UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor:

Daniela de la Caridad Herrera González

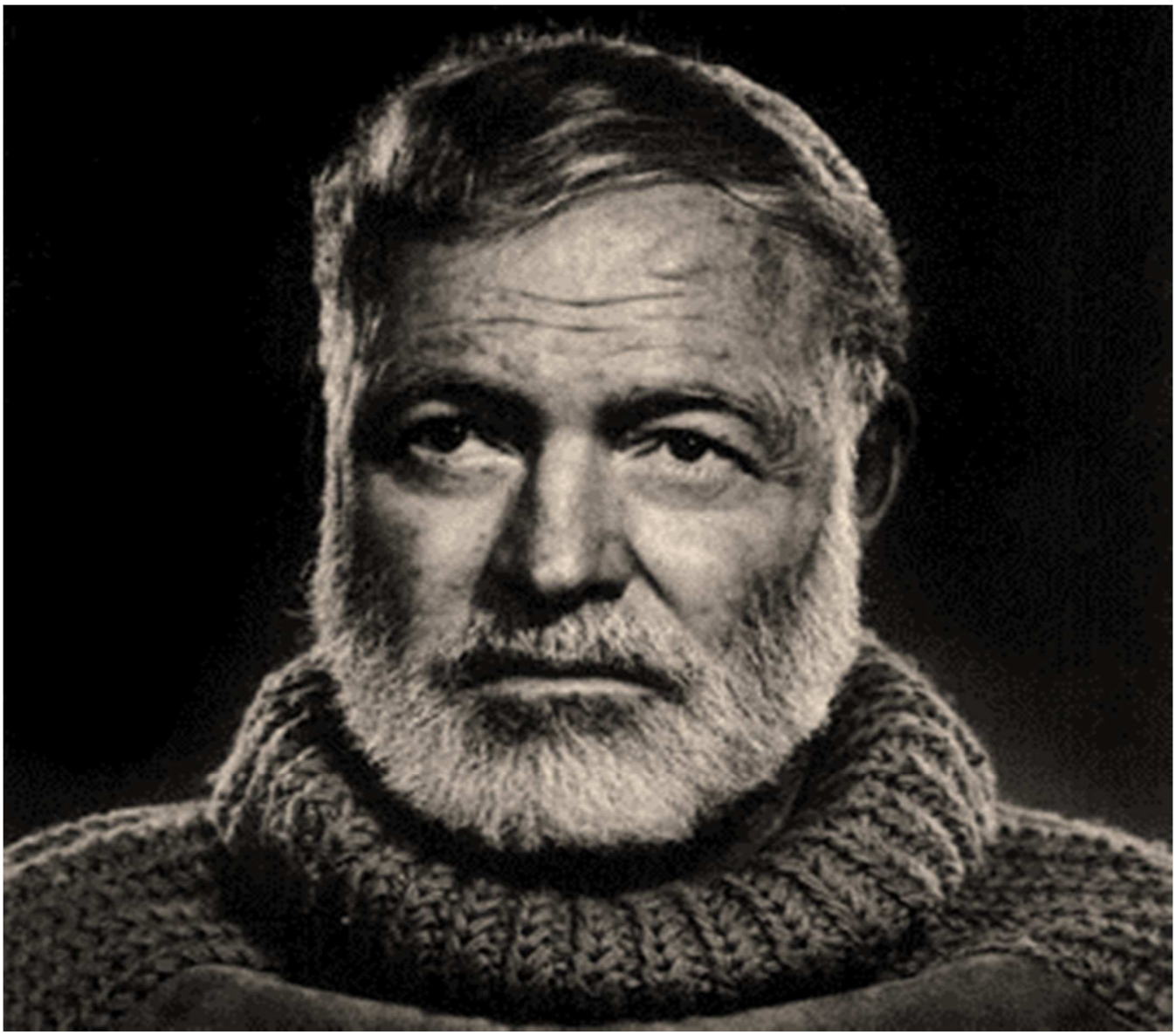
Tutores:

Ing. Yordanka Fuentes Castillo

Ing. Carlos Yordan González Herrera

Lic. Luis Enriquez Benet

La Habana, junio de 2019



"El secreto de la sabiduría, del poder y del conocimiento es la humildad".

Ernest Hemingway.

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo Daniela de la Caridad Herrera González, con carné de identidad 96061309374, soy la autora principal del trabajo titulado “Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los _____ días del mes de junio de 2019.

Ing. Yordanka Fuentes Castillo **Ing. Carlos Yordan González Herrera**

Lic. Luis Enríquez Benet

Autora: Daniela de la Caridad Herrera González

DEDICATORIA

A las estrellas de mi universo, mis padres, por confiar siempre en mí, motivarme a convertirme en una mejor persona, por apoyar cada uno de mis pasos, por su entrega y dedicación y por ser sencillamente especiales.

A mi hermano, por convertirse en mi consejero, por ser la persona que es y por hacerme saber que siempre pase lo que pase puedo confiar en él.

AGRADECIMIENTOS

Quiero agradecer de forma especial a todas las personas que me apoyaron y me dieron la fuerza que necesité para lograr que este sueño se hiciera realidad.

A los profesores que desde el inicio de la carrera, con sus consejos, su enseñanza y dedicación, perfeccionaron aquella estudiante para convertirse hoy en una futura ingeniera, en especial a Manolo, Yanet, Israel, Néstor, Máxora, Dargel, Maykel, Yurisbel, Ponce, Juan Manuel, Lenny muchas gracias por su apoyo en todo momento y por su amistad más que todo.

A mis segundos papas durante esta travesía, gracias por la constante preocupación, la ternura, la paciencia, la dedicación y la entrega para poder lograr que este sueño se volviera realidad, a la mamá por no haberme juzgado desde un primer momento y haber depositado en mi un voto de confianza, al papá por transmitirme siempre tanta paz y seguridad, muchas gracias por las incontables noches y horas dedicadas, sin ustedes no hubiese sido posible, a Luisito mi tutor por la preocupación, el carisma y por transmitir tanta alegría.

A las personas que de una forma u otra ayudaron a que este sueño se hiciera realidad, Yunior, Lierme, Nolberto, Yojahny, gracias por su paciencia y por el tiempo que me dedicaron y en especial a Juan por tanta entrega, no hay manera de retribuir toda la ayuda que has significado, gracias por dejar todo a un lado para ayudarme y por echar pie en tierra conmigo.

A las amistades que llegaron hace poco pero que me hubiese encantado contar con ellos desde un primer momento, Glenda, Jesús, Víctor, Isabela, Charli, Jesús, Reimer, Andy, Leo, Rey, Yaraysi gracias por los momentos que me han permitido compartir con ustedes.

AGRADECIMIENTOS

A Eli, gracias por haberme demostrado tanto en tan poco tiempo por preocuparte por mi, gracias por ser consejera, brazo de apoyo, por escuchar mis tormentos y por hacer tuyas mis preocupaciones.

A Ale por ser una persona de otro mundo, gracias por tu nobleza, tu ayuda incondicional, tu carisma, tu apoyo, gracias por ser el amigo hombre que siempre desee.

A los amigos del aula, a los que me acompañan desde primer año en especial a Mio que desde el primer día cuando tuve la dicha de conocerte me hiciste saber que puedo contar contigo, a Leo por su cariño y por dejarse querer tanto, a Carel por haberme demostrado en este último tiempo que es una excelente persona, al kuki por ser siempre el objeto de conversación y por los buenos momentos, a Dariem por ser un ejemplo a seguir, y a los que quedaron en el camino también, a Tony y a Ignacio, por tantos momentos juntos y tantas vivencias compartidas, también a Maye y Liana por tantas horas de estudio juntas y por tantas risas.

