales y no funcionales que debe cumplir el sistema.

El presente trabajo está estructurado en 4 capítulos, tal y como se muestra a continuación:

Capítulo 1: Fundamentación Teórica

En este capítulo se muestran los conceptos fundamentales relacionados con la investigación. Se realiza el estudio de sistemas de planificación de guardia, tanto nacional como internacional. Además se muestra el ambiente de desarrollo de la aplicación.

Capítulo 2: Características del sistema

En este capítulo se muestran los artefactos correspondientes a las disciplinas Modelamiento del Negocio y Requerimientos. Dentro de estos se encuentran los Diagramas de Casos de Uso del

Negocio y el Diagrama de Casos de Uso del Sistema. Además se especifican los requisitos funcionales y no funcionales.

Capítulo 3: Arquitectura y diseño

En este capítulo se muestra el diseño del sistema, mostrando los diagramas de clases del diseño y los diagramas de secuencia. También se muestra el modelo lógico y físico de la Base de Datos. Además se hace referencia al estilo arquitectónico Modelo Vista Controlador.

Capítulo 4: Implementación y prueba

En este capítulo se refleja el modelo de implementación, específicamente el diagrama de componentes y el diagrama de despliegue. Además se muestran las pruebas realizadas a la aplicación y el resultado de estas.

Capítulo 1: Fundamentación teórica.

Introducción al capítulo

En este capítulo se realiza un análisis de los principales conceptos asociados a los sistemas de guardia. Se exponen las características de algunos de los sistemas de guardia actuales que permiten comprender su proceso y funcionamiento. Se describen brevemente la metodología, las herramientas, tecnologías y el lenguaje que se utilizan en el desarrollo de la aplicación informática.

1.1 Conceptualización de aplicación de gestión de guardia

Para poder conceptualizar los sistemas de gestión de guardia es necesario definir cada uno de los elementos que lo integran. Es por ello que a continuación se muestran los conceptos fundamentales asociados para lograr un mejor entendimiento de lo que se plantea en el desarrollo del trabajo.

Aplicación: Una aplicación también es un programa informático, es decir; forma parte de lo que comúnmente llamamos software. Un software de aplicación es básicamente: un conjunto de programas de computación desarrollados para realizar, en combinación con la actividad humana, tareas o procesos específicos relacionados, en general, con el procesamiento de la información. (2)

Existen dos tipos fundamentales de aplicaciones: las aplicaciones de escritorio y las aplicaciones web.

Aplicaciones de escritorio: Son aplicaciones para ser instaladas, configuradas y utilizadas en ordenadores de escritorio. Pueden ser desarrolladas para el sistema operativo que requiera y contener funcionalidades adaptadas a las necesidades de la entidad o del problema en cuestión.

Aplicaciones web: Son soluciones puntuales expresadas en un software navegable, por lo tanto se operan en la web sin necesidad de requerir su instalación; por esta razón son de amplio alcance.

Para desarrollar el sistema se selecciona una aplicación web porque brinda las sigueinetes ventajas.

- ✓ Portabilidad: se ejecutan desde cualquier ordenador con conexión a internet.

- ✓ La información que manejan es accesible a través de internet, por lo que son especialmente interesantes para desarrollar aplicaciones multiusuario basadas en compartir información.
- ✓ Son aplicaciones muy ligeras (el Navegador de Internet no contiene el programa) por lo que el Usuario no necesita tener un ordenador de grandes prestaciones para trabajar con ellas.
- ✓ Su funcionalidad es independiente del sistema operativo instalado en el ordenador del usuario.
- ✓ No hay problemas de incompatibilidad entre versiones, porque todos los Usuarios trabajan con la misma

En el caso específico de un Sistema de guardia, la información que se maneja requiere ser consultada desde cualquier lugar de la universidad, además que debe ser difundida y de conocimiento para todo el personal que realiza la Guardia Obrera Estudiantil.

Planificar: la planeación es la selección de misiones y objetivos, las estrategias, políticas, programas y procedimientos para lograrlos; es también la toma de decisiones; o selección de un curso de acción entre varias alternativas. (3)

Turno: se considera un trabajo por turno toda forma de organización del trabajo en equipo según la cual los trabajadores ocupan sucesivamente los mismos puestos de trabajo, según un cierto ritmo, continuo o discontinuo, implicando para el trabajador la necesidad de prestar sus servicios en horas diferentes en un período determinado de días o de semanas. (4)

Grupo: Se denomina grupos a los distintos conjuntos de personas que se reúnen para la consecución de metas concretas. Algunos de los aspectos que pueden caracterizar a un grupo son la comunicabilidad que sus miembros establecen entre sí, su dependencia mutua para llevar a cabo las metas dispuestas, una serie de fines que se buscan cumplir. (5)

Reporte: Se denomina reporte a aun cuerpo de información destinado a servir de análisis sobre un tópico determinado. Un reporte puede revestir diversas formas, ya sea como escrito, como charla, como informe televisivo o como película documental. El uso de reportes se extiende en el plano gubernamental, en el privado, en el área educativa en el campo científico, etc. Lo importante a considerar para la confección del reporte es la claridad que el elemento seleccionado arroja sobre la cuestión tratada. (6)

Asistencia: Es la acción y resultado de asistir**,** la presencia actual o de hallarse presente. Una aglomeración de personas que está presente en un acto. El acto de prestar amparo, ayuda, socorro, protección o apoyo. Dignidad, cargo, puesto o empleo del asistente de cualquier cargo administrativo o de un funcionario público. (7)

Luego de definir algunos conceptos los autores del presente trabajo consideran que una aplicación de gestión de guardia es un programa informático para administrar la defensa, custodia y protección de medios. Sus objetivos fundamentales son precisar la organización y elaborar la estrategia de la guardia y cuidado de forma automática, en combinación con la actividad humana y el proceso de la información.

1.2 Sistemas internacionales e institucionales de gestión de guardia

En este epígrafe se realiza un análisis de los sistemas estudiados a nivel internacional e institucional, facilitando características de los mismos para la toma de decisiones y la selección de herramientas y tecnologías para el desarrollo de la aplicación.

1.2.1. Sistemas de Gestión de Guardia en el mundo

SICATU (**SI**stema de **CA**lculo de **TU**rnos)

SICATU surgió para dar solución a la planificación de guardias de un departamento de cardiología. La idea era confeccionar una interfaz de usuario sencilla, de fácil manipulación, que detallara en todo lo posible todos los elementos implicados y presentara la solución en forma de informe: (8)

- Personal.
- Vacaciones de cada personal.

- Días festivos.
- Tipos de guardias.
- Posibles preasignaciones (guardias impuestas antes del cálculo)
- Calendarios calculados
- Distintos tipos de informes.

El software es una aplicación de escritorio hecha en JAVA con interfaz en SWING y con acceso a base de datos MySQL. La aplicación ofrece la posibilidad de generar distintos tipos de informes: Calendarios, listados del personal, estadísticas por médico, etcétera. Estos informes pueden guardarse en diferentes formatos: pdf, rtf y docx. (9)

Módulo Calendario de turnos del sistema Tamigo

Tamigo es un proveedor en la gestión *online* de la planificación del trabajo en Escandinavia y otros países europeos. Es una solución web 100% online, que le permite un control total en el proceso para gestionar el personal. De forma automática el cliente puede transferir los turnos a su calendario privado como Google, Outlook o iPhone, entre otros. Cada vez que se actualice la planificación de turnos, lo hará también el calendario de forma automática y lo mostrará en el calendario. (10)

Dentro de las funcionalidades que brinda el sistema se encuentran:

- Planificación para el futuro: no es necesario introducir el calendario de turnos cada semana, basta con hacerlo sólo una vez si utiliza una plantilla. Además brinda la oportunidad de asignar un turno específico para cada empleado sin reescribir el actual.
- Creación de un calendario de turnos: permite introducir los turnos directamente en el calendario. Permite guardar los cambios y “Compartir” el calendario con los empleados de forma online y al instante.
- Cambio de turnos: permite aceptar o rechazar las solicitudes de cambio de turno de los empleados o el deseo de intercambiar los turnos entre ellos. Esto le proporciona una visión completa y 100% clara sobre los turnos que se actualiza automáticamente cuando el administrador o gestor ha tomado una decisión.

Jano Planificación Hospitalaria

Jano Planificación Hospitalaria es un sistema desarrollado en España. Es específico para centros sanitarios y hospitales que precisan de la planificación de turnos de trabajo y organización del tiempo de las personas. Permite definir las necesidades de planificación de puestos de trabajo, turnos y personal de guardia, observando las características necesarias, las normas regulatorias y los convenios laborales sectoriales. El entorno de trabajo aportado por las soluciones Jano le permite además definir de forma personalizada cada centro y su zonificación (edificios, áreas, espacios, etc.) en el cual se asignan los puestos de trabajo. (11)

La aplicación incorpora controles y asistentes que facilitan la labor de organización, optimización y reparto de cargas de trabajo optimizadas además del gestor de cuadrantes de planificación. Jano presenta información muy interpretable acerca de necesidades de planificación y necesidades de contratación de personal.

Solución inteligente para la planificación horaria y el control de accesos (Visual Time Live)

Es una solución diseñada para resolver las necesidades de seguridad y flexibilidad horaria de pequeñas y medianas empresas, permitiendo la reducción de tiempos y costes de gestión. Facilita la gestión del tiempo de presencia de los empleados de forma automatizada. Permite conocer, en todo momento, información acerca de las incidencias que se producen sobre el plan horario (ausencias, retrasos, excesos) facilitando la toma de decisiones. Los procesos relativamente complejos se realizan con la ayuda de un asistente, el cual guía, paso a paso, al gestor de aplicaciones en la creación de nuevos objetos. (12)

Para cada uno de los horarios definidos se pueden registrar condiciones específicas adicionales en función de la divergencia entre lo que debería haber hecho un empleado y lo que realmente hizo, minimizando el trabajo de administración del departamento de Recursos Humanos. Visual Time permite la generación de informes, teniendo en cuenta algunos indicadores organizados por diferentes categorías (configuración, explotación, previsión y evolución).

TURNEX

TURNEX es un software desarrollado por SOLEX para la gestión operativa detalla de turnos de trabajo de personas. Con su implementación es posible programar los turnos y/o horarios con anterioridad, permitiendo controlar las asistencias, días de descanso, estados y horas extras.

La herramienta permite integrar los sistemas de nómina para la gestión del pago de remuneraciones, así como también, llevar bitácoras de trabajo y hojas de vida en recursos humanos o empresas industriales con personal interno en turnos rotativos. Es un sistema que otorga visibilidad de las operaciones en turnos, con mayor programación y control, disminuyendo con eso los errores en los pagos de sueldo y en los cobros a clientes. (13)

1.2.2. Sistemas de Gestión de Guardia en la UCI

En la UCI han existido, desde su creación, diversos sistemas informáticos dedicados a la gestión de la Guardia Obrera Estudiantil, los cuales serán caracterizados a continuación.

Gestor Web para el control de la Guardia Obrera en la UCI

Esta aplicación se encarga de automatizar el trabajo de las entidades relacionadas con el proceso de control de la guardia obrera así como la generación de reportes con los resultados de este proceso. Además permite el control de los cuatro tipos de guardia (Guardia Obrera de trabajadores Internos, Guardia Obrera de trabajadores Externos, Guardia Obrera de Vicerrectoría de Logística y Guardia Obrera de Oficiales Operativos) por parte del Puesto de Mando de la Guardia, quienes se encargarán de su asistencia y realización.

Ofrece la posibilidad de realizar reportes semanalmente relacionados con todo el proceso de control de la guardia obrera haciendo que su generación resulte menos compleja. Fue ubicada en el Puesto de Mando de la Guardia y su principal administrador es el Jefe del Puesto de Mando de la Guardia. (14)

La principal desventaja que presenta el sistema es que no cubre las funcionalidades relacionadas con el proceso de planificación de la GOE, solamente las referidas al control de la misma.

Sistema para la Gestión de la Guardia Obrera Estudiantil en la Facultad 3

Para dar solución a las dificultades existentes en la gestión de los problemas relacionados con la Guardia Obrera Estudiantil de la Facultad 3, se desarrolla un sistema informático. El mismo brinda

la posibilidad de gestionar la planificación de la guardia en la Facultad 3, donde el sistema genera 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 tienen en cuenta algunas limitantes que son de suma importancia para la confección de dicha planificación. Permite los 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. (15)

A pesar de que el sistema es capaz de generar la planificación de guardia de manera automática presenta algunos inconvenientes tales como que no siempre la base de datos está actualizada con las personas que van a estar involucradas en la guardia realmente. No cuenta con funcionalidades que permitan realizar la planificación de las personas externas involucradas en la guardia. No se encuentra preparado para poder ser ejecutada en otras áreas de la universidad. No permite la generación de reportes que apoyen la toma de decisiones y permita un mejor control sobre la realización de la guardia. No se registra el control de cumplimiento de la guardia.

Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7

El Sistema surge como una ampliación de funcionalidades del sitio de la Facultad7 de la Universidad de las Ciencias Informáticas, PortalFacultad7. El objetivo fundamental es 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.

El sistema permite llevar un control personalizado de la guardia y la cuartelería en la residencia estudiantil. Facilita la evaluación de los estudiantes y genera un conjunto de reportes, en formato digital, acerca de estas dos actividades. Estos reportes pueden ser impresos. (16)

El principal inconveniente es que es específico para la guardia de los estudiantes y no se tienen en cuenta a los trabajadores. Además tampoco contempla el proceso de control de la guardia, elemento importante para la evaluación del cumplimiento de esta actividad.

Aplicación web para la gestión de la planificación de la Guardia Obrera Estudiantil en la Universidad de las Ciencias Informáticas

Este sistema permite la gestión de la planificación de la Guardia Obrera Estudiantil en el UCI. Utiliza un algoritmo de selección estratégica para evitar la repetición de los días, turnos y postas de guardia, de manera que se satisfagan las necesidades básicas para el cumplimiento de la misma, tanto para estudiantes y trabajadores involucrados en este proceso. (17)

Su principal inconveniente es que permitía ver las guardias y evaluaciones pero solo para los administrativos y no para los usuarios comunes. No permite llevar el control de la guardia. No se generan reportes sobre el cumplimiento de la guardia.

1.2.3. Conclusión sobre utilización de los sistemas estudiados

Para arribar a elementos concluyentes sobre el estudio de los sistemas antes caracterizados existentes en la UCI, se hace necesario realizar una comparación entre ellos, teniendo como criterios de comparación las necesidades actuales del proceso de guardia.

Criterios evaluar	Gestor Web para el control de la Guardia Obrera en la UCI	Sistema para la Gestión de la Guardia Obrera Estudiantil en la Facultad 3	Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7	Aplicación web para la gestión de la planificación de la Guardia Obrera Estudiantil en la Universidad de las Ciencias Informáticas
Realiza la planificación de la guardia	No	Si	Si	Si
Realiza el control de la guardia	Si	No	No	No
Genera reportes sobre cumplimiento de la guardia	Si	No	Si	No
Genera estadísticas	No	No	No	No
Se tienen en cuenta los trabajadores externos que realizan guardia	Si	No	No	No
Permite realizar cambios y permutas	No	Si	No	No

Se notifica la planificación al personal involucrado en la guardia	No	Si	No	No

En la tabla mostrada se puede apreciar que no existe ningún sistema en la UCI que satisfaga en su totalidad las necesidades actuales de la GOE, sobre todo si se toman como elementos fundamentales la planificación y el control de la guardia. Es válido destacar que existen elementos y escenarios comunes que pueden ser utilizados como guía para el desarrollo de un sistema de gestión de guardia. Entre estos elementos se encuentran funcionalidades, procesos, métodos y comportamiento.

Los sistemas existentes internacionalmente presentan funcionalidades que pueden tenerse en cuenta para desarrollar la solución, pero tiene como principal limitante que no son de código abierto, lo cual implica que su utilización conlleva al pago de licencias, incurriendo en un gasto innecesario de recursos.

Teniendo en cuenta los elementos expuestos anteriormente, se concluye que es necesario realizar un nuevo sistema de Gestión de la Guardia Obrera Estudiantil en la UCI, que cumpla con las necesidades actuales de este proceso.

1.3 Algoritmos de planificación de recursos mediante problemas de optimización

La programación automática, cuando se utiliza correctamente, es una de las herramientas más poderosas para mejorar la eficiencia y el coste de gestión de proyectos, planificación, o de fabricación. El objetivo de un algoritmo automático para un problema relacionado con la programación es producir una muy buena (es decir óptima o casi óptima) planificación para un conjunto de actividades y / o de los recursos que se les asignan. Esto puede implicar: (18)

- ✓ Programación: La determinación de la secuencia de actividades, y los tiempos de cada uno comienza y termina.
- ✓ Asignación: La asignación de recursos (por ejemplo, personas, máquinas) para las actividades.

Los problemas relacionados con la programación se pueden dividir en tres categorías:

- ✓ Problemas de programación relacionadas con la asignación: se centran en la asignación, en donde la secuencia y los tiempos de las actividades está predeterminado.

- ✓ Problemas de solo programación: se centran en la determinación de la secuencia y los tiempos de las actividades (generalmente conocido como tareas), donde la asignación de recursos a las actividades es predeterminada o ignorado.

- ✓ Articulación de Programación / Asignación Problemas: se deben determinar la secuencia y los tiempos de las actividades, así como la asignación de recursos a ellos. Estos problemas incluyen taller de programación, gestión de la cartera avanzada y programación de proyectos con asignación.

Los problemas relacionados con el horario son parte de una clase más amplia de los problemas llamados problemas de optimización, que tienen dos características importantes: (18)

- ✓ Restricciones: una descripción formal de los requisitos que deben cumplir las soluciones candidatas a un problema, por ejemplo, que una determinada tarea no puede comenzar hasta algunos otros finalice la tarea.

- ✓ Funciones objetivo: una caracterización matemática de la calidad de una solución. En general, una función objetivo tiene como objetivo minimizar (o maximizar) algún valor cuyo cálculo se basa en los datos de una solución candidata, por ejemplo, minimizar el tiempo que se tarda en obtener todas las actividades realizadas.

Existen algoritmos que pueden encontrar una solución óptima (uno que produce el valor mínimo/ máximo de la función objetivo) en un tiempo "razonable" (en relación con el tamaño del problema) para los problemas muy simples relacionados con la programación, en particular, para las versiones simples de programación de mano de obra y la programación básica del proyecto. Sin embargo, para casi todos los problemas relacionados con la programación que se han mencionado, los algoritmos óptimos simplemente toman demasiado tiempo. Así que para la mayoría de estos problemas, es necesario utilizar algoritmos heurísticos, que pueden producir

soluciones que son casi óptima, es decir, razonablemente cerca de la solución óptima, en un plazo de tiempo razonable.

Hay un número de enfoques heurísticos comunes a los algoritmos utilizados para abordar los problemas relacionados con la programación, que puede ser dividido en:

- ✓ Algoritmos de construcción (incluyendo algoritmos *Greedy* o voraces), que comienzan con una solución vacía o incompleta (por ejemplo, cuando no hay tareas programadas y / o no se asignan recursos), y de manera progresiva hacen que sea más completa (por ejemplo, mediante la programación de una tarea adicional y / o la asignación de un recurso adicional a la vez).