A mis amigas de toda la vida a cada una por las razones que las hacen sencillamente especiales. A mis mellis por dedicarle tanto a esta amistad, me considero una persona muy afortunada porque las tengo en mi vida. A eldrita que a pesar de estar muy lejos siempre pendiente y presente en todos mis momentos. A ti Yenly por siempre sacarme una sonrisa, por ser la amiga de la infancia, fuiste eres y serás siempre el primer ejemplo de amiga que tuve. A mi Naivy gracias por demostrarme que esto ha llegado más lejos de lo que yo misma imagine, eres la amistad más inesperada que atesoro.

AGRADECIMIENTOS

A luisi por hacerme sentir en los peores momentos que mis problemas eran suyos también, por dejar a un lado sus cosas para dedicarse a mí, gracias por tener siempre la confianza de que todo saldría bien. Gracias por compartir esta travesía conmigo desde primer año y porque siempre fueran los que fueran los términos ser feliz al ayudarme y que tu objetivo siempre haya sido hacerme sonreír.

A mi familia de la UCI, mis SISMANAS. Mis niñas hablar de ustedes resulta imposible si se trata de tiempo y espacio. Han estado en tantos momentos que no alcanzaría agradecerles todo y por temor a no poder abarcar tanto como quisiera, quiero agradecerles a cada una por las razones que sé que me convierten en una persona afortunada por tenerlas en mi vida. A la caimana, eres la hermana hembra que no tuve y que tanto desee, la consejera más razonable y la más diferente a mí en cuestión de carácter creo que por todo esto hemos congeniado tan bien y si tuviera la oportunidad de escoger mis amistades sin duda te volviera a escoger, gracias por enseñarme tanto y por siempre estar ahí. A la lissy estas en cada uno nuestros momentos más divertidos participando como protagonista, gracias por ser quien calma las tempestades, por ser siempre tan divertida y por equivocarte tan poco cuando a consejos nos referimos. A la miti que a pesar de no haber estado en una parte de este camino retomaste la travesía y nos diste el privilegio de compartirla con nosotras, gracias por tu cariño y por ser una bella persona.

A ti mi gordi porque sé que donde quiera que estés me cuidas y que estas muy orgullosa de mí, gracias por haberte convertido en la mejor amiga que una persona puede tener, por ser cómplice, confidente y hermana, y aunque la vida nos haya

robado la oportunidad de hacer perdurar esta amistad por los años eso solo significa que no estaremos juntas físicamente.

A mi familia en especial a mi hermano Fabio. Hoy hemos crecido y no imagino que nuestra relación hubiera sido otra distinta a la que tenemos, gracias por ser mi mejor amigo y por hacerme saber que siempre puedo contar contigo para lo que sea, por apoyarme y defenderme tanto, pero también por no malcriarme y hacerme saber en otras ocasiones que no tengo la razón, gracias por considerarme tu amiga y a pesar de ser tan introvertido confiarme tus cosas y tener en cuenta mis consejos.

A las estrellas de mi universo, mis padres, gracias por todo, por haberme encaminado a convertirme en la persona que soy hoy, por apoyarme, confiar en mí, malcriarme en ocasiones, desear que me pase siempre lo mejor del mundo. Mami gracias por ser la mejor amiga que tengo, por escucharme y aconsejarme siempre, por confiar en mí y contar conmigo. Papi gracias por hacerme sonreír siempre y por ser tu cucuyuca, eres un ejemplo a seguir. Gracias por ser mis padres y dejarme ser su hija, cada persona debería tener el privilegio de contar con unos padres como lo son ustedes, los amo inmensamente.

RESUMEN

En la actualidad los sistemas de gestión son herramientas dedicadas a la optimización de los procesos de negocio. Ellos permiten un desarrollo sostenible en la gestión de políticas y procedimientos, que garantizan a la organización cumplir determinadas metas.

La presente investigación está encaminada a solucionar los problemas referentes al proceso de gestión de la Guardia Obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas; a partir de un sistema que permita la planificación automática de la misma. Para dar cumplimiento al objetivo se plantean los conceptos que fundamentan la investigación, se realiza una revisión de los antecedentes sobre la gestión de la información en los servicios de guardia a nivel nacional e internacional. Se describen las herramientas utilizadas, el CMS sobre el que se implementó el sistema y los elementos necesarios para su construcción. Como resultado del mismo se obtuvo una planificación ágil y más organizada, logrando a su vez mayor rapidez en la gestión de la guardia obrera.

Palabras clave: control, gestión de la guardia, planificación, procesos, sistemas de gestión,

ÍNDICE

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Desde épocas tempranas el ser humano ha tenido la necesidad de cuidar sus bienes. A medida que la sociedad fue creciendo y evolucionando, fue inminente establecer un control más estricto sobre dichos bienes. Esta situación llevo al hombre a perfeccionar sus métodos, técnicas, herramientas y formas de proceder para velar por la protección de sus recursos. Es entonces que, en el año 27 antes de Cristo (a.c), se crea en el imperio romano la Primera Guardia, con el propósito de mantener el orden y proteger la integridad física de sus dirigentes, marcando así, el nacimiento de los primeros sistemas de seguridad (Jones Pérez, y otros, 2016).

La seguridad hoy en día es un término estrechamente relacionado a la protección de la propiedad, ya sea privada o no. Las instituciones, son un escenario donde la seguridad juega un papel indispensable para dar protección a sus recursos y velar por la integridad de todos sus componentes internos, preservando por encima de todo, la vida humana (Flores, 2011).

De acuerdo al nivel de protección que se le quiera dar a los recursos en cuestión se utilizan cámaras de seguridad, cajas fuertes, sistema de alarmas, antivirus. Otra manera que se tiene para garantizar la integridad de los bienes es mediante la contratación de personal apto para desarrollar dicha tarea. La actividad que realizan estas personas es conocida comúnmente bajo el término “guardia de seguridad” (Bravo, 2016).

Cuba, es un país que a pesar de no poseer la mayor infraestructura tecnológica ni los más modernos sistemas de vigilancia, ha logrado crecerse ante las dificultades y ha puesto en práctica un grupo de medidas y acciones en aras de garantizar la seguridad de sus organizaciones. En las instituciones del país se pueden encontrar diversas medidas y métodos de seguridad, incluyendo las guardias y dentro de estas, siendo a su vez la más empleada, la guardia obrera (GO) (Betancourt, 2018).

La Universidad de las Ciencias Informáticas (UCI) es una institución que para el cumplimiento de sus objetivos posee un gran número de bienes y recursos en su mayoría tecnológicos. La misma emplea como una de sus medidas para garantizar la seguridad y protección de estos recursos, la GO, que comprende entre sus procesos principales la planificación y el control de la misma. Dentro de las áreas que forman parte del campus universitario de la UCI, se encuentra la Facultad 1 en la que el vicedecanato de administración y economía tiene a su cargo la gestión de la GO para los centros y departamentos que

radican en el edificio docente de esta facultad, este proceso es realizado por el vicedecano de economía y administración (VDEA).

Llevar a cabo la gestión de la GO en la Facultad 1 es un trabajo engorroso y requiere de mucho tiempo y esfuerzo. Esto se debe a la cantidad de personal involucrado en la realización de la guardia es elevada y tanto el proceso de planificación, como el de control realizado una vez que concluye la guardia, se hacen de forma manual mediante hojas de cálculo o en formato duro.

Para la planificación, esta situación trae consigo que en muchas ocasiones el VDEA pueda pasar por alto que algún trabajador haya quedado sin planificación de la guardia, teniendo que replanificar la misma. También si la planificó para la misma persona más de una vez en el mismo período el mismo debe replanificar todo el proceso de guardia. También se dificulta que se pueda llevar constancia si a todos los trabajadores se les planificó la guardia con igual diferencia de días entre una y otra y no exista periodicidad.

Otro aspecto que incide durante el proceso de planificación es cada vez que un trabajador necesita ausentarse del centro por un largo período o de forma permanente, el VDEA debe replanificar toda la guardia, y todas las veces de forma manual. Algo similar ocurre cuando a un trabajador se ausenta a su guardia por causa injustificada o necesita cambiarse para otro equipo de guardia, todo el proceso debe ser replanificado.

Por otra parte, al trabajador que recibe el documento *Excel* con la planificación, debe buscar en todo el documento su nombre para saber qué día le corresponde hacer la guardia y en qué turno. Además, al ser un documento lo que contiene la planificación, el trabajador no tiene la posibilidad de recibir notificaciones sobre la proximidad de su guardia en fechas cercanas al día de realización la misma y en ocasiones el trabajador olvida la guardia y por lo tanto incumple con la misma y esta deba ser replanificada.

Para controlar la realización de la GO, el VDEA recepciona toda la documentación del cumplimiento de la misma, esta puede estar en formato duro (en una hoja o en las plantillas impresas creadas para estos fines); o en formato digital las cuales recibe vía correo o personalmente a través de dispositivos de almacenamiento portables. Esta situación ocasiona que la información referente al cumplimiento de la GO no esté en el poder del VDEA con suficiente tiempo lo que entorpece la emisión del parte diario que debe entregar al consejo de dirección, dificultando el análisis de la guardia y la toma de decisiones respecto al proceso. Por lo anteriormente descrito no se pueden generar reportes estadísticos sobre el cumplimiento de la GO en la facultad.

Por lo anteriormente descrito se define como **problema de investigación** ¿Cómo contribuir al proceso de gestión de la GO en la Facultad 1 de la Universidad de las Ciencias Informáticas?

El **objeto de estudio** de la presente investigación va dirigido al proceso de gestión de la guardia obrera, y el **campo de acción** al proceso de gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas.

Para dar solución al problema definido se plantea como **objetivo general** desarrollar un sistema informático que contribuya al proceso de gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas.

Para guiar el cumplimiento del objetivo planteado, se formulan las siguientes **preguntas científicas**:

1. ¿Cuáles son los referentes teóricos relacionados con el uso de herramientas informáticas empleadas en el desarrollo del sistema de gestión de la GO en la Facultad 1 de la UCI?

2. ¿Cuál es el estado actual de las herramientas informáticas estudiadas para el desarrollo del sistema de gestión de la GO en la Facultad 1 de la UCI?

3. ¿Qué elementos deben tenerse en cuenta para llevar a cabo el análisis y diseño del sistema de gestión de la GO en la Facultad 1 de la UCI?

4. ¿Cómo materializar, en términos de componentes y código fuente, los elementos especificados para el sistema de gestión de la GO en la Facultad 1 de la UCI?

5. ¿Qué resultados se obtendrán al validar, a través de una estrategia de pruebas de software, el sistema de gestión de la GO en la Facultad 1 de la UCI?

Para dar respuesta a las preguntas científicas formuladas, es necesario dar cumplimiento a las siguientes **tareas de investigación**:

1. Sistematización de los referentes teóricos que sustentan la investigación, relacionados con el uso de las herramientas informáticas para la gestión de la GO en la Facultad 1 de la UCI.
2. Análisis del estado actual de las herramientas informáticas para la gestión de la GO en la Facultad 1 de la UCI.
3. Análisis y diseño de las funcionalidades del sistema de gestión de la GO en la Facultad 1 de la UCI.

4. Implementación de las funcionalidades del sistema de gestión de la GO en la Facultad 1 de la UCI.
5. Validación, a través de una estrategia de pruebas, del sistema para la gestión de la GO en la Facultad 1 de la UCI.

Los **métodos de investigación** utilizados para dar solución a la presente investigación son:

Teóricos:

Histórico-Lógico: permite estudiar la trayectoria histórica real del fenómeno y las tendencias del uso actual de sistemas de gestión de guardia con el fin de seleccionar las más apropiadas para darle cumplimiento al objetivo general de la investigación.

Analítico-Sintético: permitió realizar el estudio teórico de la investigación facilitando el análisis de documentos y la extracción de los elementos más importantes relacionados con el proceso de desarrollo de aplicaciones web que hacen posible la elaboración de conclusiones relacionadas con el proceso de gestión de la GO.

Modelación: permitió modelar a través de diagramas la estructura de los procesos referentes a la planificación y control de la GO.

Empíricos:

Entrevista: Se emplea en encuentros con el cliente para conocer la necesidad del desarrollo del sistema de gestión de la GO para la Facultad 1 de la UCI, definir sus funcionalidades e identificar a la vez particularidades de cada usuario y las restricciones que se imponen.

Observación: Posibilita obtener conocimiento acerca del funcionamiento de los sistemas existentes en la actualidad para proponer una mejor solución al proceso de realización del sistema de gestión de la GO para la Facultad 1 de la UCI.

La presente investigación está estructurada de la siguiente forma:

Capítulo1: “Fundamentación teórica del Sistema para la gestión de la GO en la Facultad 1 de la UCI”: En este capítulo se exponen los conceptos asociados al dominio de la investigación y se realiza un análisis de la teoría para el sistema de gestión de la GO para la Facultad 1 de la UCI. Además, se analizan algunas herramientas para el desarrollo del mismo y se define el entorno de desarrollo que se utilizará para dar solución al problema planteado en la investigación.

Capítulo2: "Análisis y diseño de la propuesta de solución del Sistema para la gestión de la guardia obrera en la Facultad 1 de la UCI": En este capítulo se documenta todo el proceso de elaboración del sistema de manera más detallada. Se describen los requerimientos de acuerdo a lo establecido en la metodología utilizada, diagramas de clases del diseño, diagramas de despliegue, lo referente al estilo arquitectónico del sistema, así como los patrones de diseño utilizados y el entorno de despliegue propuesto para desplegarlo.

Capítulo3: "Implementación y pruebas de la propuesta de solución del Sistema para la gestión de la guardia obrera en la Facultad 1 de la UCI": En este capítulo se detalla la propuesta de solución al problema planteado. Se describe la organización del módulo en un diagrama de componentes y se especifican los estándares de codificación a utilizar. Además, se realiza la estrategia de pruebas definida para el módulo y se muestran interfaces como parte del resultado final.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA DEL SISTEMA PARA LA GESTIÓN DE LA GO EN LA FACULTAD 1 DE LA UCI

1.1. Introducción

En el presente capítulo se declara la base conceptual que sustenta la investigación, se realiza un estudio de los sistemas homólogos, las herramientas y tecnologías con el objetivo de determinar las más factibles para implementar una solución informática que resuelva el problema de investigación.

1.2. Proceso de gestión de la GO en la Facultad 1

1.2.1. Proceso de planificación

El proceso de planificación de la guardia comienza cuando el VD de administración y economía toma el listado de trabajadores de un sistema de gestión de información[1], llamado KAINOS que es el utilizado por la Universidad (modelo oficial de recursos humanos), llamado P4. Luego se agrupan los trabajadores en dos categorías según la etapa del curso en que realizan la guardia: los profesores y especialistas en el período lectivo y los técnicos y trabajadores de servicio en el período no lectivo.