- ✓ Los algoritmos de búsqueda que comienzan con una o más soluciones completas de candidatos, y de manera progresiva se combinan y / o se modifican con el fin de generar mejores soluciones completas. Estos incluyen enfoques (también denominadas metaheurísticas), tales como la Escalada de Colina, Búsqueda Tabú, el Recocido Simulado, Optimización por Enjambre de Partículas, Colonia de Abejas, Algoritmos Genéticos y la Evolución Diferencial.

- ✓ Algoritmos híbridos que utilizan una combinación de construcción y los enfoques basados en la búsqueda.

1.3.1 Selección del algoritmo

Para el desarrollo del sistema se seleccionó un algoritmo de construcción, específicamente un algoritmo *Gready*.

Los algoritmos *Greedy* presentan las siguientes características (19)

- ✓ Son algoritmos que toman decisiones de corto alcance, basadas en información inmediatamente disponible, sin importar consecuencias futuras.

- ✓ Suelen ser bastante simples y se emplean sobre todo para resolver problemas de optimización.

- ✓ Son rápidos en encontrar una solución (cuando la encuentran), la cual no siempre es la más óptima, siendo esta una de sus principales desventajas.

La estrategia general de este tipo de algoritmos se basa en la construcción de una solución, la cual comienza sin elementos y cada vez que debe tomar algún tipo de decisión, lo hace con la información que tiene a primera mano, la cual de alguna manera le permita adicionar elementos y así avanzar hacia la solución total. Cada elemento o paso de la solución se adiciona al conjunto solución y así hasta llegar a la solución final o a un punto en el cual no puede seguir avanzando, lo cual indica que no encontró una solución al problema.
Estos algoritmos utilizan los siguientes elementos:

- ✓ El conjunto C de candidatos, entradas al problema.
- ✓ Función solución, esta comprueba, en cada caso, si el subconjunto actual de candidatos elegidos forma una solución (no importa si es óptima o no).
- ✓ Función de selección, informa cuál es elemento más prometedor para completar la solución. Este no puede haber sido escogido con anterioridad. Cada elemento es considerado una sola vez. Luego, puede ser rechazado o aceptado y pertenecerá a C/S.
- ✓ Función de factibilidad, informa si a partir de un conjunto se puede llegar a una solución. Se une al conjunto de seleccionados unido con el elemento más prometedor.
- ✓ Función objetivo, es aquella que se quiere maximizar o minimizar, el núcleo del problema.

1.4 Ambiente de desarrollo

La correcta selección de métodos, tecnologías y herramientas favorecen el desarrollo de productos de *software* con mayor calidad y reducen los costos.

1.4.1 Metodología de desarrollo: Variación AUP-UCI

En correspondencia con el modelo productivo de desarrollo de la UCI se decide utilizar como metodología de desarrollo la Variación AUP-UCI (Proceso Unificado Ágil, AUP por sus siglas en inglés de *Agile Unified Process*). La variación AUP para la UCI establece tres fases: Inicio,

Ejecución y Cierre. Entre las disciplinas que contemplan se encuentran: Modelado de Negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, pruebas de liberación, Pruebas de aceptación, Despliegue, Gestión y Soporte. (20)

Como bien establece el documento, el Modelado ágil es una técnica que utiliza AUP. Es por eso que se utiliza esta técnica para el desarrollo del presente trabajo, encapsulando los requisitos funcionales en Casos de Uso (CU).

1.4.2 Lenguaje de desarrollo, tecnologías y herramientas

Lenguaje de programación: Groovy 2.3

Groovy es un lenguaje de programación con una sintaxis análoga a Java que compila para Java Bytecode y corre en la Java Virtual Machine (JVM). Se integra completamente con Java, le permite mezclar y acoplar al código Groovy y Java con un esfuerzo mínimo.

Groovy es un lenguaje dinámico, es decir, todo ocurre en tiempo de ejecución, incluida la lógica de gestión de llamadas a métodos y acceso a propiedades. Ha sido influenciado por lenguajes como Ruby, Python, Perl, y Smalltalk, así como también Java. A diferencia de otros lenguajes que se adaptaron para la JVM, Groovy fue diseñado para la JVM, así es que no existe incompatibilidad. (21)

Este lenguaje es el seleccionado para darle solución final al sistema ya que como lenguaje se aprovechan todas las ventajas de la tecnología Java. Además de que es un lenguaje dinámico, que disminuye cantidad de código respecto al lenguaje Java e incluye muchas nuevas funcionalidades en su GDK (Análogo al JDK de Java). El *framework* que utiliza este lenguaje, basado y construido para la tecnología Java es Grails.

Framework: Grails 2.4.4

La palabra *framework* define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar. Los objetivos principales que persigue son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. (22)Como *framework* se propone Grails en su versión 2.4.4

Grails es un *framework* de desarrollo web dinámico para la plataforma Java. Utiliza la flexibilidad de Groovy para proporcionar un dominio específico del lenguaje (DSL) para el desarrollo web. Está construido sobre sólidos proyectos de código abierto como Spring, SiteMesh, GORM/Hibernate, todo sobre la máquina virtual de Java. Las aplicaciones desarrolladas con Grails utilizan el patrón MVC (Modelo-Vista-Controlador). (23)

Entre sus principales características se encuentran:

- ✓ La lectura de archivos en Groovy es igual o más fácil que la lectura de archivos en Java.
- ✓ Tiene soporte nativo para manejo de colecciones.
- ✓ Posee soporte nativo para expresiones regulares y rangos.
- ✓ Permite la sobrecarga de operadores.
- ✓ Tiene soporte nativo de lenguajes de marcado, como XML.

El principal motivo de la selección de este *framework* es que soporta el lenguaje de programación Groovy, que fue el seleccionado para el desarrollo del sistema.

Lenguaje de modelado UML 2.1

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Se usa para entender, diseñar, configurar, mantener, y controlar la información sobre tales sistemas. (24) UML permite definir, detallar los artefactos, documentar y construir un sistema. Presenta una gran variedad de formas para dar soporte a una metodología de software, por lo que se aplica en el desarrollo de software. No especifica en sí mismo que metodología o procesos usar. (25)

Herramienta de modelado: Visual Paradigm para UML

Visual Paradigm para UML 8.0 es una herramienta CASE (por sus siglas en inglés que significan *Computer Aided Software Engineering*, Ingeniería de Software Asistida por Ordenador) que soporta el ciclo de vida completo de desarrollo de *software*. Permite construir, visualizar y documentar diagramas de diversos tipos (artefactos que se necesitan para dar solución al

problema de la investigación). Dicha herramienta es una aplicación fácil de instalar y actualizar, además proporciona abundantes tutoriales de UML. (26) Se usa para realizar el modelado de la solución, considerando todas las facilidades brindadas por esta herramienta y sus características:

- ✓ Software libre.
- ✓ Disponibilidad en múltiples plataformas (Windows, Linux).
- ✓ Diseño centrado en Casos de Uso y enfocado al negocio, que genera *software* con mayor calidad.
- ✓ Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.

Sistema Gestor de Bases de Datos: PostgreSQL 8.4

PostgreSQL es un sistema de gestión de bases de datos objeto-relacionales, distribuido bajo licencia de Distribución de *Software Berkeley* (BSD, por sus siglas en inglés) y con su código fuente disponible libremente. Posee buena documentación y está disponible para los sistemas operativos Linux en todas sus variantes, además puede ser utilizado, modificado y distribuido por cualquiera de forma gratuita para cualquier fin, ya sea privado, comercial o académico. (27)

Entre las principales características de PostgreSQL se encuentran: (28)

- Atomicidad (Indivisible): Es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
- Consistencia: Es la propiedad que garantiza que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la BD.
- Aislamiento: Es la propiedad que asegura que una operación no puede afectar a otras. Garantizando que dos transacciones sobre la misma información nunca generarán ningún tipo de error.
- Durabilidad: Es la propiedad que hace posible que una vez realizada la operación, esta persista y no se pueda deshacer aunque falle el sistema.

- Soporte de todas las características de una BD profesional (*triggers*, funciones, consultas, relaciones, reglas, vistas, etc.).

Se utilizó en el desarrollo de la solución para lograr una mayor integración con las tecnologías que se usan actualmente y almacenar la información necesaria para el desarrollo del sistema. Además de que es de distribución gratuita.

Servidor de aplicaciones: Apache Tomcat 7.0

Apache Tomcat 7.0 es un contenedor web basado en el lenguaje Java que actúa como motor de servlets y JSPs desarrollado bajo el proyecto Jakarta en la Apache Software Foundation. Tomcat implementa las especificaciones de los servlets y de *Java Server Pages* (JSP) de Oracle *Corporatio*n. Es desarrollado en un entorno abierto y participatorio, bajo la licencia de Apache *Software License*. Apache Tomcat es empleado en un gran número de aplicaciones como, CiteSeerX y JBoss. (29)

Se decide utilizar Apache Tomcat en su versión 7, pues el mismo posee una licencia completamente libre, facilitando la obtención de nuevas versiones. Sin embargo, el licenciamiento dual que posee su principal competidor (*Glassfish*) condiciona el uso de la versión libre, la cual puede ser cerrada en cualquier momento (provocando la perdida de soporte y actualización del servidor web).

Entorno de desarrollo integrado (IDE): Intellij IDEA 14.1.1

IntelliJ IDEA es un IDE para Java, que también comprende y proporciona asistencia de codificación inteligente para una gran variedad de otros lenguajes como SQL, JPQL, HTML, JavaScript, etc., incluso cuando se inyecta la expresión del lenguaje en una cadena literal en su código Java.

Entre las características que posee se encuentran: (30)

- ✓ Inteligencia profunda: después de IntelliJ IDEA está indexado su código fuente, ofrece rápida experiencia inteligente, dando sugerencias pertinentes en todos los contextos: la finalización de código instantánea, sobre la marcha de análisis de código y herramientas de refactorización fiables.

- ✓ Finalización de código inteligente: mientras que la conclusión básica sugiere nombres de las clases, métodos, campos y palabras clave dentro del alcance de visibilidad, la terminación inteligente sugiere sólo aquellos tipos que se esperan en el contexto actual.

- ✓ Impulsores de productividad: automatiza las tareas repetitivas y tediosas de desarrollo para que el desarrollador se centre en las tareas más complejas.

- ✓ Inteligencia discreta: la asistencia de codificación en IntelliJ IDEA ayuda a mantener la productividad, permitiendo, por ejemplo: rellenar un campo buscando a través de una lista de elementos; acceder a una ventana de la herramienta; cambiar configuración, etc.

Este IDE fue seleccionado porque presenta una versión libre (Community Edition), además de que soporta el lenguaje seleccionado Groovy.

1.5 Conclusiones del capítulo

En este capítulo se analizaron los principales aspectos teóricos asociados con el proceso planificación de la Guardia Obrera Estudiantil. Se realizó el análisis de los principales términos sobre la gestión y planificación de la guardia, favoreciendo la comprensión de los términos y su conceptualización. El estudio de sistemas de planificación de guardias, tanto nacionales como internacionales, permitió determinar características semejantes que pueden ser utilizadas para el sistema que se desea desarrollar. Además se definieron las tecnologías, herramientas y lenguaje de programación, así como la metodología, permitiendo conocer las principales características y las ventajas que estas ofrecen.

Capítulo 2: Características del sistema

Introducción al capítulo

En el presente capítulo se describe la modelación del negocio mediante procesos con el objetivo de comprender cómo se realizan las actividades de la guardia. Se presenta la propuesta de solución a la problemática planteada, a partir de la cual se definen los requisitos funcionales y no funcionales del sistema. Además, se una muestran una breve descripción de cada uno de los Casos de Uso identificados para una mejor comprensión.

2.1 Descripción de los procesos

Para comprender los procesos que se manejan en la UCI acerca de la planificación y control de la guardia obrera se muestra a continuación el flujo de los procesos actuales.

Planificación de guardia

El proceso de planificación de la guardia está dividido en dos momentos fundamentales: la organización y la planificación. Para la creación de la guardia, el planificador central designa, por facultades, las postas que se deben cubrir y se definen los horarios de su realización. Cuando son conocidos estos detalles el personal encargado de la planificación de la guardia en las facultades procede a su organización y planificación. Primeramente se obtiene el listado de todos los estudiantes y trabajadores de la facultad, que es enviado por el Vicedecano de Extensión y Residencia y el Vicedecano de Administración. Luego se identifica el personal incapacitado para realizar la guardia. Se considera personal incapacitado para realizar la guardia las mujeres embarazadas, mujeres con niños pequeños, impedidos físicos y personas que posean alguna enfermedad o situación excepcional que impida la realización de guardias. Posteriormente se procede a la creación de los equipos de guardia.

Una vez creados los equipos, conociendo las postas asignadas a la facultad y los horarios de guardia, se planifica la realización de la misma.

Para la planificación se necesitan algunos datos importantes como por ejemplo la determinación de los días hábiles y no hábiles. Los días hábiles son los que normalmente se realiza la guardia. Un día no hábil es en el que se planifica la guardia de manera excepcional, como por ejemplo el 1ero de mayo. En estos días se planifica guardia a las personas que por algún motivo justificado no pueden ejercer la tarea encomendada por la dirección de la universidad, solo los que su

condición física o enfermedad lo permita. Se excluyen los impedidos físicos y las mujeres embarazadas. Si la guardia es para los trabajadores, el máximo responsable es el Vicedecano de Administración. Los trabajadores realizan la guardia en las facultades o en las áreas administrativas a la que pertenecen. En el caso de que los estudiantes, se debe conocer el grupo al cual pertenece y su género, pues esta se organiza teniendo en cuenta estos elementos. El máximo responsable de la planificación y control de la guardia estudiantil es el Vicedecano de extensión y residencia y la guardia se realiza en la residencia.

Para planificar la guardia, también se debe tener en cuenta que los equipos deben alternar entre días de semana y fines de semana. Teniendo en cuenta todas las restricciones queda conformado el listado de la guardia, donde se especifica por cada equipo el día, el lugar y el horario en que debe realizarla. Esta planificación es enviada por correo electrónico a los estudiantes y trabajadores implicados en la guardia.

Realización de la guardia

El proceso comienza a partir de que el personal de guardia recibe la notificación de planificación de la guardia vía correo. A partir de este momento puede por motivos justificados no estar de acuerdo con la planificación. Si no está de acuerdo, puede solicitar una permuta con otro compañero que acceda a ejercer la misma, o un cambio de guardia al planificador, informándole con antelación su situación. La permuta puede ser de equipo, día, posta u horario de guardia y dispone de más de 3 días de antelación al día de la guardia. Debe ser notificado al planificador para que realice los cambios y envíe notificación a la otra persona con la que se cambia. La solicitud de cambio de guardia se realiza en caso de que la persona no pueda realizar la guardia por un motivo de urgencia y no cuente con el tiempo o la disponivilidad de otro compañero para realizar una permuta. En este caso el Vicedecano revisa la planificación y en correspondencia con las necesidades de guardia es reubicado en algún equipo. El día de la guardia, se presenta en el horario y en la posta asignada, realiza la guardia y firma el libro para que quede evidencia de su realización.

Proceso Control de la guardia

Para realizar el control de la guardia, en los puestos de mando se firma en el libro de guardia la hora de entrada y la salida. Tomando como entrada el documento de planificación de guardia, el controlador chequea la realización de la guardia por todas las postas en cada uno de los turnos.

Al terminar la guardia realiza un parte de cumplimiento que es enviado a las facultades. El planificador revisa el parte y los libros de firma y elabora un informe con los incumplimientos de la guardia, verificando ambos documentos. En caso que no coincidan se les informa a los implicados para esclarecer la situación. Luego se envía el documento a los jefes administrativos que llevan a cabo el proceso.

2.2 Reglas del negocio

Las reglas del negocio son esenciales para los modelos de negocio y para los modelos de tecnología, y una parte separada y especifica de los mismos. Son restricciones explicitas de comportamiento y/o proporcionan soporte para la dirección de las actividades de negocio. (31)

Las reglas del negocio que se definieron para el desarrollo del sistema son:

- ✓ En un mismo día, solo realiza la guardia estudiantil una brigada.
- ✓ Un estudiante o un trabajador realiza la guardia en una sola posta el día asignado.
- ✓ Se planifica una guardia mensual a cada persona.
- ✓ Si una persona tiene guardia en un mes en el fin de semana, la próxima no debe corresponderle fin de semana.
- ✓ Las personas incapacitadas no hacen guardias, solo cuando exista una situación excepcional y su condición física lo permita.
- ✓ Los trabajadores externos solamente hacen guardia en tiempo de vacaciones, a no ser que se comprometan a hacer la guardia mensual.
- ✓ Los trabajadores internos hacen guardia en tiempo que no es de vacaciones.
- ✓ Las mujeres con niños pequeños hacen guardia los domingos por el día.

2.3 Modelo de negocio

El modelado de negocio permite comprender los problemas actuales de una organización e identificar posibles mejoras. Además permiten que los clientes y el equipo de desarrollo tengan

un entendimiento común. Mediante el entendimiento del negocio se pueden identificar los roles y responsabilidades de la organización en el modelo de negocio.

2.3.1 Actores de negocio

En la siguiente tabla se muestran los actores identificados durante la modelación del negocio.

Actores del Negocio	Descripción
Vicedecano de Extensión y Residencia	Es uno de los directivos de la Facultad que solicita informaciones y orienta actividades relativas a los procesos que se realizan en la residencia estudiantil y de profesores. Lleva el control de la planificación y control de la guardia de estudiantes.
Vicedecano de Administración	Es uno de los directivos de la Facultad maneja información. Lleva le control de la planificación y control de la guardia de los trabajadores.
Directivo	Actor genérico que representa al Vicedecano de Administración y al Vicedecano de Extensión y Residencia.

Tabla 1: Actores del negocio y su descripción

2.3.2 Trabajadores del Negocio

Un trabajador del negocio representa un rol desempeñado en las realizaciones de los Casos de Uso del Negocio (CUN), colabora con otros trabajadores, es notificado de los eventos del negocio y manipula las entidades del negocio para realizar sus responsabilidades.

En la siguiente tabla se muestran los trabajadores del negocio y una breve descripción.

Trabajadores del Negocio	Descripción
Planificador central	Se encarga de la definición de turnos y postas para cada una de las facultades.

Controlador	Es el responsable del control de la guardia de los grupos en las postas y horarios en un día determinado.
Planificador	Es el responsable de planificar en cada una de las facultades la guardia estudiantil.
Estudiante	Es la persona encargada de la realización de la guardia en la residencia.
Trabajador	Es la persona encargada de la realización de la guardia en los docentes.
Personal de guardia	Actor genérico que representa a los estudiantes y trabajadores que realizan guardia.

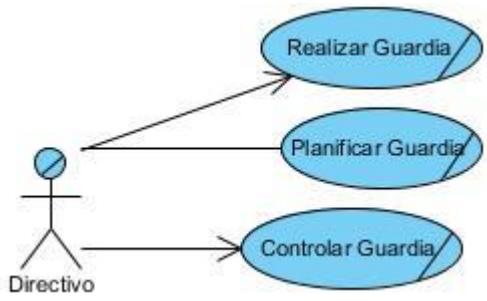
Tabla 2: Trabajadores del negocio y su descripción

2.3.3 Diagrama de Casos de Uso del Negocio (DCUN)

El Diagrama de Casos de Uso del Negocio representa un proceso como caso de uso y las relaciones con los actores del negocio.