Una vez asignado cada personal a la etapa del curso que le corresponde, se conforman los equipos de guardia. En el caso de la Facultad 1, los equipos están compuestos por 2 integrantes los cuales pertenecen generalmente a la misma área: departamentos docentes o centros de producción. En el periodo lectivo los turnos de guardias se realizan en dos horarios, de manera que cada día realizan la guardia 4 trabajadores. Entre semana el primer turno es de 11:00 pm a 3:00 am y el segundo de 3:00 am a 7:00 am. Los fines de semana se realizan de 9:00 am a 1:00 pm y de 1:00 pm a 5:00 pm.

Luego de estar distribuidos todos los trabajadores y conformados los equipos, se realiza la planificación diaria de cada turno. Además, se tiene en cuenta, un equipo solo realiza la guardia un día del fin de semana

[1] Constituye un proceso mediante el cual se planifican, organizan, dirigen y controlan los recursos de información de una organización asegurando un adecuado tratamiento, intercambio y uso de este recurso de manera que contribuyan al establecimiento de fortalezas organizacionales, permitiendo que todos sus miembros dispongan de forma adecuada de los recursos informativos que existen en ella (2008).

cuando el resto de equipos también allá cubierto uno de estos días. Se garantiza así una distribución equitativa.

Para llevar a cabo una correcta planificación se debe tener en cuenta que existen trabajadores con determinada situación: mujeres embarazadas, personas con licencia sin sueldo, trabajadores con problemas personales, madres de niños pequeños. A estas personas, por lo general, se les planifica los turnos de guardia los fines de semana en el horario de la mañana.

Una vez culminada la planificación se circula el documento resultante a todos los trabajadores de la facultad. Suele ocurrir, la planificación se realice de forma trimestral, garantizando que los trabajadores conozcan con suficiente tiempo de antelación los días en que le corresponde realizar la guardia.

Comunicación de la planificación

Una vez realizada la planificación de la GO el VDEA de administración y economía envía un documento que contiene la información referente al proceso a todos los trabajadores y estos se encargan de buscar su nombre en todo el documento para saber qué día y en qué turno les corresponde hacer la guardia.

Proceso de replanificación

El proceso de replanificación tiene lugar en caso de existir algún cambio o afectación en la planificación realizada inicialmente. Los principales cambios están asociados a bajas, licencias sin sueldo, certificados médicos y otros. En estos casos, si existe previa comunicación con el VDEA, se replanifica la guardia de algún trabajador que no la haya realizado en el día que le correspondía por algún motivo. En la mayoría de los casos el VDEA no conoce estas situaciones, provocando la no realización de la GO en el docente. Como medida alternativa, se realiza el proceso de replanificación. Otro elemento importante, son las solicitudes de baja de los trabajadores, lo cual influye muy negativamente en la planificación de la GO de la Facultad el cual se ve afectado debido a que existirían turnos vacíos.

1.2.2. Proceso de control

La guardia es controlada por un directivo de la Universidad (oficial de guardia superior o rector en funciones), el cual realiza esta función de acuerdo a una planificación del centro. Esta persona tiene la función de tomar, durante las horas nocturnas, las decisiones respecto a la Universidad y de controlar el servicio de guardia

en ese momento. Come resultado de este proceso de control[2] durante varios horarios de la noche, se emite un parte al amanecer del día siguiente que es recibido por el decano de la facultad, quien a su vez lo envía al VDEA y es usado como contraparte de lo plasmado en la planilla de control de la guardia por parte de los trabajadores

1.3. Análisis de sistemas que gestionan procesos de guardia

En la actualidad existen aplicaciones para la gestión de los servicios de guardia. Con el objetivo de identificar ventajas y desventajas de estas, se realiza el siguiente análisis haciendo énfasis en las características y funcionalidades de este tipo de herramientas.

1.3.1. Sistemas homólogos a nivel internacional

Andes Airport Services

Es una solución web que consiste en una interfaz de captura de datos a la que se le ingresa información desde el área de RRHH. El proceso completo toma aproximadamente 20 días, depende en gran medida de las disponibilidades horarias de cada una de las personas involucradas, y no admite ningún tipo de administración o gestión de los roles generados. Además, esta solución no permite la obtención de reportes periódicos de eficacia, de cantidad de días libres utilizados o de sobre/baja utilización de la fuerza laboral disponible (CAMPO, 2013). Dicha solución corresponde a un sistema que permite realizar las asignaciones del personal considerando los turnos, cargos y restricciones legales, sindicales y de RRHH. Además, el sistema es capaz de entregar la asignación completa de cada trabajador para un mes dado, en tiempos de ejecución razonables. El resultado de esta asignación es editable y se acopla al proceso actual de entrega de turnos para los empleados de la empresa. Cabe hacer notar que la asignación de turnos propiamente tal, es realizada por un modelo matemático.

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

[2] Conjunto de actividades que se emprenden, para medir, examinar y evaluar la ejecución de los planes y los resultados obtenidos, con el fin de detectar y prever desviaciones, diagnosticando la razón de las desviaciones y tomando las medidas correctivas necesarias para asegurar la obtención de los objetivos (2008).

Es una solución completamente web, que permite obtener en tiempo real la información necesaria para la gestión de recursos humanos con un ahorro favorable del mismo y reduce los costes. Tiene como objetivo automatizar el control de horarios y flexibilizar calendarios para optimizarlos en función de las necesidades de trabajo de la empresa. No realiza la gestión de turnos y postas y tampoco incluye el control de la guardia. Esta herramienta fue seleccionada por las características de los sistemas web y por gestionar información. El sistema gestiona las incidencias producidas sobre el plan de horarios, ausencias y retrasos. Almacena la información necesaria por cada usuario. Permite crear calendarios anuales, mensuales, o de la cantidad de días deseados.

Incluye de forma estándar un módulo completo de planificación de calendarios, individuales o de grupo, optimizado para disponer de todos los datos visualmente y poder acceder a toda la información de forma rápida (Kriesi, 2015).

Tecno Hospital

Esta herramienta además de gestionar las guardias, informa las coberturas existentes para cada área de trabajo y controla la asistencia a los turnos. Informa detalladamente a cada miembro del equipo el turno asignado y la fecha del calendario. La aplicación es totalmente accesible desde Internet, lo que permite a los responsables del personal el manejo de la información desde cualquier parte del mundo, al igual que la consulta de sus trabajadores. La herramienta se comporta como un sistema de gestión web automático que permite planificar turnos en cualquier institución, contribuyendo con una reducción de tiempo favorable dedicado a esta tarea.

La herramienta genera automáticamente la planificación en base a parámetros definidos inicialmente como: área, categoría, número de horas máxima de guardia, o número mínimo y máximo de enfermeros necesarios por turno. Permite intercambiar de forma sencilla los turnos entre empleados, modificar horarios, establecer turnos personalizados o trabajar con turnos predefinidos. Permite consultar en todo momento si un turno está cubierto de acuerdo a los parámetros predefinidos y, en caso de faltar personal, propone una lista de trabajadores para cubrir dicho turno.

Este sistema integra dentro de sus funcionalidades la generación automática del portal de consulta para los trabajadores de cada área, donde pueden chequear sus turnos, horarios, guardias o calendario laboral, informa el número de horas mensuales y anuales que lleva acumuladas el trabajador de acuerdo a la

programación que se le ha establecido, cuenta con un control de incidencias laborales diarias, pero no permite escoger el periodo a planificar (Kolbe, 2015).

1.3.2. Sistemas homólogos a nivel nacional

Sistema de Guardia Estudiantil (GE) de la Facultad 3 de la UCI.

Este sistema se centra solamente en el proceso de planificación de la guardia estudiantil. No permite realizar permutas de guardia y tampoco tiene en cuenta el control de la misma (Mata Sanchez, et al., 2013). El sistema permite generar la planificación de la guardia, configurar las postas y los turnos. Realiza la planificación teniendo en cuenta criterios como: no repetir los turnos, planificar primero a los evaluados de M del mes anterior, planificar seguidamente a los que no hayan realizado la guardia y después al resto.

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

Sistema de Guardia de la Facultad 2 de la UCI.

Para el desarrollo de este sistema se utilizó como lenguaje de programación C#. Se centra solamente en el proceso de planificación de la guardia. Dentro de sus funcionalidades está la planificación automática de la guardia, la cual tiene en cuenta el turno realizado y asigna el turno más lejano posible a ese. No permite realizar permutas de guardia y tampoco tiene en cuenta el control de la misma. No notifica la planificación, pues es solo para uso administrativo, posee poca documentación y no está funcionando, pues no permite la gestión de los turnos postas y grupos de guardia (Gonzalez Guadalupe, y otros, 2015).

El análisis realizado sobre las aplicaciones homólogas al sistema de gestión de la GO en la Facultad 1 que se pretende desarrollar, permitió realizar un resumen (ver Tabla 1) teniendo en cuenta los siguientes parámetros:

- **Licencia:** se refiere al estado jurídico de la aplicación en cuanto a su uso, modificación y distribución, esta puede ser pública, en aquellas que no necesitan un pago para ser utilizadas; o privativa, en aquellas que si lo requieren.
- **Aplicación web:** se refiere a si la herramienta es una aplicación web o no.
- **Open Source:** especifica si la aplicación es de fuente abierta o código abierto.
- **Planificación:** se refiere a si la herramienta incluye el proceso.

- **Control:** se refiere a si la herramienta incluye el proceso.

Tabla 1. Análisis de los sistemas homólogos existentes.

Sistema	Licencia	Dominio de aplicación (web)	Open source	Planificación	Control
Andes Airport Services	Privativa	Si	No	q	No
Visual Time	Privativa	Si	No	No	No
Tecno Hospital	Pública	Si	No	Si	Si
Sistema de GE de la Facultad 3.	Pública	Si	Si	Si	No
Sistema de Guardia de la Facultad 2.	Pública	Si	Si	Si	No

Conclusión de los sistemas que gestionan procesos de guardia

Luego del análisis realizado, se puede concluir que de los sistemas analizados de la Universidad de las Ciencias Informáticas (UCI) ninguno cumple con todos los requerimientos definidos para la planificación y control de la GO, pues no abarcan los dos procesos definidos en el marco teórico de la investigación, aun así, pueden ser utilizadas algunas de las funcionalidades brindadas por ellos, tales como que todas son aplicaciones de dominio web. También para el desarrollo de la nueva propuesta de solución se tuvo en cuenta algunos de los requisitos funcionales de dichos sistemas. Por el contrario, los sistemas estudiados a nivel internacional aportan funcionalidades que si pudieran ser utilizadas en el desarrollo de la solución, pero privatizan el acceso a su código fuente y niegan el derecho a copiar interfaz de usuario para su posterior modificación, adaptándolos a los requerimientos del sistema en cuestión, por lo que no cumplen con el paradigma de independencia tecnológica por el cual está abogando el país. Por lo planteado anteriormente, ninguno de los sistemas analizados se toma como solución ya que no cumplen con los parámetros que se miden para considerarlo como tal. De esta manera se evidencia la necesidad de desarrollar una aplicación web que permita disminuir el esfuerzo y el tiempo invertidos en la ejecución del proceso de planificación de la GO.

1.4. Entorno de desarrollo de la propuesta de solución

1.4.1. Metodología de desarrollo de software

En ingeniería de software, una metodología de desarrollo constituye un conjunto de métodos, procedimientos, técnicas, herramientas y soportes documentales utilizados para estructurar, planificar y controlar el proceso de desarrollo de sistemas de información (PINZÓN, 2006).

Para la adecuada implementación de la propuesta de solución es necesario la selección de una metodología que guíe el desarrollo de la misma, para asegurar un producto de calidad. Se selecciona, en consecuencia, la metodología AUP-UCI teniendo en cuenta que es la metodología adaptada al ciclo de vida de los proyectos productivos de la Universidad, es ampliamente usada en el área de la UCI y es extremadamente flexible al proceso de desarrollo de software. AUP-UCI constituye una variante de AUP (Proceso Unificado Ágil, por sus siglas en inglés), elaborada a partir del modelo CMMI-Dev v1.3 que constituye una guía para aplicar las mejores prácticas en una entidad desarrolladora en busca de productos y servicios de calidad (RODRÍGUEZ SÁNCHEZ, 2014).

Esta metodología propone tres fases para el desarrollo del software (Inicio, Ejecución y Cierre) y siete disciplinas (Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas, Pruebas de liberación y Pruebas de aceptación) (RODRÍGUEZ SÁNCHEZ, 2014).

AUP-UCI propone, además, cuatro escenarios a utilizar para modelar el sistema en los proyectos. Para la presente investigación, se selecciona el escenario número 2, que aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades, de esta forma modelarían exclusivamente los conceptos fundamentales del negocio. Este escenario se emplea mayormente para proyectos donde el objetivo primario es la gestión y presentación de información.

1.4.2. Herramientas y tecnologías para el desarrollo

Las herramientas informáticas son programas, aplicaciones o simplemente instrucciones usadas para efectuar tareas de modo más sencillo (Yanover, 2016). Con el objetivo de minimizar los costos, se propone utilizar tecnologías y herramientas que permitan su uso sin necesidad de pago de licencias. A continuación, se hace un estudio y se comparan las herramientas para seleccionar las que se usarán en la solución propuesta.

Lenguaje de modelado

Lenguaje Unificado de Modelado (UML)

UML (Unified Modeling Language) fue adoptado como estándar del Object Management Group (Grupo Gestor de Objetos) en 1997 debido a que representa una colección de las mejores prácticas de ingeniería que han sido probadas con éxito en el modelado de sistemas. Es un lenguaje para la especificación, visualización, construcción y documentación de sistemas, no solo de *software*. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, pues ha sido diseñado para modelar cualquier tipo de soluciones informáticas, arquitectura o cualquier otra rama (Sarmiento, 2016).

Herramienta de modelado

Visual Paradigm for UML es una herramienta CASE[3] que soporta el ciclo de vida completo del desarrollo de *software*: análisis y diseño orientados a objetos, implementación y pruebas. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Visual Paradigm., 2015). Por las razones antes expuestas se emplea esta herramienta de modelado para la propuesta de solución en su versión 8 .0.

Herramienta de desarrollo

Según (Patrice Lamothe, 2009), al crear un sitio web o una aplicación web, lo primero que se debe decidir es qué herramientas utilizar. Se puede construir desde cero, pero no todos conocen lenguajes de programación como PHP o lenguajes de visualización como HTML y CSS. Los sistemas de gestión de contenido reducen la necesidad de tener esa experiencia, mientras que los *frameworks* (Marco de trabajo) proporcionan un conjunto de características que puede utilizar para crear una aplicación personalizada. Los *CMS* (*content managment system*) y los *frameworks* son muy diferentes entre sí.

[3] Del Inglés: *Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadora

Un *CMS* es una interfaz que puede controlar una o varias bases de datos en las que se encuentra el contenido del sitio web. Se puede administrar de forma independiente el contenido del diseño por lo que en cualquier momento es posible, con el mismo contenido, visualizarlo de forma distinta cambiando el estilo sin tener que cambiar el formato del contenido (Morell Álex, 2014).

Por su parte, *framework* hace referencia en la informática a una estructura conceptual con un soporte definido que sirve de base para el desarrollo de software. Puede tener soporte de programas y bibliotecas, además de otras herramientas para facilitar el desarrollo y la unión de las diferentes partes de un proyecto. En resumen, una serie de herramientas que nos permiten y facilitan el desarrollo de un programa, aplicación o página web (Mulet, 2018).