El DCUN se muestra a continuación.

Figura 1: Diagrama de Casos de Uso del Negocio



2.3.4 Descripción de los CUN

Para un mejor entendimiento de los CUN mostrados en el DCUN se muestra una breve descripción de cada uno de ellos.

Caso de Uso: Planificar Guardia	
Actor(es)	Directivo (Vicedecano de Administración y Vicedecano de Extensión y Residencia)
Trabajador(es)	Planificador central, planificador
Resumen	El CU inicia cuando los directivos solicitan la asignación de postas y horarios al planificador central. Se envía un listado con todos los estudiantes y trabajadores al planificador (de estudiantes y trabajadores), para que se creen los turnos, postas y designen horario a cada uno de los grupos. El CU termina cuando se envía el listado de guardia a las personas involucradas.

Tabla 3: Descripción del CUN Planificar Guardia

Caso de Uso: Realizar Guardia	
Actor(es)	Directivo (Vicedecano de Administración y Vicedecano de Extensión y Residencia)
Trabajador(es)	Trabajador y estudiante
Resumen	El CU inicia cuando los directivos envían la planificación de la guardia. Los estudiantes y trabajadores revisan el listado de guardia. Si no están de acuerdo pueden solicitar una permuta entre compañeros. El día de la guardia cada persona firma el libro de guardia como constancia de su realización. Termina así el CU.

Tabla 4: Descripción del CUN Realizar Guardia

Caso de Uso: Controlar Guardia	
Actor(es)	Directivo (Vicedecano de Administración y Vicedecano de Extensión y Residencia).
Trabajador(es)	Controlador

Resumen	El CU inicia cuando los directivos envían la planificación diaria de guardia al controlador. Este la revisa y chequea la realización de la misma, emitiendo un reporte de cumplimiento. El Planificador realiza un informe de cumplimiento verificando el reporte enviado por el controlador y el libro de firma y lo envía a los superiores o archiva. Termina así el CU.

Tabla 5: Descripción del CUN Controlar Guardia

2.3.5 Diagrama de actividades

Los Casos de Uso del negocio describen las secuencias de actividades que son observables para el actor del negocio. La descripción de los CUN identificados se refleja en los siguientes diagramas de actividades. Las actividades señaladas en naranja representan que serán automatizadas en el sistema.

Figura 2: Diagrama de actividades del CUN Planificar Guardia

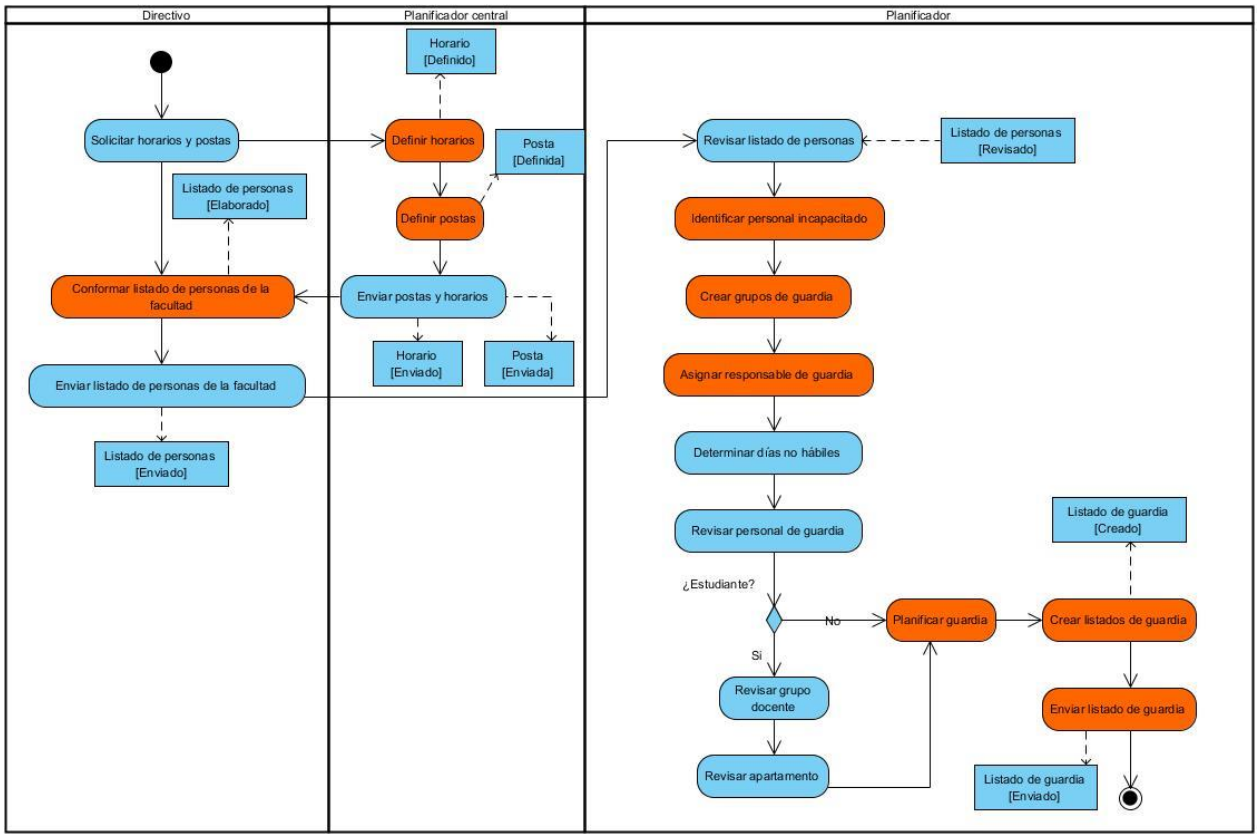


Figura 3: Diagrama de actividades del CUN Realizar Guardia

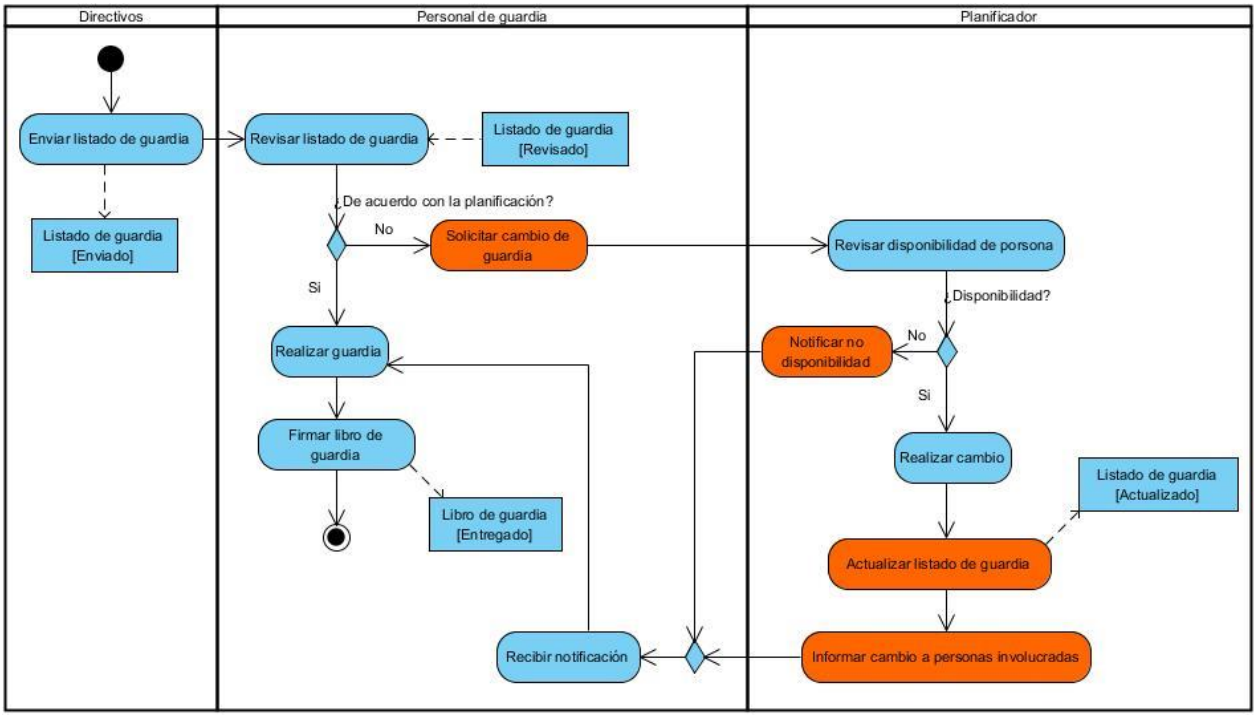
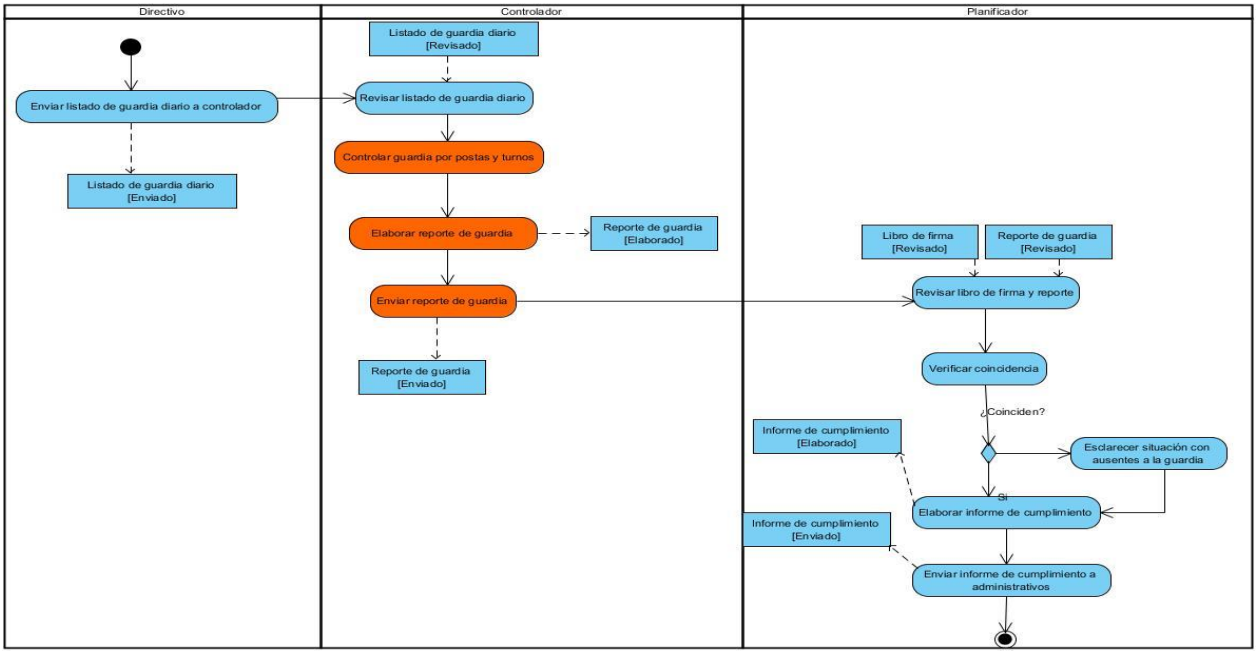


Figura 4: Diagrama de actividades del CUN Controlar Guardia



2.4 Descripción de la propuesta de solución

Para dar solución a la problemática planteada en el capítulo anterior se decide desarrollar una aplicación web. Esta aplicación contempla la planificación de la guardia en las diferentes facultades de la universidad y el control por parte de los administrativos correspondientes a cada una de ellas. Para una mejor administración de la guardia se permite la creación de postas, turnos y equipos.

Para acceder al sistema cada usuario debe autenticarse con su usuario y contraseña de la UCI, la cual será verificada contra el servicio LDAP[1] de la institución. Se asignarán roles a los usuarios y los permisos estarán asignados a los roles. La visibilidad de las funcionalidades en el sistema estará en dependencia de los permisos de cada usuario.

Los estudiantes y trabajadores podrán consultar en cualquier momento la planificación de la guardia y podrán realizar permutas entre ellos. Estos también podrán solicitar al planificador cambios de guardia, en caso de que se presentara alguna situación extraordinaria.

El sistema permitirá además la obtención de reportes sobre la realización de la guardia. En este caso se notificarán las ausencias al personal encargado del control de la guardia en las facultades. Se obtendrán además reportes que permitan establecer criterios para evaluar el cumplimiento de la guardia en las diferentes facultades de la universidad como por ejemplo el cumplimiento de guardia por facultades, cumplimiento de guardia por grupos de una facultad.

2.5 Requisitos del sistema

Los requisitos del sistema son una parte imprescindible para lograr la comunicación entre los clientes y el equipo de desarrollo. Una buena definición de los requisitos ayuda, en gran medida, a la obtención de productos de calidad y que estén en correspondencia de las necesidades del cliente. Los requisitos del sistema se clasifican en funcionales y no funcionales.

[1] LDAP: Por sus siglas en inglés (***L**ightweight **D**irectory **A**ccess **P**rotocol)* en español Protocolo Ligero/Simplificado de Acceso a Directorios. Protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.

2.5.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir. Son considerados característica requerida del sistema que expresa una capacidad de acción del mismo, una funcionalidad; generalmente expresada en una declaración en forma verbal.
Son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. (32)

En la tabla siguiente se muestran los requisitos funcionales del sistema y una breve descripción de los mismos.

Requerimientos	Descripción
RF1. Autenticar Usuario	Permite la autenticación de cada uno de los usuarios para acceder al sistema
RF2. Administrar sistema	Permite agregar un usuario, darle o quitarle privilegios, eliminarlo. Se permitirá crear y eliminar turnos, postas, áreas y módulos en caso de ser necesario.
RF3. Gestionar Parámetros de Conexión LDAP.	Permite la gestión de los parámetros de conexión al servicio LDAP (url, nombres de dominio, usuario y contraseña).
RF4. Gestionar Estudiante **RF4.1. Crear Estudiante** **RF4.2. Actualizar Estudiante** **RF4.3. Eliminar Estudiante** **RF4.4. Consultar Estudiante**	Permite la inserción de los estudiantes. Incluye su creación, actualización y eliminación.

RF5. Gestionar Trabajador **RF5.1. Crear Trabajador** **RF5.2. Actualizar Trabajador** **RF5.3. Eliminar Trabajador** **RF5.4. Consultar Trabajador**	Permite la inserción de los trabajadores. Incluye su creación, actualización y eliminación.
RF6. Reportar Guardia	Permite llevar el control de la realización de la guardia.
RF7. Exportar Reporte a PDF	Permite exportar los reportes a formato PDF.
RF8. Realizar Búsqueda de usuario	Permite realizar búsquedas de usuarios, mostrando el perfil de un usuario y el histórico de guardias planificadas.
RF9. Generar Estadísticas	Permite generar estadísticas sobre el cumplimiento de guardia por facultades, cumplimiento de guardia por grupos de una facultad.
RF10. Efectuar Cambio de Guardia	Permite efectuar el cambio de guardia.
RF11. Gestionar Turno de Guardia. **RF11.1. Crear Turno de Guardia** **RF11.2. Actualizar Turno de Guardia** **RF11.3. Eliminar Turno de Guardia** **RF11.4. Consultar Turno de guardia**	Permite la creación de los turnos de guardia según los horarios establecidos. Incluye su creación, actualización y eliminación.
RF12. Gestionar Posta de Guardia. **RF12.1. Crear Posta de Guardia** **RF12.2. Actualizar Posta de Guardia** **RF12.3. Eliminar Posta de Guardia** **RF12.4. Consultar Posta de guardia**	Permite la gestión de las Postas de Guardias. Incluye su creación, actualización y eliminación.

RF13. Gestionar Nomenclador Motivo **RF13.1. Crear Nomenclador Motivo** **RF13.2. Actualizar Nomenclador Motivo** **RF13.3. Eliminar Nomenclador Motivo** **RF13.4. Consultar Nomenclador Motivo**	Permite la gestión de los motivos que se manejan en el sistema. Incluye su creación, actualización y eliminación.
RF14. Solicitar Permuta	Permite solicitar permutas de guardia entre los usuarios.
RF15. Gestionar Facultad **RF15.1. Crear Facultad** **RF15.2. Actualizar Facultad** **RF15.3. Eliminar Facultad** **RF15.4. Consultar Facultad**	Permite la gestión de las facultades. Incluye su creación, actualización y eliminación.
RF16. Notificar planificación de la guardia	Permite que se le notifique al usuario la planificación de la guardia.
RF17. Realizar control de la guardia	Permite realizar el control de guardia. Para ellos se especifica en el sistema, por día, el cumplimiento de la guardia.
RF18. Planificar Guardia	Permite planificar la guardia.
RF19. Gestionar Guardia **RF19.1. Crear Guardia** **RF19.2. Actualizar Guardia** **RF19.3. Eliminar Guardia** **RF19.4. Consultar Guardia**	Permite la gestión de la guardia. Incluye su creación, actualización y eliminación.
RF20. Gestionar Grupo **RF20.1. Crear Grupo** **RF20.2. Actualizar Grupo**	Permite la gestión de los grupos. Incluye su creación, actualización y eliminación.

RF20.3. Eliminar Grupo **RF20.4. Consultar Grupo**	

Tabla 6: Requisitos funcionales del Sistema

2.5.2 Requisitos no funcionales

"Los requisitos no funcionales rara vez se asocian con características particulares del sistema. Más bien, estos requerimientos especifican o restringen las propiedades o cualidades emergentes del sistema (…)". (32)

Para el desarrollo de la actual investigación se definen los siguientes requerimientos no funcionales:

Prefijo	Requerimiento	Descripción
Usabilidad	RNU 1. Diseño centrado en usuario	Para la usabilidad se tuvieron en cuenta la norma ISO/IEC ISO 13407:1999 [2]. Diseño de páginas sin mucho contenido ni imágenes. Se tienen en cuenta los posibles errores que se pudiera cometer durante el llenado de los datos, mostrando mensajes sugerentes para orientar al usuario.
Usabilidad	RNU 2. Utilizar patrón de navegación	El sistema permitirá el fácil acceso a las funcionalidades del mismo. Además, se mostrarán accesos directos a las funcionalidades que más se utilicen, haciendo uso de un menú local y uno global en el sistema.
Disponibilidad	RND 1. Disponibilidad	El sistema contribuye a la planificación y control de la guardia obrera, proceso de vital

[2] ISO 13407:1999 proporciona una guía para alcanzar la calidad en el uso mediante la incorporación de actividades de naturaleza iterativa involucradas en el Diseño Centrado en el Usuario (DCU).

		importancia dentro de la universidad, por lo cual debe estar disponible las veinticuatro horas del día.
Eficiencia	RNE 1. Capacidad	El sistema debe soportar 500 usuarios conectados concurrentemente.
Diseño e Implementación	RNDI 1. Lenguajes de programación	Se utiliza para la construcción del sistema los lenguajes de programación Groovy, HTML, JavaScript y las herramientas que se utilizan son de distribución bajo licencias libres. La estructura actual del sistema permite el desarrollo de nuevos módulos o subsistemas en caso de que se deseen agregar nuevas funcionalidades.
Diseño e Implementación	RNDI 2. Forma de acceso	Al sistema se puede acceder a través de un navegador web desde los sistemas operativos Windows y GNU/Linux.
Interfaz	RNI 1. Interfaces de usuario	Para acceder al sistema debe usarse una versión del navegador Mozilla Firefox v 35.0. No se garantiza la correcta visualización en otros navegadores. Además, que para acceder al sistema el navegador debe tener habilitado el soporte para JavaScript.
Seguridad	RNS 1. Protección de los datos almacenados	El sistema debe garantizar la protección de la información almacenada. Es por ello que se establecen diferentes roles a los cuales se asignan permisos. De esta forma se garantiza

		que cada usuario acceda solamente a las funcionalidades permitidas.
Consumo de recursos	RNCR 1. Requisitos de Software de la PC cliente	Pueden ser ejecutadas desde cualquier plataforma, con el único requisito de contar con un navegador que soporte los estándares definidos en el W3C.
Consumo de recursos	RNCR 2. Requisitos de Software de la PC servidor	El sistema debe integrarse con el Gestor de base de datos PostgreSQL 8.4 y el servidor web Apache Tomcat 7.0.
Consumo de recursos	RNCR 3. Requisitos de Hardware de la PC cliente	Para uso del sistema se requerirán máquinas con procesador de velocidad igual o mayor a 2.3 GHz de micro, tarjeta de red a 100 Mb/s o equivalente y 512 Mb de RAM.
Consumo de recursos	RNCR 4. Requisitos de Hardware de la PC servidor	El hardware donde se instalará el sistema debe poseer al menos una Interfaz de red cuya velocidad de transferencia iguale o supere los 100 Mbps. El Servidor donde se instale el sistema debe tener como mínimo un procesador con velocidad igual o mayor a 3.0 GHz, 4 GB de memoria RAM y un espacio libre en Disco Duro de 40 GB.