Luego de realizar una comparación se decide que la opción más adecuada para el desarrollo del sistema en cuestión es utilizar un *CMS* porque brinda ventajas tales como: es una aplicación con una base de datos asociada en la que se almacenan los contenidos, separados de los estilos o diseño. El CMS controla también quién puede editar y visualizar los contenidos, convirtiéndose en una herramienta de gestión integral para la publicación de sitios web (Patrice Lamothe, 2009).

Según (Schaferhoff, 2019) algunos de los *CMS* genéricos más utilizados en la actualidad son *Wordpress*(59.9%), *Joomla*(6.6%) y *Drupal*(4.6%), aunque existen diferencias notables entre ellos, comparten algunas características comunes pues todos son *software* libre y gratuito, se programan en PHP y pueden correr en un servidor Apache. Teniendo en cuenta que el *CMS* que se utiliza en el Centro de Ideoinformática (CIDI) es *Drupal* en su versión 7.66 se hace un análisis de esta herramienta como base de la propuesta de solución.

Sistema gestor de contenido

Drupal

Según (Olivares, 2016)[4] al igual que *WordPress*, *Drupal* es un *CMS*, que puede ser utilizado para administrar el contenido de una página web. *Drupal* presenta marcadas características que respaldan la decisión de

[4] Ingeniero de software. Trabaja en un marco específico basado en PHP. Emplea Propel como ORM (Mapeo relacional de objetos). Desarrollador de nuevos módulos, tareas de mantenimiento, sistemas de traducción, listas de control de acceso, servicio de tickets y soporte, API, terceros, funcionalidades de facturación,

utilizarlo como sistema gestor de contenidos en la propuesta de solución, además de ser con el que se desarrollan los proyectos en la Facultad:

1. Ofrece agilidad a los negocios

El tiempo entre los lanzamientos concuerda con la planificación estratégica. Esta es una ventaja para los desarrolladores de software pues *Drupal* añade una nueva funcionalidad más rápido de lo que se podría esperar con otros *CMS*. *Drupal* permite a los negocios adaptarse rápidamente a los cambios del mercado y el entorno de maneras más productivas y rentables. Además, la adaptabilidad permite alcanzar un rendimiento más alto de forma rápida y a costos más bajos.

2. Escalabilidad

Esta es otra de las ventajas de usar *Drupal* en lugar de *WordPress*. Actualmente *Drupal* es compatible con los sitios del mundo como el caso de Twitter, The Economist o Weather. Su escalabilidad le permite ser capaz de manejar los picos de tráfico regulares o un gran volumen de visitantes.

3. Capacidades de integración

Es quizás una de las ventajas de *Drupal* porque se incluye dentro del propio ecosistema del negocio. En la parte superior proporciona una adecuada manera de gestionar los contenidos y capacidades de marketing digital. Pero también es capaz de modelar los datos e integrar aplicaciones y servicios, haciendo que sea más fácil su adopción dentro del sistema que se desea desarrollar.

4. Contenido optimizado

Incluye herramientas para la gestión y optimización de palabras clave, informes de contenido, títulos de página, integración con Google Analytics, mapas de sitio entre otros. Se integra además con plataformas de medios de comunicación y soporta una gran variedad en los tipos de archivos.

---

sistema de plantillas específicas, aprovisionamiento de dispositivos de hardware, tareas de monitoreo, implementación de pruebas funcionales.

La selección de las herramientas que se mencionan a continuación está basada fundamentalmente en la selección de *Drupal* en su versión 7.66 como CMS a emplear durante el desarrollo de la propuesta de solución.

Servidor de aplicaciones web

Apache

Se escoge el servidor web Apache en su versión 2.2.21, debido a su alto nivel de configuración, robustez y estabilidad, lo que aporta confiabilidad en el proceso de desarrollo. Este servidor presenta características tales como: posibilidad de ejecución en diferentes sistemas operativos, tecnología gratuita de código abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto. Es un servidor altamente configurable de diseño modular. Es sencillo ampliar las capacidades del servidor web Apache, a partir de la disponibilidad de varios módulos adaptables a este. Se integra con lenguajes como Perl, PHP y otros lenguajes de script. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor, al configurar la ejecución de un determinado script cuando ocurra un error en concreto. Facilita la configuración en la creación y gestión de registros, de este modo se tiene un mayor control sobre lo que sucede en el servidor. (Gonzalez, 2014).

Sistema gestor de base de datos

MySQL Database Server es una base de datos de código fuente abierto. Su arquitectura lo hace rápido y fácil de personalizar. La reutilización del código dentro del software y la aproximación para producir características funcionales, ha dado lugar a un sistema de administración de la base de datos de alta velocidad, compactación, estabilidad y facilidad de despliegue (Heredia Hanza, y otros, 2010).

Dentro de sus características se encuentran:

- Acceso a las bases de datos de forma simultánea por varios usuarios y/o aplicaciones.
- Seguridad, en forma de permisos y privilegios, determinados usuarios tendrán permiso para consulta o modificación de determinadas tablas.
- Esto permite compartir datos sin que peligre la integridad de la base de datos o protegiendo determinados contenidos.
- Potencia: SQL es un lenguaje para consulta de bases de datos, usar un motor ahorra una enorme cantidad de trabajo.

- Portabilidad: SQL es también un lenguaje estandarizado, de modo que las consultas hechas usando SQL son portables a otros sistemas y plataformas. Esto, unido al uso de C/C++ proporciona portabilidad. (Heredia Hanza, y otros, 2010)

JQuery

JQuery es la biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM (*Document Object Model*), manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX (*Asynchronous Javascript and XML*) a páginas web. Es un *framework* de código abierto y se ha convertido en un producto con aceptación por los programadores debido a su seriedad, estabilidad, documentación y desarrolladores a disposición de su mejora y actualización, permite su uso de manera gratuita en proyectos libres y privados ya que tiene licenciamiento doble bajo la licencia MIT y la Licencia Pública General de GNU v2. La manipulación de este *framework* permite lograr resultados en menos tiempo y espacio. Algunas de las principales razones por las que fue seleccionado para ser utilizado en la propuesta de solución son que permite la selección y manipulación de elementos HTML y CSS, ofrece compatibilidad en la programación en JavaScript con distintos navegadores, en la solución son visibles estos efectos y animaciones (Gutiérrez, 2009). Se pretende utilizar con la versión 1.10.2.

Herramientas de validación

Las pruebas de validación en la ingeniería de *software* son el proceso de revisión que verifica que el sistema de *software* producido cumple con las especificaciones y logra su cometido. La validación es una parte del proceso de pruebas de *software* de un proyecto, que también utiliza técnicas tales como evaluaciones, inspecciones y tutoriales. Es el proceso de comprobar que las especificaciones del usuario fueron cumplidas (PRESSMAN, 2002). Para realizar las pruebas de validación a la solución propuesta se emplearán las herramientas Acunetix 8.0 y JMeter 3.1.

Acunetix

Acunetix es una herramienta de seguridad de aplicaciones web automatizada. Es capaz de escanear cualquier sitio Web o aplicación Web que es accesible a través del protocolo HTTP / HTTPS (*HiperText Transfer Protocol*). Acunetix también proporciona herramientas de pruebas de penetración manuales que aumentan y contribuyen a las pruebas automatizadas, así como ayudar con la prueba de vulnerabilidades lógicas (Ferran, 2016).

JMeter

La aplicación JMeter de Apache es un *software* de código abierto, una aplicación Java diseñada para cargar el comportamiento funcional y medir el rendimiento. Originalmente fue diseñado para probar aplicaciones web, pero se ha expandido a otras funciones de prueba. Apache JMeter puede emplearse para probar el rendimiento en recursos estáticos y dinámicos, aplicaciones dinámicas en la Web. Se puede utilizar para simular una carga pesada en un servidor, grupo de servidores, red u objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga. Las características de Apache JMeter incluyen: Capacidad para cargar y realizar pruebas de rendimiento en diferentes tipos de aplicaciones / servidor / protocolo. Fácil correlación mediante la capacidad de extraer datos de formatos de respuesta más populares, HTML, JSON (*JavaScript Object Notation*), XML (*Extensible Markup Languajes*) o cualquier formato textual, portabilidad completa, almacenamiento en caché y análisis *offline*/ reproducción de los resultados de las pruebas, núcleo extensible (ApacheJMeeter, 2016).

1.4.3. Lenguajes de Programación

Un lenguaje de programación según (Gervacio, 2018) consiste en un idioma artificial diseñado para expresar computaciones que pueden ser llevadas a cabo por máquinas como las computadoras. Estos suelen usarse para crear programas que controlen el comportamiento físico y lógico de una máquina y para expresar algoritmos con precisión. A continuación, se describen los lenguajes de programación a utilizar en el desarrollo de la propuesta de solución:

Lenguajes del lado del cliente

Lenguaje HyperText Markup Language (HTML)

Lenguaje de publicación especificado como un estándar por el W3C (*World Wide Web Consortium*) que permite la creación de páginas web. Inicialmente fue presentado por Tim Berners-Lee que propuso un sistema basado en hipertexto para el intercambio de información en la Web. La aparición del lenguaje influyó notablemente en el crecimiento de Internet, dónde la información era distribuida mediante colecciones fragmentadas de textos, imágenes y sonidos. HTML es independiente de la plataforma utilizada y se basa fundamentalmente en el uso de etiquetas estructurales y semánticas, adecuadas para la creación de documentos relativamente simples que permiten simplificar su estructura (Jose, y otros, 2018).

Algunas de sus características son:

- Estructura del cuerpo: permite agrupar todas estas partes de una web en nuevas etiquetas que representarán cada una de las partes típicas de una página.
- Etiquetas para contenido específico: utiliza etiquetas específicas para cada tipo de contenido en particular, como audio, video, entre otros.
- Bases de datos locales: permite el uso de una base de datos local, con la que se puede trabajar en una página web por medio del cliente y a través de un API[5].
- Aplicaciones web *offline*: permite la creación de aplicaciones web que funcionen sin necesidad de estar conectados a internet.

Por lo anteriormente descrito se decide trabajar con dicha herramienta en su versión 5**.**

Java Script (JS)

JavaScript es un lenguaje de programación que no forma parte de la plataforma Java; comparte algunos atributos con el lenguaje de programación Java, pero se desarrolló de forma independiente. Es un lenguaje informático interpretado que permite incluir macros en páginas web. Estas macros se ejecutan en el ordenador del visitante de las páginas, y no en el servidor (porque los servidores web suelen estar sobrecargados, mientras que las PC's de los usuarios no suelen estarlo) (Valdés, 2007).

Se escoge JavaScript en su versión 1.8.5 para el desarrollo de la presente solución informática porque proporciona los medios para:

- Controlar las ventanas del navegador y el contenido que muestran.
- Programar páginas dinámicas simples.
- Evitar depender del servidor Web para cálculos sencillos.
- Capturar los eventos generados por el usuario.
- Optimizar los tiempos de carga y el tráfico del servidor.
- Comprobar los datos que el usuario introduce en un formulario antes de enviarlos.

5 Interfaz de programación de aplicaciones

Lenguaje Cascading Style Sheets 3 (CSS)

CSS es un lenguaje que describe la presentación de los documentos estructurados en hojas de estilo utilizado para especificar el aspecto de una página web. Se basa en reglas que rigen el comportamiento del estilo de los elementos.

Por lo anteriormente descrito se decide trabajar con dicha herramienta en su versión 3.

Bootstrap

Es una herramienta que permite crear interfaces de usuario, además, *bootstrap* dispone de numerosas herramientas necesarias para diseñar cualquier tipo de sitio web utilizando los estilos y elementos de sus librerías (Mauricio, 2017).

Se hará uso de *bootstrap* en su versión 3.0 por permitir crear interfaces que sean adaptables a cualquier navegador, incluido internet explorer usando *HTML Shim* (Romero, 2016), de igual manera a equipos de escritorio, *tablets* y móviles a distintas escalas, permitiendo ser usado de forma muy flexible para desarrollo *web* con excelentes resultados. Permite agregar imágenes responsive, es decir, con solo insertar la imagen cuya clase sea *"img-responsive"* las imágenes se acoplarán al tamaño. Se integra perfectamente con las principales librerías javascript, por ejemplo, jquery.

Lenguaje del lado del servidor

Lenguaje Hypertext Pre-processor (PHP)

Lenguaje de alto nivel con técnicas de Programación Orientada a Objetos, multiplataforma, sencillo de usar, rápido, integrable, además de ser de *software* libre. PHP es uno de los lenguajes de programación que permite programar *scripts* del lado del servidor, insertados dentro del código HTML, con una variedad de funciones y documentación. PHP es orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. La programación en PHP es segura y confiable debido a que el código PHP es invisible al navegador y al cliente, el servidor es el encargado de ejecutar el código y enviar su resultado HTML al navegador (Castillo, 2006).

Entre las características por las cuales se selecciona este lenguaje se encuentran:

- *Drupal* está implementado en PHP.
- Multiplataforma.

- Presenta soporte para bases de datos como: MySQL y PostgreSQL.

Por lo anteriormente descrito se decide trabajar con dicha herramienta en su versión 7.

1.5. Conclusiones del capítulo 1

En este capítulo se han abordado los elementos teóricos que dan sustento a la propuesta de solución del problema planteado, en tal sentido se puede arribar a las siguientes conclusiones:

1. La definición de los principales conceptos asociados al dominio de la presente investigación y las relaciones entre estos, permitió alcanzar una mayor comprensión de los procesos asociados a la gestión de la guardia obrera.

2. El análisis de los principales elementos de la teoría en función de la gestión del proceso de guardia obrera, permite definir cómo deben ser tratados los mismos desde la propuesta de solución.

3. El análisis de los sistemas homólogos, permite identificar las tendencias en cuanto al desarrollo de herramientas informáticas para el sistema de gestión de la GO.

4. El análisis de la documentación de los sistemas de gestión de la GO y el resto de los sistemas similares, sobre la metodología de desarrollo, así como las herramientas, tecnologías y lenguajes de programación utilizados en su implementación, permiten especificar el ambiente de desarrollo para la propuesta de solución.

CAPÍTULO 2. ANÁLISIS Y DISEÑO DEL SISTEMA PARA LA GESTIÓN DE LA GUARDIA OBRERA EN LA FACULTAD 1 DE LA UCI

2.1. Introducción

En el proceso de creación de un software es importante el entendimiento del negocio para implementar una solución adecuada. En el presente capítulo se describirá el sistema a desarrollar, a partir del estudio de los procesos actuales del negocio, la descripción y el modelado de los mismos, con el objetivo de comprender el funcionamiento y llevar a cabo una correcta implementación de dicho sistema. Además, se especifican los aspectos funcionales del sistema que se propone construir, así como los diferentes artefactos asociados a la metodología de desarrollo.