Tabla 7: Requisitos no funcionales

2.6 Actores del sistema

Un actor es toda entidad externa al sistema que guarda una relación con el mismo y que le demanda una funcionalidad. Esto incluye a los operadores humanos, los sistemas externos y entidades abstractas como el tiempo. Un actor representa un rol en el sistema, no un usuario en específico del mismo. (25)

Los actores del sistema identificados se representan en la siguiente tabla:

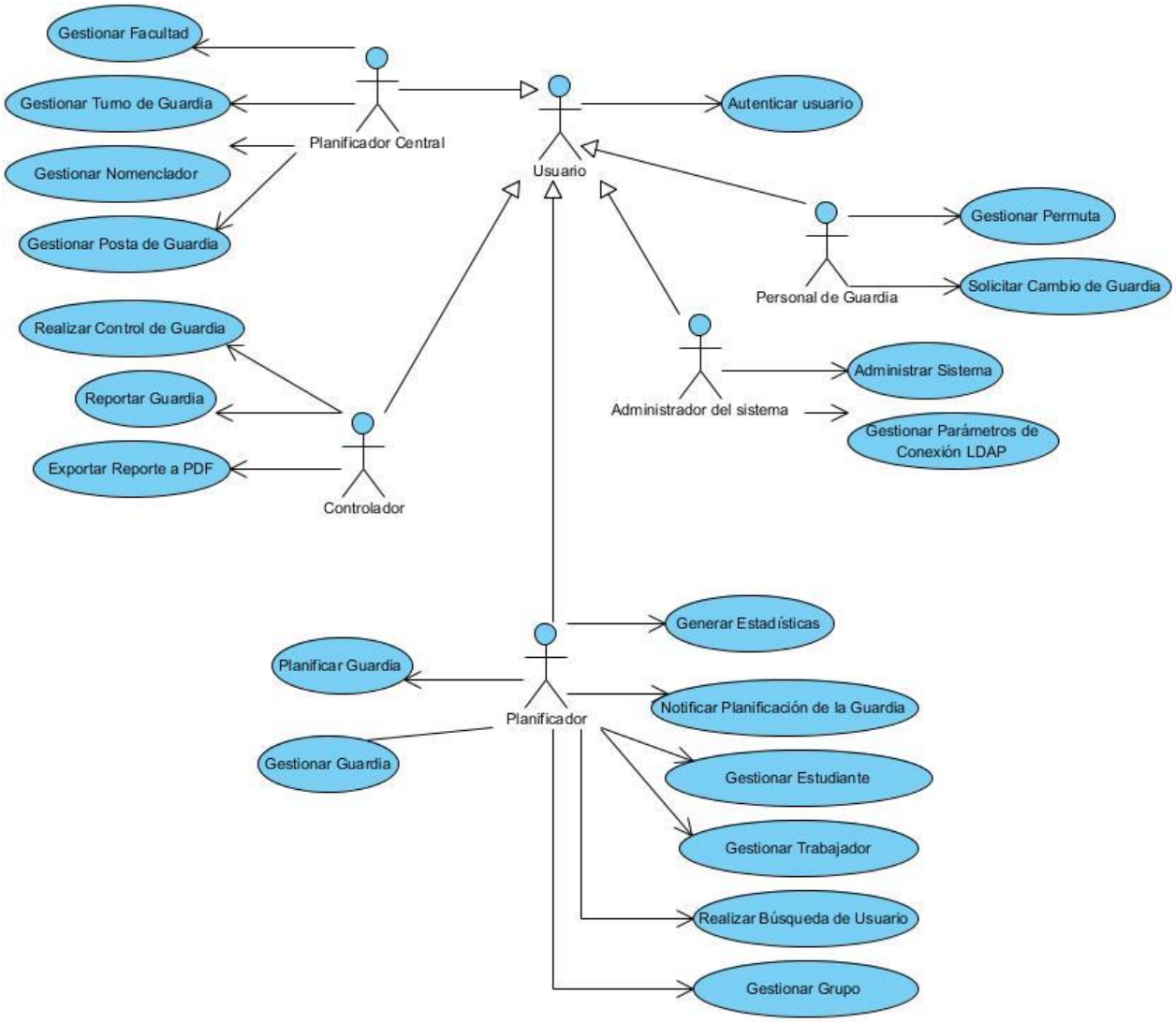
Actores del Sistema	Descripción
Planificador central	Asigna permiso de planificador a los usuarios, gestiona usuarios, los turnos, las postas, los motivos y las facultades.
Controlador	Es el responsable del control de la guardia de los grupos en las postas y horarios en un día determinado.
Planificador	Es el encargado de registrar los estudiantes y trabajadores en el sistema. Planifica la guardia en el área que le corresponde, realiza cambios y la notifica.
Usuario	Actor genérico que representa a todos los usuarios del sistema.
Administrador del sistema	Es el encargado de administrar el sistema y de la gestión de los parámetros para la conexión al LDAP.
Personal de guardia	Actor genérico que incluye a los estudiantes y trabajadores.

Tabla 8: Actores del sistema

2.7 Diagrama de Casos de Uso del Sistema (DCUS)

Luego de definir los requisitos funcionales (a partir de los cuales se obtuvieron los Casos de Uso del Sistema (CUS)) y los actores del sistema se procede a la modelación del DCUS, que se representa a continuación.

Figura 5: Diagrama de Casos de Uso del Sistema



2.7.1 Descripción de los CUS

Las tablas que se muestran a continuación representan la descripción de los CUS identificados.

CU Autenticar usuario	
Objetivo	Identificar las personas que acceden al sistema.
Actores	Usuarios del sistema

Resumen	El CU inicia cuando el actor introduce su usuario y contraseña. El sistema verifica los datos introducidos. En dependencia de su rol, se le muestra las interfaces según los permisos que tenga asignados. El usuario accede al sistema y termina el CU.
Complejidad	Baja
Prioridad	Media
Precondiciones	El usuario que intenta acceder al sistema debe estar registrado como usuario del sistema.
Post-condiciones	El usuario accede al sistema.
Referencia	RF 1

Tabla 9: Descripción del CUS Autenticar usuario

CU Administrar sistema	
Objetivo	Agregar un usuario, darle o quitarle privilegios, puede también eliminar un usuario determinado. Además se permite crear y eliminar turnos, postas, áreas y módulos en caso de ser necesario.
Actores	Administrador del sistema.
Resumen	El CU inicia cuando el actor decide realizar cualquier tarea de administración del sistema e introduce los datos para ejecutar cada una de ellas. El sistema verifica los datos introducidos. Se realizan las tareas de administración y termina el CU.
Complejidad	Media
Prioridad	Media

Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de administrador.
Post-condiciones	Se realizan las tareas de administración.
Referencia	RF 2

Tabla 10: Descripción del CUS Administrar sistema

CU Gestionar Parámetros de Conexión LDAP	
Objetivo	Configurar los parámetros necesarios para la conexión con el LDAP de la UCI.
Actores	Administrador del sistema.
Resumen	El CU inicia cuando el actor decide gestionar los parámetros para realizar la conexión al LDAP de la UCI. Una vez definido los parámetros puede consultarlos, modificarlos o eliminarlos. El sistema verifica los datos introducidos. Se realizan las tareas sobre los parámetros y termina el CU.
Complejidad	Alta
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de administrador.
Post-condiciones	Se realizan las tareas de administración.
Referencia	RF 3

Tabla 11: Descripción del CUS Gestionar Parámetros de Conexión LDAP

CU Gestionar Estudiante

Objetivo	Crear, modificar, eliminar y consultar un estudiante.
Actores	Planificador
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar un estudiante. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos del estudiante (según selección). Termina el CU.
Complejidad	Media
Prioridad en Negocio	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador. Para poder modificar o eliminar un estudiante debe haber al menos un estudiante registrado previamente.
Post-condiciones	Se crea, modifica, elimina o consulta el estudiante.
Referencia	RF 4

Tabla 12: Descripción del CUS Gestionar Estudiante

CU Gestionar Trabajador	
Objetivo	Crear, modificar, eliminar y consultar un trabajador.
Actores	Planificador

Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar un trabajador. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos del trabajador (según selección). Termina el CU.
Complejidad	Media
Prioridad en Negocio	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador. Para poder modificar o eliminar un trabajador debe haber al menos un trabajador registrado previamente.
Post-condiciones	Se crea, modifica, elimina o consulta el trabajador.
Referencia	RF 5

Tabla 13: Descripción del CUS Gestionar Trabajador

CU Reportar Guardia	
Objetivo	Reportar el cumplimiento de la guardia.
Actores	Controlador
Resumen	El CU inicia cuando el actor decide realizar el reporte de la guardia. El sistema valida los datos introducidos. Se guardan los datos ingresados y termina el CU.
Complejidad	Baja

Prioridad	Alta
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de controlador.
Post-condiciones	Se crea el reporte de cumplimiento de la guardia.
Referencia	RF 6

Tabla 14: Descripción del CUS Ingresar Reportar Guardia

CU Exportar Reportes a PDF	
Objetivo	Mostrar en formato PDF los datos del reporte.
Actores	Controlador
Resumen	El CU inicia cuando el actor, una vez realizado el reporte de la guardia, decide exportarlos al formato PDF. Se muestra el reporte en el formato indicado y termina el CU.
Complejidad	Media
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de controlador.
Post-condiciones	Se muestra el reporte de la guardia en formato PDF.
Referencia	RF 7

Tabla 15: Descripción del CUS Exportar Reportes a PDF

CU Realizar Búsqueda de Usuario	
Objetivo	Buscar un determinado usuario en el sistema.

Actores	Planificador (de estudiantes y trabajadores)
Resumen	El CU inicia cuando el actor decide realizar la búsqueda de un usuario. Para ello introduce los criterios de búsqueda (nombre completo, correo y usuario). El Sistema muestra el perfil del usuario que coincide con los criterios de búsqueda introducidos y el histórico de guardias planificadas. Termina el CU.
Complejidad	Baja
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador.
Post-condiciones	Se realiza la búsqueda por el criterio introducido.
Referencia	RF 8

Tabla 16: Descripción del CUS Realizar búsquedas de Usuario

CU Generar Estadísticas	
Objetivo	Mostrar las estadísticas del sistema.
Actores	Planificador (de estudiantes y trabajadores)
Resumen	El CU inicia cuando el actor decide obtener estadísticas del sistema y selecciona el tipo de estadística. El sistema genera estadísticas, ya sea sobre el cumplimiento de guardia por facultades como el cumplimiento de guardia por grupos de una facultad. Termina el CU.
Complejidad	Media

Prioridad	Alta
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador.
Post-condiciones	Se muestran las estadísticas especificadas.
Referencia	RF 9

Tabla 17: Descripción del CUS Generar Estadísticas

CU Realizar Cambio de Guardia	
Objetivo	Realizar un cambio de guardia.
Actores	Planificador.
Resumen	El CU inicia cuando el actor decide solicitar un cambio en la planificación de la guardia. El sistema realiza el cambio solicitado. Termina el CU.
Complejidad	Baja
Prioridad en Negocio	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador. Debe ser realizado antes de notificar la guardia.
Post-condiciones	Se efectúa el cambio.
Referencia	RF 10

Tabla 18: Descripción del CUS Solicitar Cambio de Guardia

CU Gestionar Turno de Guardia

Objetivo	Crear, modificar, eliminar y consultar un turno de guardia.
Actores	Planificador central
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar un turno de guardia. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos del turno (según selección). Termina el CU.
Complejidad	Media
Prioridad en Negocio	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador central. Para poder modificar o eliminar un turno debe haber al menos un tuno registrado previamente.
Postcondiciones	Se crea, modifica, elimina o consulta el turno de guardia.
Referencia	RF 11

Tabla 19: Descripción del CUS Gestionar Turno de Guardia

CU Gestionar Posta de Guardia	
Objetivo	Crear, modificar, eliminar y consultar una posta de guardia.
Actores	Planificador central

Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar una posta de guardia. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos de la posta (según selección). Termina el CU.
Complejidad	Media
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador central. Deben haberse creado en el sistema las facultades. Para poder modificar o eliminar una posta debe haber al menos una posta registrada previamente.
Postcondiciones	Se crea, modifica, elimina o consulta la posta de guardia.
Referencia	RF 12

Tabla 20: Descripción del CUS Gestionar Postas de Guardia

CU Gestionar Nomenclador Motivo	
Objetivo	Crear, modificar, eliminar y consultar un motivo.
Actores	Planificador central
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar un motivo. Selecciona la opción deseada e introduce los

	datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos del motivo (según selección). Termina el CU.
Complejidad	Media
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador central. Para poder modificar o eliminar un motivo debe haber al menos un motivo registrado previamente.
Post-condiciones	Se crea, modifica, elimina o consulta un motivo.
Referencia	RF 13

Tabla 21: Descripción del CUS Gestionar Nomenclador

CU Solicitar Permuta	
Objetivo	Solicitar una permuta.
Actores	Estudiante, trabajador
Resumen	El CU inicia cuando el actor decide solicitar una permuta en la planificación de su guardia con otro compañero también planificado y de muto acuerdo. El sistema valida los datos de ambos y realiza la permuta. Termina el CU.
Complejidad	Media
Prioridad	Media

Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de estudiante o trabajador. El tiempo en que se realiza la permuta tiene que ser 3 días antes de la planificación de la guardia.
Post-condiciones	Se realiza la solicitud de permuta.
Referencia	RF 14

Tabla 22: Descripción del CUS Gestionar Permuta

CU Gestionar Facultad	
Objetivo	Crear, modificar, eliminar y consultar una facultad.
Actores	Planificador central
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar una facultad. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos de la facultad (según selección). Termina el CU.
Complejidad	Media
Prioridad	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y tener permiso de planificador central.

	Para poder modificar o eliminar una facultad debe haber al menos una facultad registrada previamente.
Post-condiciones	Se crea, modifica, elimina o consulta una facultad.
Referencia	RF 15

Tabla 23: Descripción del CUS Gestionar Facultad

CU Notificar Planificación de la Guardia	
Objetivo	Notificar al personal involucrado en la guardia la planificación de la misma.
Actores	Planificador
Resumen	El CU inicia cuando el actor, una vez realizada la planificación de guardia, decide notificarle al personal involucrado. El sistema valida los datos introducidos y notifica la planificación. Termina el CU.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador. El tiempo en que se realiza la solicitud tiene que ser 3 días antes de la planificación de la guardia.
Post-condiciones	Se notifica al personal de guardia la planificación.
Referencia	RF 16

Tabla 24: Descripción del CUS Notificar Planificación de la Guardia

CU Realizar Control de la Guardia	
Objetivo	Realizar el control por todas las postas en cada uno de los turnos, y al culminar el horario de guardia por día especificar el cumplimiento o no de la guardia.
Actores	Controlador
Resumen	El CU inicia cuando el actor realiza el control de la guardia por postas y turnos. El sistema valida los datos introducidos y los registra en el sistema. Termina el CU.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de controlador.
Post-condiciones	Se realiza el control de la guardia.
Referencia	RF 17

Tabla 25: Descripción del CUS Realizar Control de la Guardia

CU Planificar Guardia	
Objetivo	Realizar la planificación de la guardia.
Actores	Planificador
Resumen	El CU inicia cuando el actor realizar la planificación de la guardia. Selecciona la guardia a planificar. Para ello se indica el día, el horario en que cada grupo que realiza la guardia. Se especifica además la posta que

	debe cubrir cada integrante del equipo. El sistema valida los datos introducidos y crea la planificación de la guardia. Termina el CU.
Complejidad	Alta
Prioridad	Alta
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador. Se deben haber definido en el sistema las postas, los turnos y las facultades.
Post-condiciones	Se realiza la planificación de la guardia.
Referencia	RF 18

Tabla 26: Descripción del CUS Planificar Guardia

CU Gestionar Guardia	
Objetivo	Crear, modificar, eliminar y consultar una guardia.
Actores	Planificador
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar una guardia. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos de la guardia (según selección). Termina el CU.
Complejidad	Media
Prioridad	Media

Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y tener permiso de planificador. Para poder modificar o eliminar una guardia debe haber al menos una guardia registrada previamente.
Post-condiciones	Se crea, modifica, elimina o consulta una guardia.
Referencia	RF 19

Tabla 27: Descripción del CUS Gestionar Guardia

CU Gestionar Grupo	
Objetivo	Crear, modificar, eliminar y consultar un grupo.
Actores	Planificador
Resumen	El CU inicia cuando el actor decide crear, modificar, eliminar o consultar un grupo. Selecciona la opción deseada e introduce los datos de acuerdo a la operación que selecciona. El sistema crea, modifica, elimina o muestra los datos del grupo (según selección). Termina el CU.
Complejidad	Media
Prioridad en Negocio	Media
Precondiciones	El usuario que intenta acceder a esta funcionalidad debe estar registrado como usuario del sistema y poseer permisos de planificador.

	Para poder modificar o eliminar un grupo debe haber al menos un grupo previamente.
Post-condiciones	Se crea, modifica, elimina o consulta el grupo.
Referencia	RF 20

Tabla 28: Descripción del CUS Gestionar

Para determinar la complejidad y la prioridad se utilizó el documento Evaluación de CU establecidos para los proyectos de la UCI, el cual se encuentra descrito en el Anexo 1.

2.8 Conclusiones del Capítulo

En este capítulo se realizó una propuesta para dar solución a la problemática planteada. Se describieron los procesos de la guardia, agrupándolos en CUN y se realizaron los diagramas de actividades correspondientes a cada caso de uso. En los diagramas de actividades se identificaron las tareas a automatizar y las entidades utilizadas. A partir de los CUN se identificaron 20 requisitos funcionales con los que debe contar el sistema, permitiendo comprender la manera que serán presentados al usuario. También fueron descritos los requisitos no funcionales que el sistema debe tener, en aras de lograr un producto más usable y atractivo.

En correspondencia con los requisitos funcionales identificados se definieron los CU y se describieron los mismos. Estos CU fueron presentados en un DCUN. El modelo de CUN es tomado como entrada fundamental para diseñar el sistema, con el objetivo de cumplir con las funcionalidades descritas en este capítulo.

Capítulo 3: Arquitectura y diseño del sistema

Introducción

En este capítulo se propone el diseño del sistema, a partir de los Diagramas de Clases del Diseño utilizando estereotipos web, los diagramas de colaboraciones y el diseño de la base de datos. Se describen los elementos de estos diagramas, así como el patón arquitectónico y los patrones de diseño utilizados.

3.1 Modelo de diseño

El modelo de diseño constituye el conjunto de diagramas que describen el diseño lógico de un sistema. Es un modelo de objetos que describe la realización física de los Casos de Uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además sirve de abstracción de la implementación del sistema y es utilizado como una entrada fundamental de las actividades de implementación. (33)

3.1.1 Diagramas de clases del diseño utilizando estereotipos web

Los diagramas de clases del diseño especifican la estructura de clases de un sistema, así como sus relaciones.

A continuación se muestran los diagramas de clases del diseño de los CU de prioridad alta.

Figura 6: Diagrama de Clases del Diseño del CU Planificar Guardia

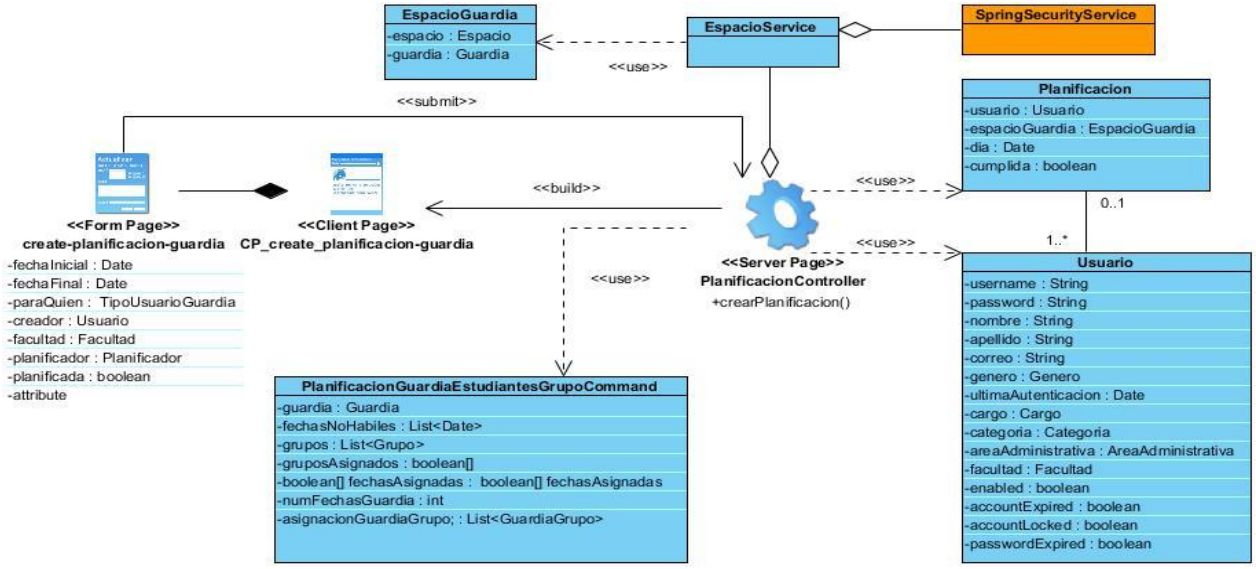


Figura 7: Diagrama de Clases del Diseño del CU Realizar Control Guardia

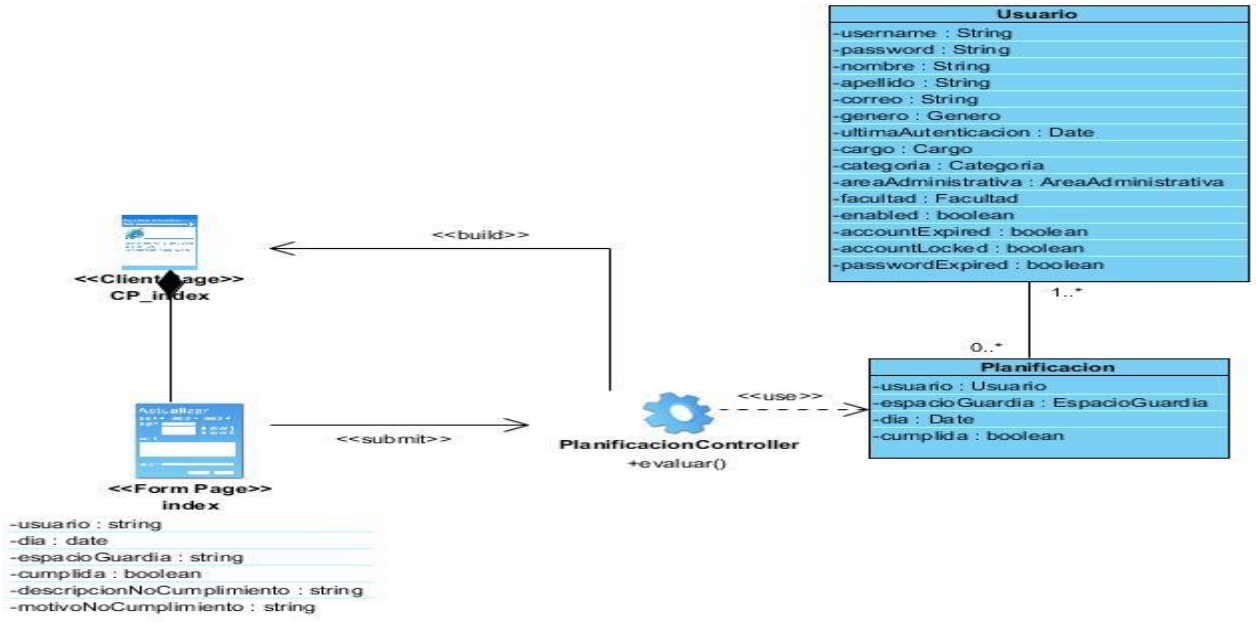
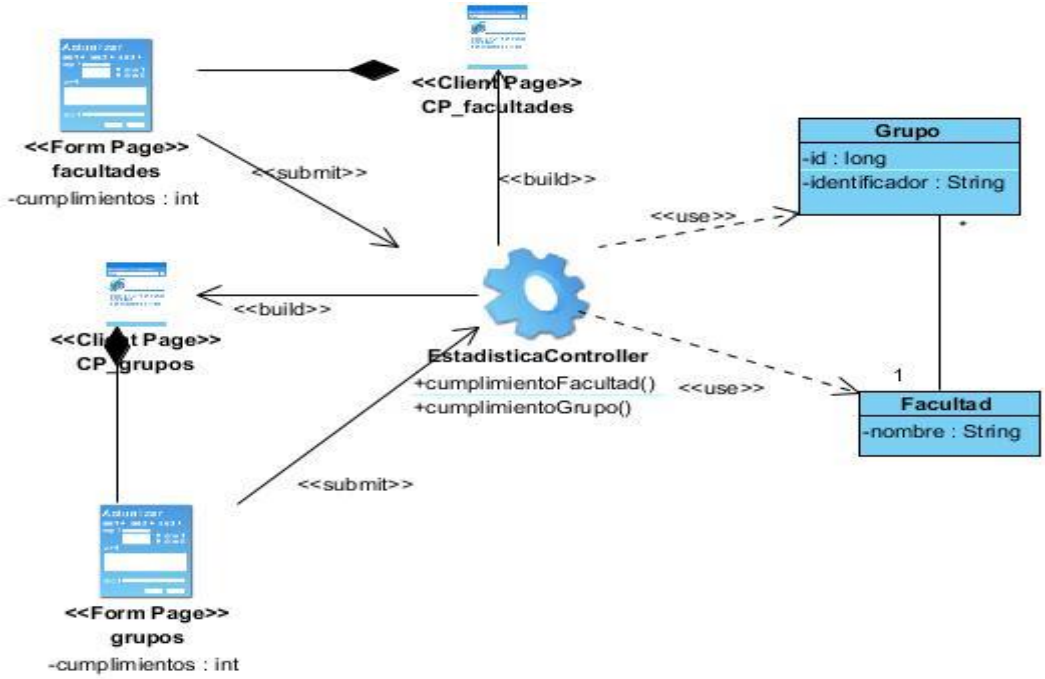


Figura 8: Diagrama de Clases del Diseño del CU Generar Estadísticas



3.1.2 Diagramas de interacción

Los diagramas de interacción explican gráficamente cómo los objetos interactúan a través de mensajes para realizar las tareas. El UML define dos tipos de estos diagramas, ambos sirven para expresar interacciones semejantes o idénticas de mensaje: diagrama de colaboración y de secuencia. (32)

- ✓ Diagrama de Colaboración: describen las interacciones entre los objetos en un formato de grafo o red.

- ✓ Diagrama de Secuencia: describe las interacciones en una especie de formato de cerca o muro.

Para representar la interacción de objetos en el sistema se seleccionan los diagramas de secuencia.

Un diagrama de secuencias muestra la interacción de un conjunto de objetos de una aplicación a través del tiempo, en el cual se indican los módulos o clases que formaran parte del programa y las llamadas que se hacen cada uno de ellos para realizar una tarea determinada. Por esta razón permite observar la perspectiva cronológica de las interacciones. Es importante recordar que el diagrama de secuencias se realiza a partir de la descripción de un caso de uso. (34)

Así como se representaron los Diagramas de clases del diseño para los CU de prioridad alta, también se muestran los diagramas de secuencia de los mismos.

Figura 9: Diagrama de secuencia del CU Planificar Guardia

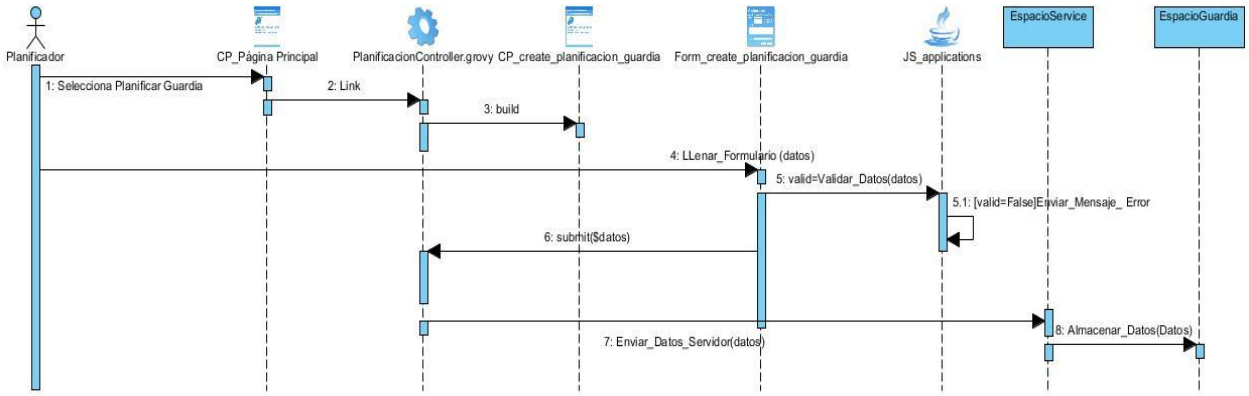


Figura 10: Diagrama de secuencia del CU Realizar Control Guardia

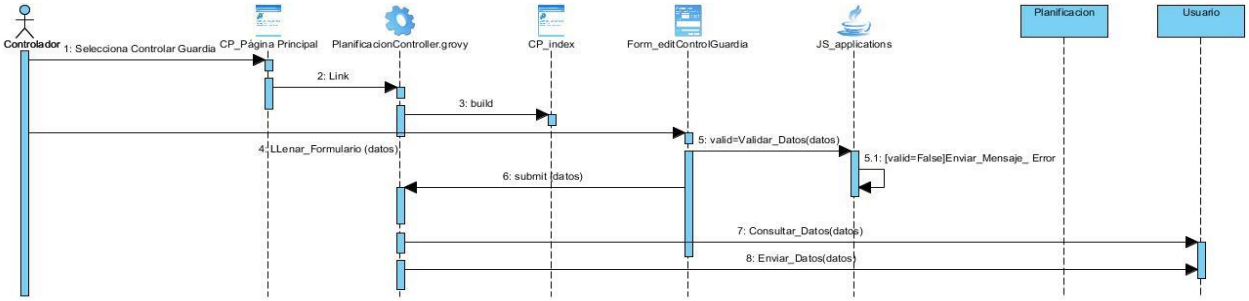
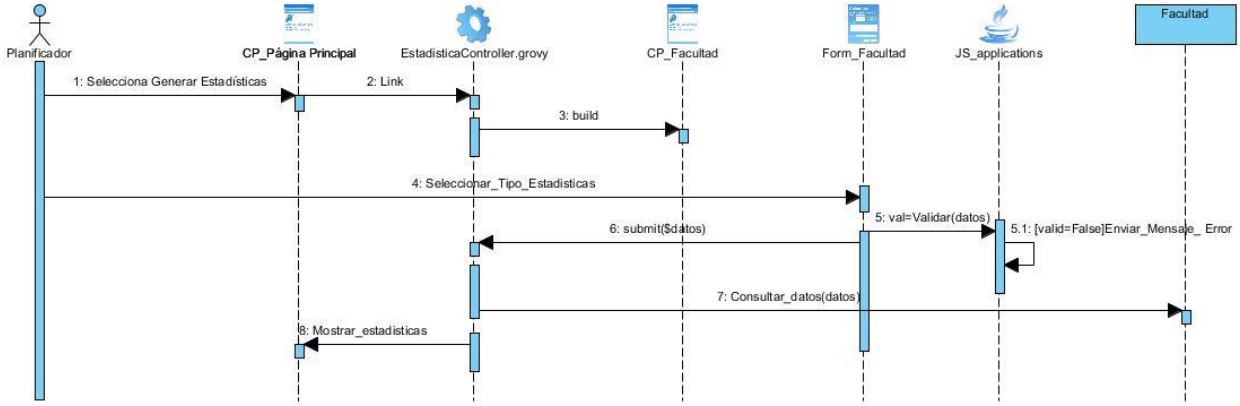


Figura 11: Diagrama de secuencia del CU Generar Estadísticas



3.2 Descripción del patrón arquitectónico

El patrón arquitectónico que se utiliza para realizar el sistema está en correspondencia con el que utiliza el *framework* Grails, que es el Modelo-Vista-Controlador (MVC). El modelo contiene todos los componentes que representan y gestionan los datos almacenados en la aplicación. Los componentes de la Vista son los encargados de mostrar al usuario el estado de los modelos de datos, así como las acciones que se pueden ejecutar. El controlador contiene los componentes que reciben órdenes del usuario, gestionan la lógica de negocio sobre el modelo de datos y deciden qué punto de vista se debe mostrar. A continuación se explica cómo se definen cada una de estas capas en Grails. (35)

- ✓ Modelo: los modelos de datos se definen mediante las llamadas "clases de dominio", denominadas Entidades. Tiene con un gestor de persistencia basado en Hibernate, llamado GORM (por sus siglas en inglés de Mapeador de Objeto Relacional de Grails), que gestiona las entidades ciclos de vida y proporciona métodos predeterminados de búsqueda. Cada operación realizada a través de una clase de dominio será traducido por Hibernate en alguna consulta SQL, necesario para reflejar la operación realizada en la base de datos.

- ✓ Vistas: las vistas se desarrollan utilizando páginas servidoras de Groovy (SGP), que es una versión simplificada de JSP (páginas servidoras de Java) que permite colocar las expresiones en el código HTML y utilizar JSTL como etiquetas. Cuando la aplicación se

está ejecutando, se puede decidir qué vista se debe procesar y enviar al cliente por medio del método "convertir" en los controladores o se puede dejar que Grails utilice una vista predeterminada.

- ✓ Controlador: los controladores son los componentes responsables de atender las solicitudes de cliente entrantes, la gestión de la ejecución de la lógica de negocio y la actualización de la vista para que el usuario la puede conocer el estado final del modelo de datos sobre la ejecución de la acción. La convención en Grails es que cualquier controlador en la carpeta "controlador" de la aplicación y con el nombre terminado en "controlador" se considerará un controlador. Grails identificará a cada solicitud HTTP, que el controlador se debe llamar en base a la definición establecida en el "conf / UrlMappings.groovy".
  La configuración por defecto se cambia mediante la definición de las direcciones URL con el formato "[Controlador] / [acción] / [id]", dónde "Controlador" es la primera parte del nombre (eliminando el sufijo "Controller"); "Acción" es el método que se ejecutará; "Id" es el identificador de una instancia de la clase de dominio.

Las principales ventajas de hacer uso del patrón MVC son (36):

- ✓ La separación del Modelo de la Vista, es decir, separar los datos de la representación visual de los mismos.
- ✓ Facilita agregar nuevos tipos de datos según sea requerido por la aplicación ya que son independientes del funcionamiento de las otras capas.
- ✓ Facilita el mantenimiento en caso de errores.
- ✓ Ofrece maneras más sencillas para probar el correcto funcionamiento del sistema.
- ✓ Permite el escalamiento de la aplicación en caso de ser requerido.

3.3 Patrones de diseño

Los patrones son simples soluciones a problemas recurrentes. Entre sus características se encuentra describir el problema de forma sencilla, el contexto en el que ocurre, puntualizar los pasos a seguir, hacer énfasis en los puntos fuertes y débiles de la solución, referir otros patrones asociados, entre otras. A continuación se aborda sobre los patrones de diseño utilizados para conformar el diseño de la solución propuesta:

Patrones GRASP

Los patrones GRASP (*General Responsabilty Assignment Software Patterns*, "Patrones de *Software* para la Asignación General de Responsabilidad") describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Constituyen un apoyo para la enseñanza que ayuda a entender el diseño de objeto esencial y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. (25) A continuación se muestran los patrones GRASP presentes en la solución:

Experto: La utilización del patrón permite asignar las responsabilidades a las clases que cuenten con la información necesaria para cumplirlas. Se evidencia en cada una de las clases del modelo del sistema, las cuales manejan la información que dominan para la realización de una tarea (Ejemplo: la clase PlanificaciónController es la encargada de la planificación de la guardia). Esto favorece el hecho de tener un sistema más robusto y de más fácil mantenimiento; también definiciones de clase sencillas y fáciles de comprender. El uso de este patrón permite que se cumpla también el Bajo Acoplamiento, se explica a continuación en que consiste.

Bajo Acoplamiento: El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con qué las conoce y con qué recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras. (25)

El Bajo Acoplamiento soporta el diseño de clases más independientes que reducen el impacto de los cambios, son más reutilizables y acrecientan la oportunidad de una mayor productividad. Las clases contenidas en el sistema fueron implementadas de forma independientes, esto mitiga el impacto de los cambios que se puedan realizar en una clase.

Alta Cohesión: El objetivo es asignar una responsabilidad de modo que la cohesión siga siendo alta. La cohesión es una medida de cuan relacionadas y enfocadas están todas las responsabilidades de una clase.

En el sistema este patrón se pone de manifiesto al definir a cada clase una función específica y única.

Controlador: El patrón controlador funciona como intermediario entre una interfaz y el algoritmo que la implementa. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, con el objetivo de aumentar la reutilización de código y a la vez tener un mayor control de las funcionalidades. Si se recurre a un diseño orientado a objetos, hay que elegir los controladores que manejen esos eventos de entrada. El encargado de dicha tarea es el patrón Controlador, y se evidencia en la propia arquitectura MVC que utiliza Grails.

Patrones GOF (***Gang of Four*** o **Grupo de los 4**)

Los patrones GOF son muy utilizados como soluciones a problemas que se pueden surgir en el desarrollo de software. Se clasifican en tres categorías basadas en su propósito: creacionales, estructurales y de comportamiento.

Inversión de Control (IoC): la Inversión de Control es otro patrón utilizado por Grails, según el cual las dependencias de un componente no deben gestionarse desde el propio componente para que este solo tenga la lógica necesaria para hacer su trabajo. El objetivo de esta técnica es mantener los componentes lo más sencillos que sea posible, incluyendo únicamente código que tenga relación con la lógica de negocio. De esta forma, la aplicación será más fácil de comprender y mantener. (37)

IoC es un estilo de programación en el cual un *framework* o librería controla el flujo de un programa. En el caso específico del sistema este patrón se evidencia con la propia funcionalidad del *framework* utilizado. Grails configura Spring para que gestione su ciclo de vida (cuándo se crea, cuántas instancias se mantienen vivas a la vez, cómo se destruyen, etc.) y sus dependencias (qué otros componentes necesita para realizar su trabajo y cómo conseguirlos).

Singleton: El patrón Singleton asegura que una clase tenga una sola instancia y proporciona un punto de acceso global a ella. Los servicios en Grails por defecto hacen uso de este patrón, de esta forma solo existe una instancia de los servicios durante la ejecución, logrando una reducción de uso de memoria y de tiempo de procesamiento (se elimina la sobrecarga de crear nuevos objetos).

3.4 Modelo de datos

El modelo de datos es un lenguaje orientado a describir una base de datos. Permite describir la representación de la información en términos de datos, sus relaciones, sus restricciones y las operaciones de manipulación de los datos. Otro enfoque es pensar que un modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan dichos elementos entre sí. No hay que perder de vista que una base de datos siempre está orientada a resolver un problema determinado, por lo que los dos enfoques propuestos son necesarios en cualquier desarrollo de *software*.

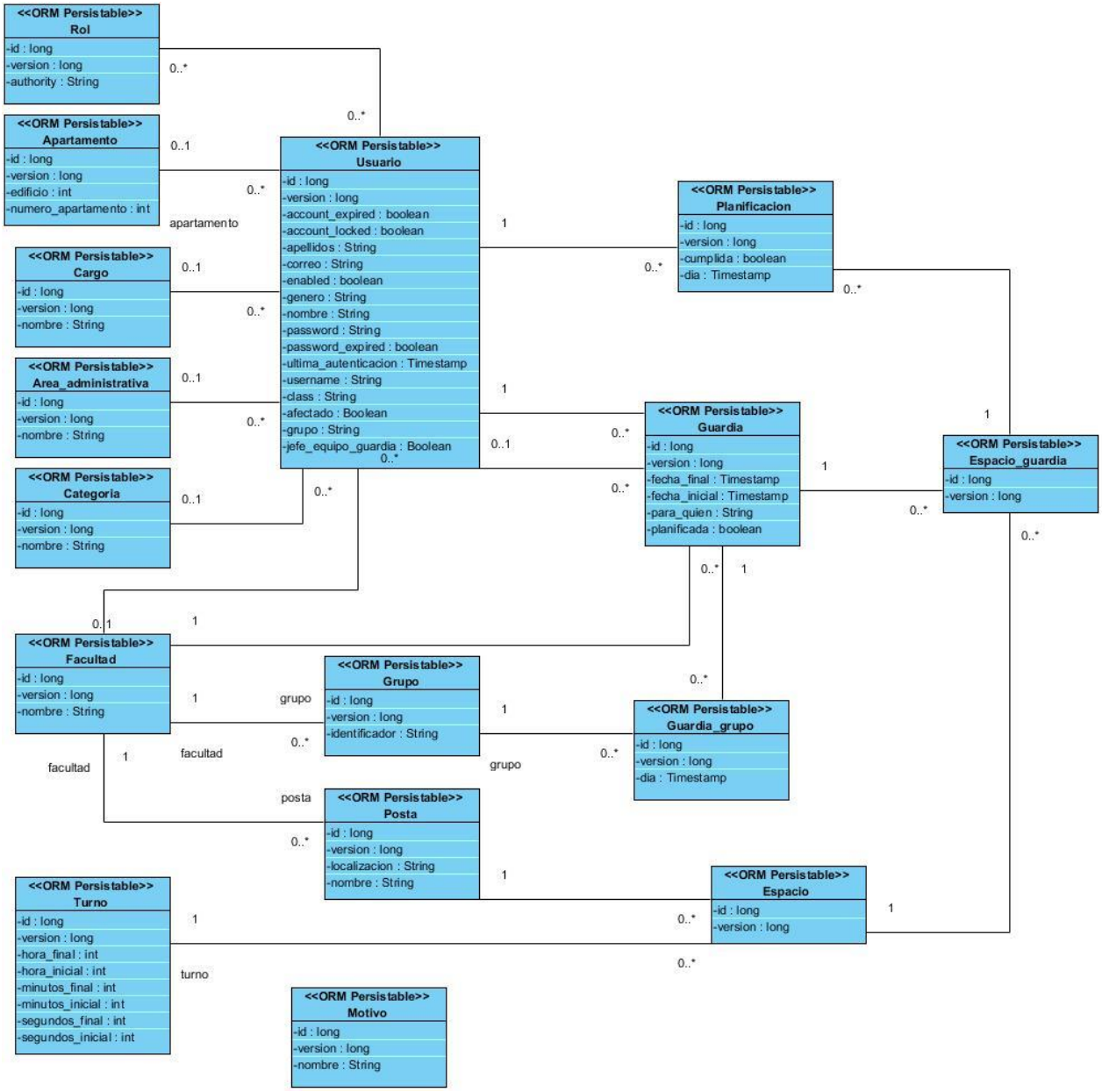
3.4.1 Modelo lógico

El diagrama del Modelo lógico de datos o Diagrama de clases persistentes, muestra las clases capaces de mantener su valor en el espacio y en el tiempo. El diagrama de clase se puede usar para modelar la estructura lógica de la base de datos, con clases representando tablas, y atributos de clase representando columnas. Las clases persistentes y sus atributos hacen referencia directamente a las entidades lógicas y a sus atributos.

Cuando se define correctamente el Modelo Lógico, se hace mucho menos engorroso llegar al Modelo de Datos o Modelo Físico.

A continuación se muestra el modelo lógico de la BD.

Figura 12: Modelo lógico de datos

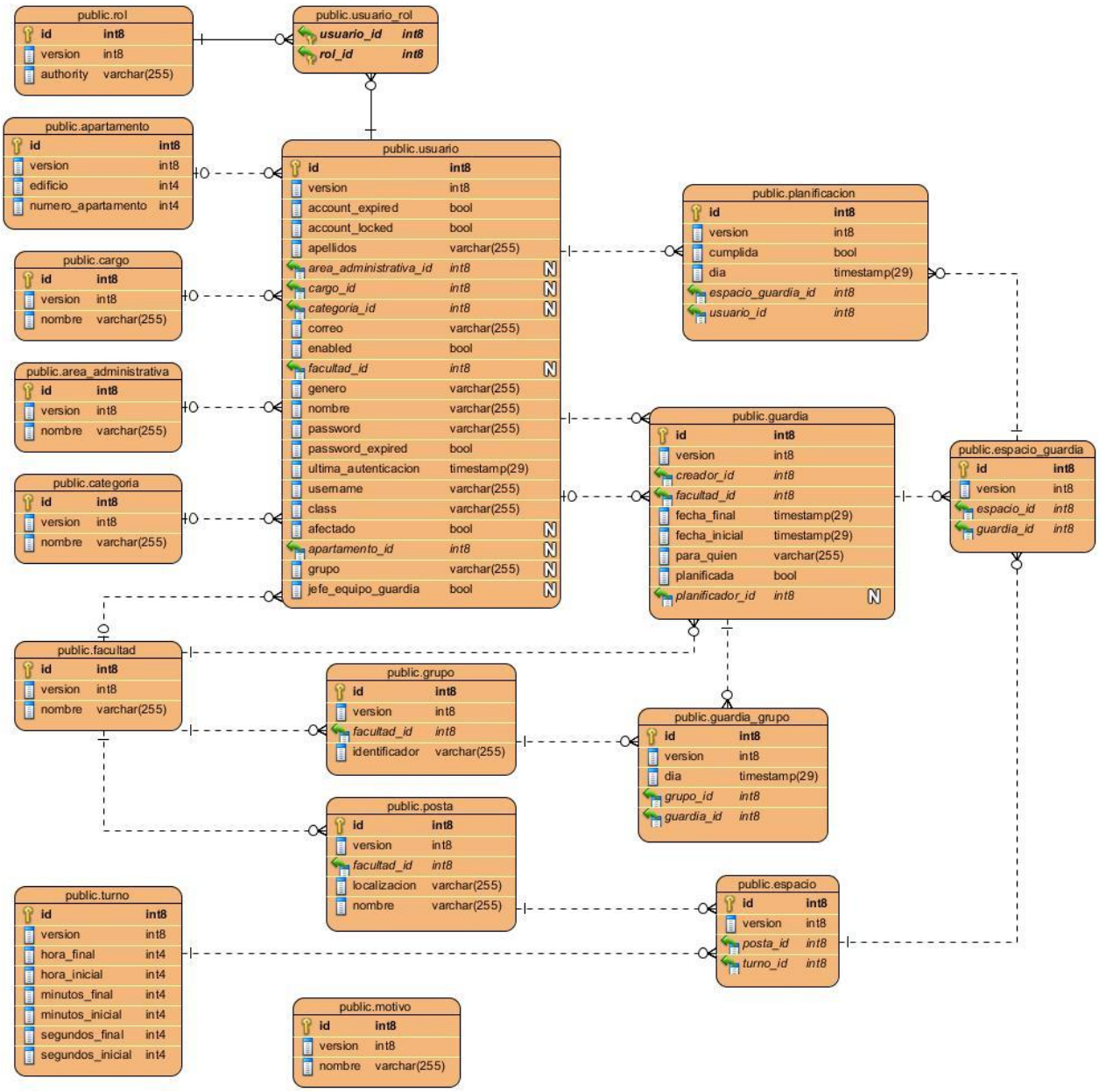


3.4.2 Modelo físico

A continuación se muestra el modelo de datos físico del sistema, el cual ofrece una descripción abstracta sobre la representación de los datos en un Sistema Gestor de Base de Datos (SGBD). Los componentes que lo conforman, son entidades existentes en el sistema. Este modelo

contiene a su vez, características propias de estos objetos y la forma en que se relacionan entre sí.

Figura 13: Modelo Físico de la Base de Datos



3.4.3 Descripción de las tablas del modelo físico

Cargo: Representa le cargo que puede tener un usuario.

Categoría: Representa la categoría de un usuario (estudiante, trabajador).

Espacio: Es el turno definido para una posta determinada.

Espacio-guardia: Es el turno de guardia definido para una posta determinada en una facultad.

Facultad: Representa a las facultades existentes en la universidad.

Grupo: Son los diferentes grupos docentes de una facultad.

Guardia: Representa al equipo de guardia en una fecha determinada y con su personal de guardia asignado.

Guardia-grupo: Representa el día en que un grupo docente tiene guardia.

Motivo: Es un nomenclador que almacena todos los motivos que se almacenan en el sistema. Ejemplo de ello es el motivo de ausencia a guardia.

Planificación: Representa la planificación de la guardia en un día, en una posta determinada y el personal de guardia asignado.

Posta: Representa las diferentes postas definidas para una facultad o un área.

Rol: Es el rol que tiene un usuario en el sistema.

Turno: Es el horario definido para la guardia.

Usuario: Representa a todas las personas que interactúan con el sistema.

Usuario-rol: Almacena el rol que un usuario determinado desempeña en el sistema.

3.5 Conclusiones del capítulo

Durante el desarrollo de este capítulo se hizo referencia y se describieron los elementos fundamentales del diseño del sistema, teniendo como guía los flujos de trabajo Análisis y Diseño que propone la metodología de desarrollo AUP. El diseño estuvo también en correspondencia con los requisitos definidos para el sistema, ajustados a la arquitectura Modelo Vista Controlador.

Fueron presentados los Diagramas de Clases del Diseño de las funcionalidades más significativas, los cuales cumplen con el patrón arquitectónico definido. En los mismos se utilizaron patrones de diseño para mejorar la flexibilidad y un mejor entendimiento para la implementación de las funcionalidades.

Otro de los resultados significativos de este capítulo es el diseño del Modelo de datos para la persistencia de la información, definiendo las entidades y sus relaciones.

Capítulo 4: Implementación y prueba

Introducción al capítulo

Este capítulo expone lo referente a los flujos de trabajo Implementación y Pruebas, los cuales son determinantes en el proceso de desarrollo de *software*. Para ello, se modela el diagrama de componentes, haciendo una representación de la implementación de las clases de diseño en términos de componente y cómo estas se organizan, de acuerdo con los nodos específicos en el modelo de despliegue. Además, se realiza un análisis de los casos de prueba, teniendo en cuenta los datos de entrada, resultados esperados y condiciones que deben cumplirse mientras se ejecutan las pruebas, con el objetivo de comprobar los errores que pueda tener el sistema, corregirlos y obtener un óptimo funcionamiento del mismo.

4.1 Implementación del Algoritmo *Greedy* para la planificación

Como variables de entradas se definen:

- *Grupos.*
- *Fechas Hábiles.*
- *Guardia.*

Proceso de Algoritmización: El algoritmo se encarga de distribuir los grupos donde la distancia entre la fecha actual y la última fecha en que se asignó guardia a dicho grupo sea máxima

Se distribuyen los grupos entrados de acuerdo a las siguientes reglas:

1- Los grupos que no han realizado guardia son los primeros en el listado de grupos.
2- Al final del listado de grupos se apendizan los grupos restantes (que han realizado al menos 1 guardia) ordenados por la última fecha en que se le asigno guardia, ordenados ascendentemente. A este listado le llamaremos Distribucion_Grupos.
3- Se crean las asignaciones. Una asignación consiste en la relación de 1 grupo, en 1 guardia, en 1 día. Para ello, por cada grupo en el listado de grupos obtenidos en los pasos 1 y 2 se asigna la fecha correspondiente. Esto sería: Asignacion {Distribucion_Grupos[i], Fechas_Habiles[i], Guardia}.

Se repite el proceso tantas veces hasta se hayan cubierto todos los fechas hábiles.

4.2 Modelo de implementación

El modelo de implementación representa cómo los elementos del modelo de diseño se implementan en términos de componentes. De igual forma describe cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y la relación existente entre ellos. (33)

4.2.1 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo de diseño. (33)

A continuación se muestra el diagrama de componentes para los CU Planificar Guardia, Realizar Control de Guardia y Generar Estadísticas.

Figura 14: Diagrama de componentes del CU Planificar Guardia

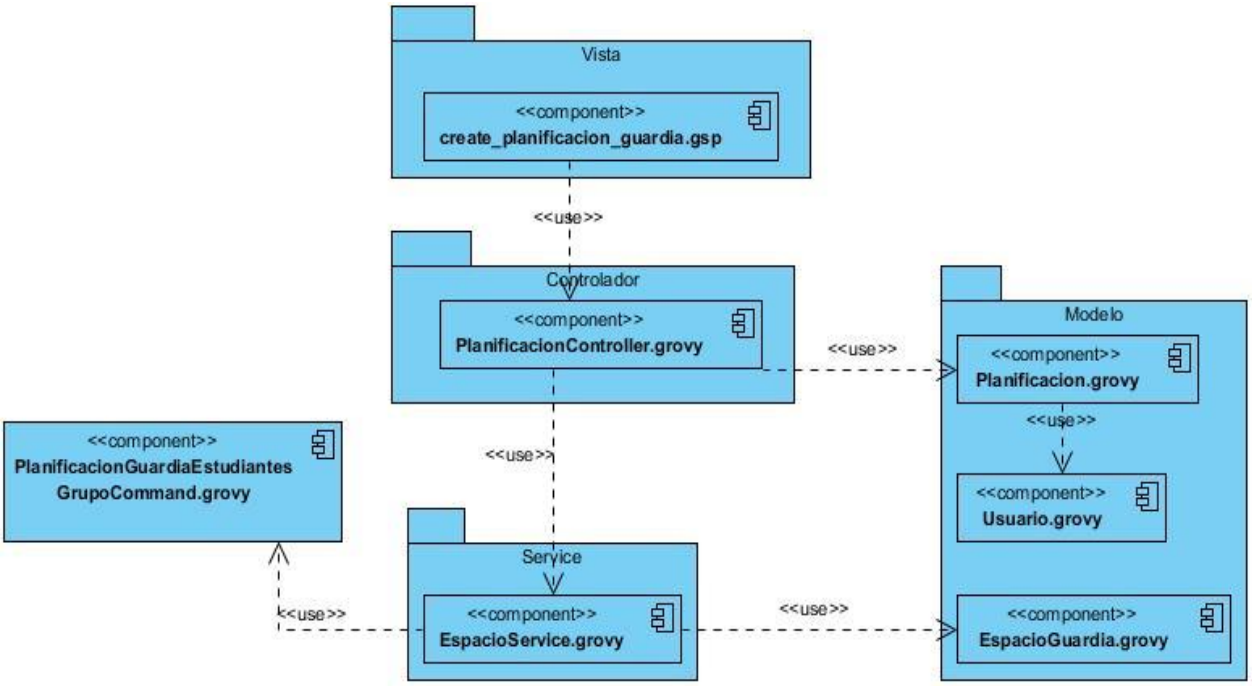


Figura 15: Diagrama de componentes del CU Realizar Control de Guardia

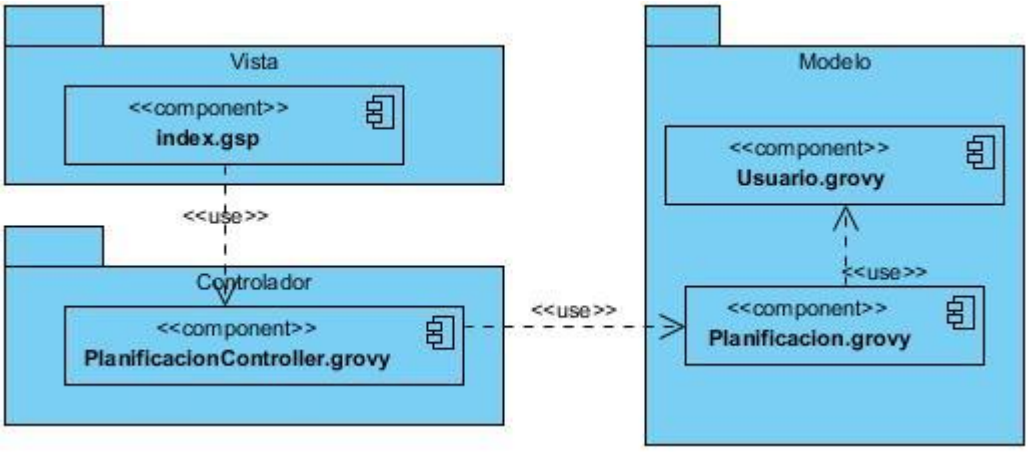
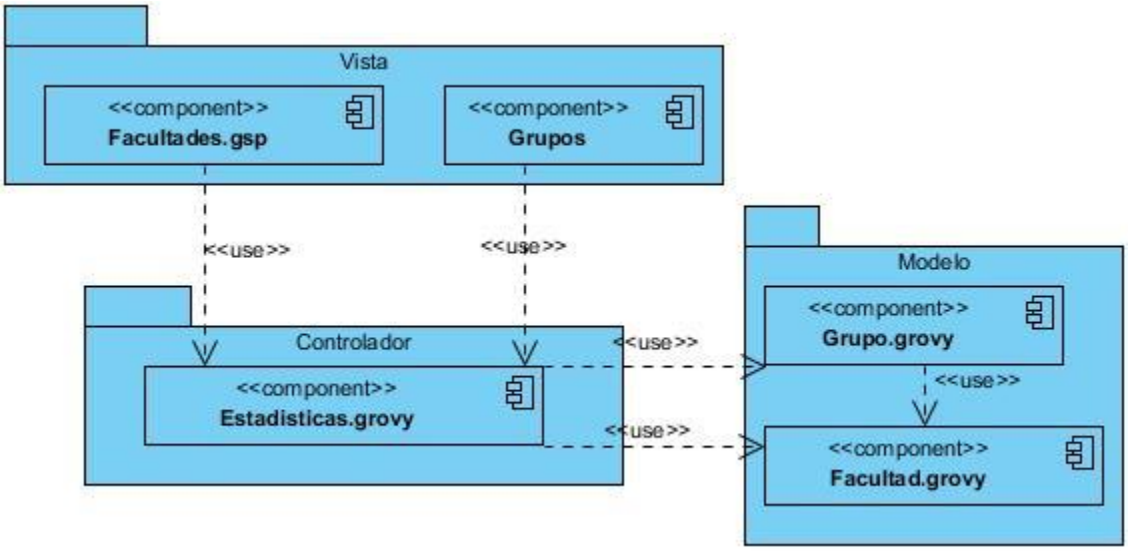


Figura 16: Diagrama de componentes del CU Generar Estadísticas



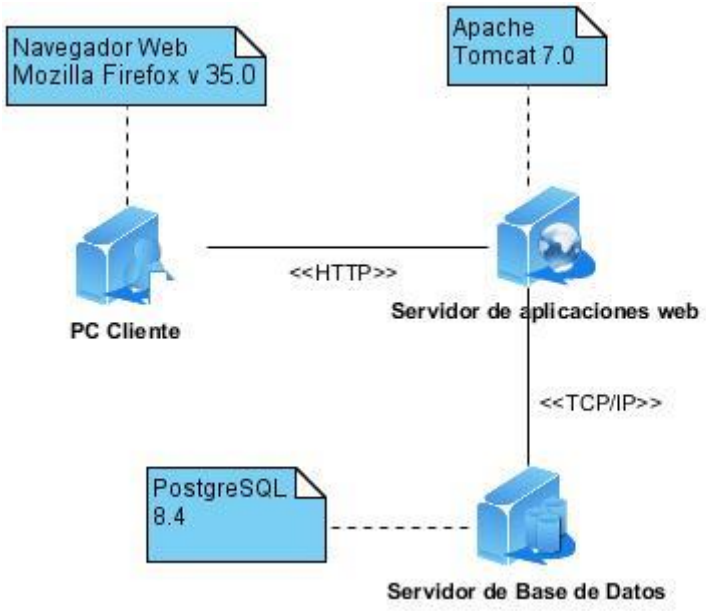
4.2.2 Diagrama de Despliegue

El diagrama de despliegue se utiliza para modelar la configuración de los elementos de procesado en tiempo de ejecución y de los componentes, procesos y objetos de *software* que estos agrupan y sus relaciones. Además, en este modelo se tienen en cuenta los protocolos de comunicación entre los nodos.

Al sistema se puede acceder desde un cliente utilizando un navegador web que cumpla con los estándares definidos por la W3C (*World Wide Web Consortium*).

A continuación se muestra el diagrama de despliegue del sistema.

Figura 17: Diagrama de Despliegue



Descripción de los elementos del diagrama

- ✓ **Nodo PC Cliente:** Representa las computadoras que utilizarán los usuarios para interactuar con la aplicación. Establece comunicación con el servidor de aplicaciones a través de protocolo HTTP.

- ✓ **Nodo Servidor de aplicaciones:** En este nodo se encuentran los scripts de la aplicación.

- ✓ **Nodo Servidor de Base de Datos:** En este nodo se encuentra el servidor de Base de datos del sistema, el cual se conecta al servidor de aplicaciones a través de la familia de protocolos TCP/IP.

Descripción de los protocolos de comunicación

- ✓ **Familia de protocolos TCP/IP:** se utiliza para enlazar computadoras que usan sistemas operativos similares o diferentes, incluyendo pc, minicomputadoras y computadoras centrales sobre redes de área local (LAN). En el caso del sistema se utiliza para interconectarlo con el servidor de base de datos.

- ✓ **HTTP:** Protocolo de Transferencia de Hipertexto, es usado en cada transacción de la web. Define la sintaxis y semántica que utilizan los elementos software de la arquitectura web (cliente, servidor, proxy para comunicarse). En el sistema se utiliza este protocolo para el acceso de los clientes web a los servicios que brinda la aplicación.

4.3 Pruebas de *software*

La calidad del software es uno de los mayores problemas que se afrontan actualmente en la esfera de la informática. Por su parte el proceso de pruebas es sin dudas uno de los aspectos fundamentales para medir el estado de calidad de un sistema informático.

Las pruebas son un elemento crítico, que ayuda a mejorar la calidad de *software* y representa una revisión final de las especificaciones del diseño y de la codificación. Son una actividad en la cual, el producto desarrollado es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados, registrados y se realiza una evaluación en cuanto a los resultados esperados.

A continuación se enuncias los elementos a tener en cuenta para que una prueba tenga éxito:

- Estrategia de prueba
- Niveles de prueba
- Tipo de prueba
- Método de prueba
- Técnica de prueba
- Caso de prueba

Estrategia de Prueba

Una estrategia de prueba del software integra los métodos de diseño de caso de pruebas del software en una serie bien planeada de pasos que desembocará en la eficaz construcción del software. Cualquier estrategia de prueba debe incorporar la planeación, el diseño de casos de pruebas, la ejecución, la recolección y evaluación de los datos resultantes.

Una estrategia de prueba del software debe ser lo suficientemente flexible como para promover un enfoque personalizado. Al mismo tiempo, debe ser lo adecuadamente rígido como para promover una planeación razonable y un seguimiento administrativo del avance del proyecto.

Con el objetivo de verificar el correcto funcionamiento del sistema se realizaron **pruebas a nivel de sistema.** Dentro de los tipos de pruebas existentes, con el objetivo de verificar el correcto funcionamiento de las funcionalidades desarrolladas, se realizaron **pruebas de función**. Además se aplicaron **pruebas de regresión**; el objetivo de estas es garantizar que ante cualquier modificación en el código, ya sea por mantenimiento o por la incorporación de una nueva funcionalidad, no afecte el correcto funcionamiento de otra parte del producto. Generalmente se realizan después de la corrección de un defecto, o la adición de nuevas funcionalidades.

Existen dos **métodos de prueba**, el procedimiento de Caja Negra y el de Caja Blanca, de ellos a la solución desarrollada se aplicó el de **Caja Negra**. Básicamente este método de prueba se lleva a cabo sobre la interfaz del sistema y se basa en el análisis de los datos de entrada y salida, sin tener en cuenta la estructura interna del *software*.

Para evaluar en qué grado el sistema cumple con las expectativas, dentro del método de Caja Negra se utilizó **la técnica Partición de Equivalencia**, que permite examinar los valores válidos e inválidos de las entradas y salidas en el *software*. A través del **diseño de casos de prueba**, la técnica seleccionada, descubre de forma inmediata los errores existentes, pues la definición de las clases de equivalencia permite reducir el número de casos de pruebas a efectuar.

Un caso de prueba, es un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. Siempre es ejecutada como una unidad, desde el comienzo hasta el final; debe verificar:

- Si el producto satisface los requerimientos del usuario, tal y como se describe en la especificación de los requisitos.
- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

4.4 Diseño de casos de prueba

Un caso de prueba es un conjunto de acciones con resultados y salidas previstas basadas en los requisitos de especificación del sistema. (38)

Para comprobar el correcto funcionamiento de las funcionalidades definidas, se realizó el diseño de los casos de prueba para cada interfaz del sistema. Se especificó en cada caso de prueba la

información de entrada, los resultados que se deben obtener una vez ejecutado el caso de prueba y las condiciones que deben cumplirse mientras estos se ejecuten.

Se presentan a continuación el Caso de Prueba del CU Gestionar Guardia. A pesar de no ser de prioridad alta, se considera uno de los más completos a mostrar, pues contiene cuatro escenarios de prueba.

Los elementos que contiene el Caso de Prueba son:

Descripción general: Breve descripción del caso de prueba.

- ✓ **Condiciones de ejecución**: Condiciones que deben cumplirse para poder ejecutar el caso de prueba.
- ✓ **Sección**: Representado por las siglas SC y corresponden a cada uno de los requisitos funcionales que agrupa el Caso de Uso.
- ✓ **Escenario**: Representado por las siglas EC y corresponden a cada uno de los flujos básicos y alternos del Caso de Uso.
- ✓ **Descripción**: Es una descripción del escenario a probar y refleja su objetivo fundamental.
- ✓ **Variables**: Campos que deben tenerse en cuenta para completar los datos de los formularios de entrada.
- ✓ **Respuesta del sistema:** Es la respuesta que da el sistema ante un determinado escenario de prueba.
- ✓ **Flujo básico:** Secuencia de pasos para poder probar el Caso de Prueba contra la aplicación.

En la descripción de las variables se especifica:

- ✓ **Nombre del campo:** Nombre que tiene la variable en la interfaz de usuario.
- ✓ **Clasificación:** Clasificación que tiene la variable en el formulario.
- ✓ **Valor nulo:** Especifica si la variable puede ser vacía o no.

- ✓ **Descripción:** Es la descripción de la variable. Se especifican los valores que esta puede tomar.

Descripción general

Permite registrar, mostrar, actualizar y eliminar una guardia.

Condiciones de ejecución

El usuario debe estar autenticado.

SC1. Registrar guardia

Escenario	Descripción	Fecha inicial	Fecha Final	Para quien	Postas	Turnos	Respuesta del sistema	Flujo central
EC 1.1 Registrar guardia	Permite registrar una guardia exitosamente en el sistema.	V 30 05 2016	V 31 05 2016	V Estudiantes	V Cajeros	V 10:00-2:00	Registra una guardia en el sistema. Muestra el mensaje: "La guardia ha sido registrada	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Nuevo Guardia". 3- Introducir correctamente los datos de la guardia.

							satisfactoriamente." .	4- Presionar el botón "Crear".
EC 1.2 Cancelar la operación	Permite cancelar el registro de una guardia.	N/A	N/A	N/A	N/A	N/A	Cancela las operaciones realizadas sobre la entidad. Permite regresar a la interfaz anterior.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Nuevo Guardia". 3- Presionar el botón “Cancelar”.
EC 1.3 Campos obligatorios vacíos	Permite validar que no existan campos obligatorios vacíos.	V	V	V	I	V	Muestra el mensaje de error: “Existen campos obligatorios vacíos, por favor complete estos campos”.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Nuevo Guardia". 3-Dejar algún campo obligatorio vacío. 4- Presionar el botón "Crear".
		30 05 2016	31 05 2016	Estudiantes	Vacío	10:00-2:00		
		V	V	V	V	I		
		30 05 2016	31 05 2016	Estudiantes	Cajeros	Vacío		

SC2. Mostrar guardia

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 2.1 Visualizar indicador	Permite visualizar los detalles de un indicador.	Muestra una tabla con las guardias registradas. Permite: - Ver Planificación en caso de que la guardia esté planificada. - Planificar la guardia en caso de que no esté planificada (Ver CP Planificar Guardia). - Mostrar guardia.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar la opción "Mostrar", correspondiente a la guardia que se desea consultar.
EC 2.2 Cerrar la interfaz	Permite cerrar la interfaz con los datos del indicador.	Cierra la interfaz que contiene los datos del indicador y regresa a la interfaz anterior.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho.2- Seleccionar la opción "Mostrar", correspondiente a la guardia que se desea consultar. 3- Seleccionar botón "Cerrar".

SC3. Actualizar guardia

Escenario	**Descripción**	**Número**	**Nombre**	**Consulta**			**Respuesta del sistema**	**Flujo central**
EC 3.1 Actualizar guardia	Permite actualizar una guardia exitosamente en el sistema.	V	V	V	V	V	Actualiza una guardia en el sistema. Muestra el mensaje: "Se han actualizado los datos.". Permite: - Borrar Planificación. - Ver planificación.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Mostrar". 3- Seleccionar el botón "Editar". 4- Introducir correctamente los datos de la guardia. 5- Presionar el botón "Actualizar".
		30 05 2016	31 05 2016	Estudiantes	Cajeros	10:00-2:00		

EC 3.2 Cancelar la operación	Permite cancelar la edición de los datos del indicador seleccionado.		N/A	N/A			Cancela las operaciones realizadas sobre la entidad. Permite regresar a la interfaz anterior.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Mostrar". 3- Seleccionar el botón "Editar". 4- Presionar el botón “Cancelar”.
EC 3.3 Campos obligatorios vacíos	Permite validar que no existan campos vacíos.	V 30 05 2016 V	V 31 05 2016 V	V Estudiantes V	I Vacío V	V 10:00-2:00 I	Muestra el mensaje de error:	1- Seleccionar la opción "Guardia"

		30 05 2016	31 05 2016	Estudiantes	Cajeros	Vacío	“Existen campos obligatorios vacíos, por favor complete estos campos”.	dentro del menú superior derecho. 2- Seleccionar el botón "Mostrar". 3- Seleccionar el botón "Editar". 4- Dejar vacío algún campo obligatorio. 5- Presionar el botón "Actualizar" .

SC4. Eliminar guardia

Escenario	Descripción	Respuesta del sistema	Flujo central

EC 4.1 Eliminar una guardia	Permite eliminar una guardia.	Muestra un mensaje para confirmar la eliminación de la guardia: "¿Está seguro que desea eliminar el elemento?". Elimina el elemento seleccionado.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Mostrar". 3- Seleccionar el botón "Eliminar".
EC 4.2 Cancelar la operación	Se cancela la eliminación de un indicador.	Muestra los detalles de la guardia.	1- Seleccionar la opción "Guardia" dentro del menú superior derecho. 2- Seleccionar el botón "Mostrar". 3- Seleccionar el botón "Eliminar". 4- Presionar el botón "Cancelar".

SC5. Planificar Guardia

Escenario	Descripción	Respuesta del sistema	Flujo central

EC 5.1 Planificar guardia a estudiantes	Permite planificar la guardia a los estudiantes, comprobando en 163 días.	Planifica la guardia de estudiantes, planificando cada grupo con al menos 15 días entre una guardia y la otra	1- Seleccionar el menú "Guardias" en la parte superior. 2- Seleccionar la opción "Planificar" 3- Seleccionar los grupos o los trabajadores involucrados. 4- Seleccionar las fechas no hábiles para realizar guardia. 5- Seleccionar el botón "Planificar".

Descripción de las variables.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	*Fecha inicial*	*Fecha*	*No*	*Es la fecha inicial de la guardia. Debe tener el formato DD/MM/AAAA*
2	*Fecha Final*	*Fecha*	*No*	*Es la fecha final de la guardia. Debe tener el formato DD/MM/AAAA*
3	*Para quien*	*Lista de selección*	*No*	*Identifica al personal que realiza la guardia. Tiene valores predeterminados: Estudiante y trabajador*
4	*Postas*	*Lista de selección*	*No*	*Identifica las postas a cubrir durante la guardia. Los valores que toma dependen de los datos registrados en el sistema.*

5	*Turnos*	*Lista de selección*	*No*	*Identifica los turnos a cubrir durante la guardia. Los valores que toma dependen de los datos registrados en el sistema.*

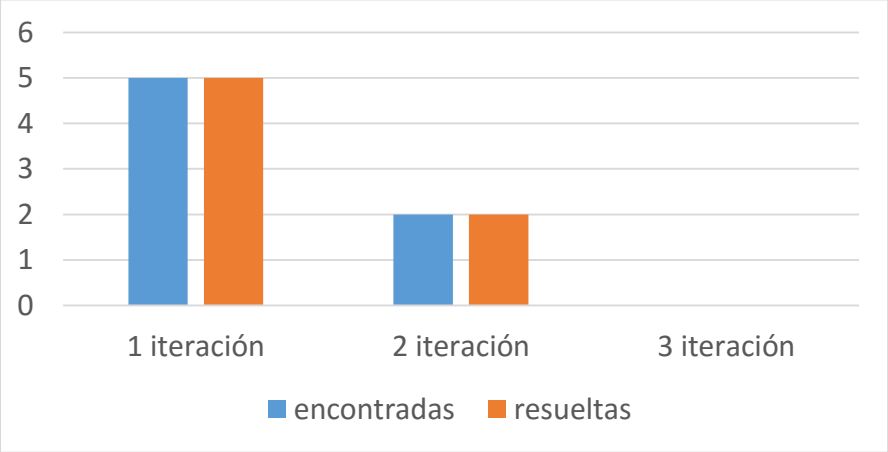
4.5 Resultados de las pruebas

Luego de realizar las pruebas se arribaron a los siguientes resultados.

Se realizaron 3 iteraciones de pruebas. En una primera iteración se identificaron un total de 5 No Conformidades (NC), las cuales fueron analizadas y resueltas. Luego se realizan las pruebas de regresión para comprobar que todas las NC encontradas fueron resueltas y que estos cambios no afectan el resto de la solución. En una segunda iteración se detectaron un total de 2 NC y también fueron resueltas. Aplicadas las pruebas de regresión se procede al a tercera iteración donde no se detectaron NC.

En la siguiente tabla se muestran los resultados de las NC detectadas.

Figura 18: Resultados de las pruebas funcionales



4.6 Conclusiones del capítulo

En este capítulo se abordaron los flujos de trabajo Implementación y Prueba. Inicialmente se modeló el Diagrama de Componentes, representando la relación entre los mismos dentro del sistema; lo cual permitió, conocer de qué manera se ubican los ficheros que contendrán el código fuente.

De modeló el Diagrama de Despliegue, indicando la distribución física de *hardware* y *software* para el despliegue de la solución.

Se realizaron las pruebas definidas en la Estrategia de Pruebas, encontrando un total de 7 NC. Todas las NC fueron corregidas de forma satisfactoria al concluir cada iteración, demostrando la calidad de la solución y el cumplimiento de los requisitos funcionales definidos.

Conclusiones

Luego del desarrollo del presente Trabajo de Diploma, se concluye con el cumplimiento del objetivo general propuesto desarrollándose una Aplicación informática para gestionar la Guardia Obrera Estudiantil en las facultades de la Universidad de las Ciencias Informáticas. De manera general se obtuvieron los siguientes resultados:

- ✓ Se definió el marco teórico en el que se sustenta la investigación, conceptualizando la aplicación informática de gestión de guardia.
- ✓ Se realizó el análisis y el diseño de la aplicación informática para la gestión de la Guardia Obrera Estudiantil, obteniendo como resultado la generación de los artefactos correspondientes con la metodología AUP-UCI.
- ✓ Se implementó la propuesta de solución haciendo uso de las herramientas, tecnologías y lenguajes definidos.
- ✓ Se validó la solución aplicando los métodos de pruebas, lo que permitió la identificación de no conformidades existentes en la aplicación y su posterior corrección.

Recomendaciones

Se recomienda que el sistema sea desplegado en la UCI e instituciones que realicen este tipo de procesos.

Glosario de términos

Administrador: Persona que tiene privilegios para determinadas funcionalidades del sistema.

Caso de uso: Operación/tarea específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o bien desde la invocación desde otro caso de uso.

Clase: Elemento de software que describe o caracteriza una entidad del mundo real o un entidad del espacio de implementación.

Día hábil: Día en que normalmente se realiza la guardia.

Framework: Conjunto de APIs (Interfaz de Programación de Aplicaciones, del inglés: *Application Programming Interface*) y herramientas destinadas a la construcción de un determinado tipo de aplicaciones de manera generalista.

Gestión: Según el Diccionario de la Real Academia Española la palabra "gestión" tiene dos significados: acción y efecto de gestionar y acción y efecto de administrar.
Es la acción de gestionar y administrar una actividad profesional destinada a establecer los objetivos y medios para su realización, a precisar la organización de sistemas, con el fin de elaborar la estrategia del desarrollo y a ejecutar la gestión del personal. Asimismo en la gestión es muy importante la acción, porque es la expresión de interés capaz de influir en una situación dada**.** (39)
Guardia: La Real Academia Española define el vocablo guardia como: "Acción de guardar, tener cuidado con algo. Defensa, custodia, protección. En algunas profesiones o establecimientos, servicio que asegura la continuidad de prestaciones fuera de su horario habitual. Cuerpo encargado de las funciones de vigilancia o defensa." (40)
Herramientas CASE: Las herramientas CASE (Ingeniería de Software Asistida por Ordenador, del inglés: *Computer Aided Software Engineering*) son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el coste de las mismas en términos de tiempo y de dinero.

Implementación: Proceso por el cual se escribe (en un lenguaje de programación), se prueba, se depura y se mantiene el código fuente de un programa informático.

Personal Incapacitado: Son las personas que presentan algún impedimento para realizar la guardia mensualmente. Entre estas personas se encuentran: las que tienen cualquier enfermedad que no permita exponerse al sereno, los impedidos físicos, las mujeres embarazadas y cualquier otra situación de caracter excepcional.

Proceso: Conjunto de actividades relacionadas lógicamente que producen una salida o resultado.

Requisito Funcional: Requisito que especifica una acción que debe ser capaz de realizar el sistema, sin considerar restricciones físicas. Requisito que especifica comportamiento de entrada/salida de un sistema.

Requisito No Funcional: Requisito que especifica propiedades del sistema, como restricciones del entorno o de implementación, rendimiento, dependencias de la plataforma, extensibilidad o fiabilidad. Requisito que especifica restricciones físicas sobre un requisito funcional.

Bibliografía

1. **Lotti, Alina M**. *Informatización de la sociedad cubana: Impostergable desafío. Trabajadores. Órgano de la Central de Trabajadores de Cuba.* [En línea] 23 de Febrero de 20015. [Citado el: 29 de Mayo de 2016.] http://www.trabajadores.cu/20150223/informatizacion-de-la-sociedad-cubana-impostergable-desafio/.

2. **Perissé, Marcelo Claudio.** Ciencia y Técnica Administrativa. *cyta.* [En línea] [Citado el: 2 de abril de 2016.] http://www.cyta.com.ar/ta0104/articulos/ti/ti.htm.

3. Gitman, Lawrence J. *Fundamentos de administración financiera.* México : Oxford University Press, 1997.

4. **Nogareda Cuixart , Clotilde y Nogareda Cuixart, Silvia.** NTP 455: Trabajo a turnos y nocturno: aspectos organizativos. *http://www.insht.es.* [En línea] [Citado el: 15 de Junio de 2016.] http://www.insht.es/InshtWeb/Contenidos/Documentacion/FichasTecnicas/NTP/Ficheros/401a500/ntp_455.pdf.

5. Definición. [En línea] [Citado el: 15 de Junio de 2016.] http://definicion.mx/grupos/.

6. Definición. [En línea] [Citado el: 15 de Junio de 2016.] http://definicion.mx/reporte/.

7. Definicion y etimología. [En línea] [Citado el: 15 de Junio de 2016.] https://definiciona.com/asistencia/.

8. SICATU - Sistema de Cálculo de Turnos. [En línea] [Citado el: 4 de abril de 2016.] https://sites.google.com/site/sicatu30/.

9. **molina, David.** molda.es. [En línea] 9 de febrero de 2012. [Citado el: 4 de abril de 2016.] http://molda.es/sicatu-sistema-de-calculo-de-turnos/.

10. Tamigo. [En línea] 2015-2016. [Citado el: 4 de abril de 2016.] http://www.tamigo.es/productos/m%C3%B3dulos/calendario-de-turnos/.

11. EIMER software. [En línea] [Citado el: 4 de abril de 2016.] http://www.eimer.es/jano-planificacion-hospitalaria.html.

12. Control. Soluciones tecnológicas. [En línea] [Citado el: 4 de abril de 2016.] http://www.controlsistemes.com/upload/documents/folleto-extendido-visualtime-live.pdf.

13. TURNEX: Excelencia en la Gestión Operativa de Turnos. *SOLEX.* [En línea] 08 de Julio de 2013. [Citado el: 15 de Junio de 2016.] http://solexservices.com/index.php/noticias/110-turnex-excelencia-en-la-gestion-operativa-de-turnos.

14. **Torres Aguilera, Grettell y Almeida Oquendo, Humberto**. *Gestor Web para el control de la Guardia Obrera de la Universidad de las Ciencias Informáticas (UCI). Sitio de la Biblioteca de la Universidad de las Ciencias Informáticas.* [En línea] Junio de 2009. [Citado el: 18 de Abril de 2016.]
http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2309_09/1/TD_2309_09.pdf.

15. **Mata Sánchez, Dailín y Navarro Quintero, Aroldo.** *Sistema para la Gestión de la Guardia Obrera-Estudiantil en la Facultad 3*. *Sitio de la Biblioteca de la Universidad de las Ciencias Informáticas.* [En línea] Junio de 2013. [Citado el: 18 de Abril de 2016.] http://bibliodoc.uci.cu/RDigitales/2013/noviembre/27/TD_06874_13.pdf.

16. **Bejerano González, Karel y Montes de Oca Barrios, Félix Miguel.** *Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7. Sitio de la Biblioteca de la Universidad de las Ciencias Informáticas.* [En línea] Junio de 2009. [Citado el: 18 de Abril de 2016.]
http://repositorio_institucional.uci.cu/jspui/bitstream/ident/TD_2076_09/1/TD_2076_09.pdf.

17. **Jiménez Barreto, Nardely y Concepción Fernández, Yasser.** *Aplicación web para la gestión de la planificación de la guardia Obrera estudiantil en la Universidad de las Ciencias Informáticas. Sitio de la Biblioteca de la Universidad de las Ciencias Informáticas.* [En línea] Junio de 2013. [Citado el: 18 de Abril de 2016.] http://bibliodoc.uci.cu/RDigitales/2013/noviembre/19/TD_06843_13.pdf.

18. **Cohen, Ellis.** project.net. *An Introduction to Algorithms for Solving Schedule-Related Problems.* [En línea] Enero de 2015. [Citado el: 20 de Junio de 2016.] http://www.project.net/introduction-algorithms-solving-schedule-related-problems.

19. **Franco Martínez, Edgardo Adrián.** *Algoritmos ávidos.* [En línea] 1 de Julio de 2015. [Citado el: 20 de Junio de 2016.]
http://eafranco.com/docencia/analisisdealgoritmos/files/10/Diapositivas10.pdf.

20. **Sánchez Rodríguez, Tamara.** *Metodología de desarrollo para la Actividad productiva de la UCI.* Habana : s.n., 2014.

21. **Abdul-Jawad, B.** *Groovy and Grails Recipe.* New York : NY Apress, 2009. ISBN 978-1-4302-1600-1.

22. **J. Gitiérrez, Javier.** *¿Qués es un framework? Lenguajes y Sistemas Informáticos. Universidad de Sevilla.* [En línea] [Citado el: 19 de Abril de 2016.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

23. **Dickinson, Jon.** *Grails 1.1 Web Application Development.* s.l. : Packt Publishing, 2009. ISBN 9781847196682.

24. **Cockbun, Alistar.** *Agile Software Development, The Cooperative Game.* s.l. : Addison-Wesley Professional, 2006. ISBN 0-321-48275-1.

25. **Larman, Craig.** *Uml y Patrones.* Buenos Aires. Argentina : Prentice Hall, 2004. ISBN 8420534382.

26. Visual paradigm. *¿Qué es Visual Paradigm?* [En línea] [Citado el: 19 de Abril de 2016.] https://www.visual-paradigm.com/features/.

27. Postgres.sql. *Qué es Postgres SQL.* [En línea] [Citado el: 19 de Abril de 2016.] http://www.postgresql.org/docs/9.3/static/intro-whatis.html.

28. **Martinez Gerrero, Rafael.** PostgreSQL-es. *Sobre PostgreSQL.* [En línea] 2 de Octubre de 2010. [Citado el: 19 de Abril de 2016.] http://www.postgresql.org.es/sobre_postgresql.

29. Foundation, The Apache Software. Apache Tomcat. [En línea] [Citado el: 19 de Abril de 2016.] http://tomcat.apache.org/index.html.

30. Intellig IDEA. [En línea] JetBrains s.r.o, 2016. [Citado el: 21 de abril de 2016.] https://www.jetbrains.com/idea/.

31. **G. Ross, Ronald.** *The Business Rules Manifesto. businessrulesgroup.* [En línea] 1 de noviembre de 2003. [Citado el: http://www.businessrulesgroup.org/brmanifesto.htm25 de abril de 2016.]

32. *Softweare Engineering.* 2006.

33. **Jacobson, Ivar, Booch, Grady y Rumbaugh,James.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000. ISBN 84-7829-036-2..

34. **Cevallos, Karla.** *UML: Diagrama de Secuencia. Ingeniería de software. Portafolio Digital.* [En línea] 10 de Julio de 2015. [Citado el: 3 de Mayo de 2016.] https://ingsotfwarekarlacevallos.wordpress.com/2015/07/07/uml-diagrama-de-secuencia/.

35. Grails Model View Controller pattern. *Wideskills Widen Skills an Expand Oportunities.* [En línea] 2015. [Citado el: 22 de Abril de 2016.] http://www.wideskills.com/grails/grails-model-view-controller-pattern.

36. Arquitectura de software. [En línea] [Citado el: 1 de Mayo de 2016.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/capitulo2.pdf.

37. **Brito, Nacho.** *Manual de desarrollo web con grails.* s.l. : Ediciones ágiles, 2009. ISBN 978-84-613-2651.

38. **Aristegui O, José Luis.** *Los casos de prueba en la prueba del software. http://www.funlam.edu.co.* [En línea] 12 de Abril de 2010. [Citado el: 2 de Mayo de 2016.] http://www.funlam.edu.co/revistas/index.php/lampsakos/article/viewFile/785/754.

39. *eumed.net. Enciclopedia virtual.* [En línea] [Citado el: 2 de Abril de 2016.] http://www.eumed.net/libros-gratis/2013a/1321/gestion.html.

40. Diccionario de la Lengua Española. [En línea] [Citado el: 2 de Abril de 2016.] http://dle.rae.es/?id=JjpuE9J.

41. **Noguera García, Manuel .** *Introducción al Modelado de Procesos de Negocio. Universidad de Granada .* [En línea] 1 de Febrero de 2016. [Citado el: 20 de Abril de 2016.] http://www.ugr.es/~mnoguera/collaborative_systems-business_processes_10-11.pdf.

42. Concepto.de. *Concepto.de.* [En línea] [Citado el: 1 de Mayo de 2016.] http://concepto.de/gestion/.

Anexos

Anexo 1. Criterios para evaluar la complejidad y prioridad de los CU

Criterios de Complejidad
La complejidad de los requisitos se utiliza para estimar el esfuerzo de implementación de este y planificar en qué iteración se implementará. Un requisito puede tener complejidad alta, media o baja .Para determinar la complejidad de los requisitos se analizan los criterios que se describen a continuación.
• **Complejidad por número de transacciones:** Una transacción es un intercambio de datos a nivel de procesos elementales entre las fronteras del sistema, que fuerzan a la lógica de la aplicación a realizar un procesamiento para entrar y/o mostrar información. La complejidad viene representada en el esfuerzo de implementación de un número elevado de procedimientos reflejados en una transacción. **Alta:** El Caso de Uso contiene más de 8 transacciones. **Media:** El Caso de Uso contiene de 5 a 8 transacciones. **Baja:** El Caso de Uso contiene de 1 a 4 transacciones.
• **Complejidad por entidades candidatas:** Las entidades candidatas son las futuras clases de la aplicación que modelarán los elementos persistentes de la base de datos. Una clase envuelve atributos, procedimientos y está relacionada a clases de control, por lo que un monto elevado de entidades o de atributos de la entidad refleja una complejidad en la implementación de las operaciones de un caso de uso que las contiene. **Alta:** Más de 5 entidades. **Media:** De 3 a 5 entidades. **Baja:** De 1 a 2 entidades.

• **Complejidad por interfaces de comunicación con actores:** Las interfaces de comunicación muestran a un actor del sistema la forma de intercambiar información ya sea para la entrada, como para la salida. Un actor puede estar instanciado en una persona o un sistema externo que interactúe con el sistema. La complejidad viene reflejada por el esfuerzo de la implementación de dichas interfaces unida a las restricciones y funcionalidades de su uso. **Alta:** Una persona que interactúa con el sistema mediante una interfaz gráfica donde se evidencie una de las situaciones siguientes:- La interfaz posee elementos de animación en tercera dimensión 3D y se vale de ellos para la gestión de la misma. - La interfaz utiliza gráficos en segunda dimensión 2D para la muestra de información y se vale de ellos para la gestión de la misma. - La interfaz está formada por más de 15 componentes tradicionales de captura y muestra de información. **Media:** - La interfaz contiene desde 8 hasta 15 componentes tradicionales de captura y muestra de información. **Baja:** - La interfaz contiene desde 1 hasta 7 componentes tradicionales de captura y muestra de información. - Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, DLL) - Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto (Servicios Web, XML, Correo electrónico).
• **Complejidad por número de requisitos no funcionales:** Los requisitos no funcionales son características que imponen restricciones al sistema, por lo que condicionan la implementación de los Casos de Uso al agregarle especificidades a las funcionalidades del mismo. **Alta:** Más de 6 requisitos no funcionales asociados. **Media:** Entre 4 y 6 requisitos no funcionales asociados. **Baja:** Menos de 4 requisitos no funcionales asociados.

• **Complejidad por tipo de tecnología:** La utilización de los avances tecnológicos de software o hardware que necesiten de una asimilación de su funcionamiento o comunicación condicionan el esfuerzo de la implementación de Casos de Uso que los empleen. **Alta:** La realización del caso de uso utiliza elementos, plataformas, componentes tecnológicos avanzados de conocimiento externo al equipo de programación que necesiten de capacitación. **Media:** La realización del caso de uso utiliza elementos tecnológicos avanzados donde el conocimiento reside en el equipo de implementación, producto de la selección y preparación inicial en los acuerdos tecnológicos del contrato. **Baja:** No se emplean nuevas técnicas, plataformas, componentes ni conceptos de programación.
• **Complejidad por reutilización:** El uso de patrones, componentes predefinidos del lenguaje o plataformas que generen partes de las funcionalidades del sistema aminoran el esfuerzo en la construcción del software. **Alta:** No se reutilizan elementos, el caso de uso se codifica por completo por un programador. **Media:** Se reutilizan elementos donde es necesario ajustar la codificación antes de la generación de la implementación del caso de uso. **Baja:** Se utilizan elementos que implementan el caso de uso y sólo es necesario realizar pequeños ajustes.

Tabla 29: Criterios para evaluar la complejidad de los CU

Criterios de prioridad
Para determinar la prioridad de los Casos de Uso a implementarse y elaborar un cronograma que responda a la división en iteraciones priorizando los Casos de Uso arquitectónicamente más significativos, se valorarán los Casos de Uso de acuerdo a los indicadores que se muestran:
• **Beneficios:** Importancia desde el punto de vista del negocio. Debe ser suministrada por un experto funcional del negocio. **Crítico o primario (Alta):** Se refieren a las tareas principales del sistema, las que al fallar llevan a la no realización de los objetivos del sistema.

Importante o secundario (Media): Tienen que ver con el soporte de actividades del sistema, como generación de reportes, datos estadísticos, supervisión. Si estas tareas faltan el sistema puede continuar por un tiempo prudencial cumpliendo con su misión, pero con una degradación de sus servicios. **Auxiliares u Opcionales (Útiles o deseables)(Media):** Son facilidades de uso, no afectan en nada la funcionalidad del sistema en el logro de su misión.
• **Dependencia:** Se refiere a la naturaleza y cantidad de relaciones que se establece entre un caso de uso y otro, de tipo inclusión, extensión o generalización. **Alta:** Si las relaciones donde el caso de uso incluye, extiende o especializa a otros Casos de Uso, se encuentran en mayor número. **Media:** Si existe un balance entre las relaciones tipo. **Baja:** Si las relaciones donde el caso de uso es incluido, extendido o generalizado por otros Casos de Uso, se encuentran en mayor número.
• **Estabilidad:** Criterio de estabilidad del caso de uso ante los procesos de cambio que ocurren en la organización. Debe ser suministrada por un experto funcional del negocio. **Alta:** Mayor del 50 %. **Media:** Entre el 10 % y el 50 %. **Baja:** Menor del 10 %.
• **Frecuencia:** Ayuda a identificar los Casos de Uso que mayor nivel de ocurrencia tienen y por lo tanto se desean tener implementados con anterioridad. Debe ser suministrada por un experto funcional del negocio. **Alta:** Diaria. **Media:** Semanal. **Baja:** Mensual o Rara vez.

Tabla 30: Criterios para evaluar la prioridad de los CU