2.2. Descripción de la propuesta de solución

La propuesta de solución a desarrollar surge fundamentalmente por la necesidad de automatizar el proceso dentro de la facultad. Debe contar con características tales como, llevar a cabo la planificación de la guardia para cada una de las áreas pertenecientes al docente, permitir planificar tanto a los equipos de guardia como a trabajadores por individual, crear los equipos de guardia de manera fácil con la información referente a cada trabajador y notificar (vía correo) al mismo, tanto de la planificación de su turno de guardia, como de la creación de los equipos, registrar incidencias de los trabajadores asociadas a ausencias o incumplimientos de la realización de la guardia, estas incidencias deberán ser insertadas en el sistema manualmente por el administrador del mismo, generar reportes estadísticos referentes a varios aspectos relacionados con el proceso y con la propia información almacenada en las incidencias. La propuesta mostrará los datos de todos los trabajadores de la facultad que realicen guardia, esto será posible mediante la importación desde un documento Excel conjuntamente con un servicio web de la propia Universidad.

Para lograr esto será desarrollado un sistema web, que brinda las facilidades de trabajar con esta información de forma fácil tanto para el administrador del sistema como para el acceso del usuario (trabajador) al sistema. Una vez creada la propuesta el usuario tiene la facilidad de poder acceder al sistema para consultar la información y el administrador de optimizar el trabajo mediante la importación de un documento previamente elaborado que contiene los datos de los trabajadores conjuntamente con el consumo de un servicio web de la propia Universidad.

2.2.1. Requisitos de la propuesta de solución

Los requerimientos para un *software* son las descripciones de lo que el sistema debe hacer, el servicio que ofrece y las restricciones en su operación. Tales requerimientos reflejan las necesidades de los clientes por un sistema que atiende cierto propósito (Sommerville, 2011). Estos se dividen en:

Requisitos funcionales

Los requisitos funcionales son declaraciones de las funcionalidades que debe cumplir el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (PRESSMAN, 2002).

Tabla 2. Requisitos funcionales

Requisitos funcionales	Nombre del requisito funcional	Prioridad	Complejidad
RF1	Adicionar trabajador	alta	baja
RF2	Editar trabajador	alta	baja
RF3	Mostrar trabajador	alta	baja
RF4	Eliminar término de taxonomía	baja	baja
RF5	Eliminar contenido	baja	baja
RF6	Adicionar equipo de guardia	alta	baja
RF7	Editar equipo de guardia	alta	baja
RF8	Mostrar equipo de guardia	alta	baja
RF9	Adicionar término de taxonomía turno	baja	baja
RF10	Editar término de taxonomía turno	baja	baja
RF11	Listar términos de taxonomía turno	baja	baja
RF12	Adicionar término de taxonomía período-horario	baja	baja
RF13	Editar término de taxonomía período-horario	baja	baja
RF14	Listar términos de taxonomía período-horario	baja	baja
RF15	Adicionar usuario del sistema	alta	baja
RF16	Editar usuario	alta	baja
RF17	Mostrar datos de usuario	alta	baja
RF18	Listar usuario	alta	baja

RF19	Filtrar usuario	alta	baja
RF20	Autenticar usuario	alta	baja
RF21	Cambiar estado de actividad de usuario	alta	baja
RF22	Cancelar cuentas de usuario	alta	baja
RF23	Adicionar roles	alta	baja
RF24	Editar roles	alta	baja
RF25	Listar roles	alta	baja
RF26	Cambiar asignación de permisos	alta	media
RF27	Adicionar planificación guardia	alta	alta
RF28	Mostrar planificación guardia	alta	alta
RF29	Editar planificación guardia	alta	alta
RF30	Exportar planificación guardia	alta	alta
RF31	Enviar planificación guardia por correo	alta	alta
RF32	Imprimir planificación guardia	alta	alta
RF33	Replanificar guardia	alta	baja
RF34	Notificar por correo proximidad de la guardia	alta	alta
RF35	Notificar por correo planificación de guardia	alta	alta
RF36	Notificar por correo confección de equipo de guardia	alta	baja
RF37	Adicionar incidencia	alta	baja
RF38	Editar incidencia	alta	baja
RF39	Mostrar incidencia	alta	baja
RF40	Adicionar reporte	alta	baja
RF41	Mostrar reporte	alta	baja
RF42	Importar trabajadores	alta	alta

Requisitos no funcionales

Los requisitos no funcionales representan características generales y restricciones de la aplicación o sistema que se esté desarrollando. Son aquellos que no se refieren directamente a las funciones

específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. Suelen expresarse de una manera general y sin hacer referencia a algún modulo, transacción o característica del sistema (Herreros, 2015).

Requisitos de usabilidad

RnF 1.1: El sistema para la gestión de la GO en la Facultad 1 debe ser una aplicación web.

RnF 1.2: La aplicación debe presentar una interfaz agradable e intuitiva.

Requisitos de confiabilidad

RnF 2.1: El sistema debe ser tolerante a fallos, y mostrar solo la información necesaria para orientar al usuario.

Requisitos de eficiencia

RnF 3.1: El sistema debe permitir que los usuarios interactúen con él de manera concurrente.

RnF 3.2: El tiempo de demora de una petición al servidor debe ser menor de cinco (5) segundos aproximadamente.

Requisitos de soporte

RnF 4.1: El sistema debe contar con toda la documentación definida en el expediente de proyecto asociada a su proceso de desarrollo para las actividades de soporte.

Requisitos de restricciones de implementación y diseño

RnF 5.1: El componente debe ser desarrollado en su totalidad con tecnologías de código abierto**.**

Requisitos de interfaz

RnF 6.1: La interfaz de usuario se debe guiar por la vista de arquitectura de presentación definidas en el documento “Pautas de diseño XABAL-SGC”.

Requisitos de software

RnF 7.1: Para el uso del sistema se requiere una PC cliente con cualquier sistema operativo, que se pueda instalar navegadores web para el uso de la aplicación.

RnF 7.2: La comunicación entre la PC cliente y el servidor de aplicaciones web se realiza a través del protocolo HTTPS.

Requisitos de hardware

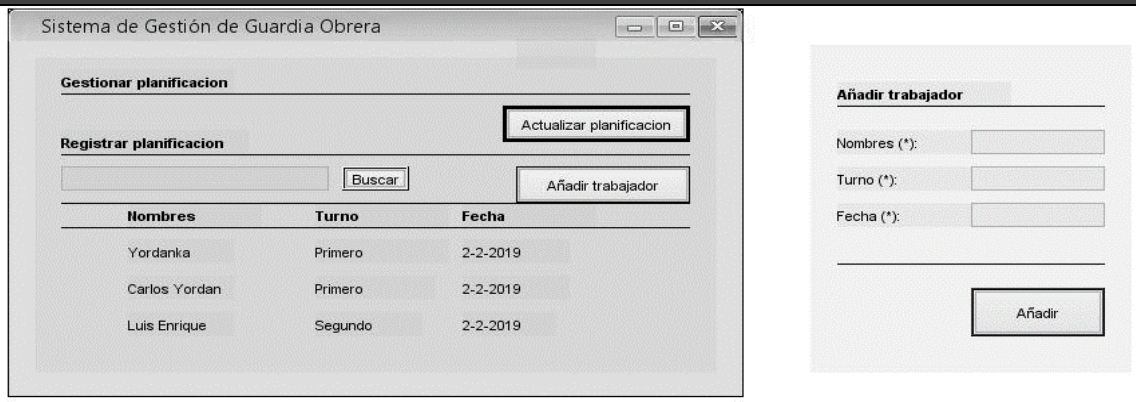
RnF 8.1: Teniendo en cuenta que en la computadora va a ejecutarse un servidor web, se requiere como mínimo 80gb de disco duro, tarjeta de red de 100 MB, un procesador Pentium 4 y 4 GB de RAM para que la aplicación funcione correctamente.

2.2.2. Descripción textual caso de uso Gestionar planificación

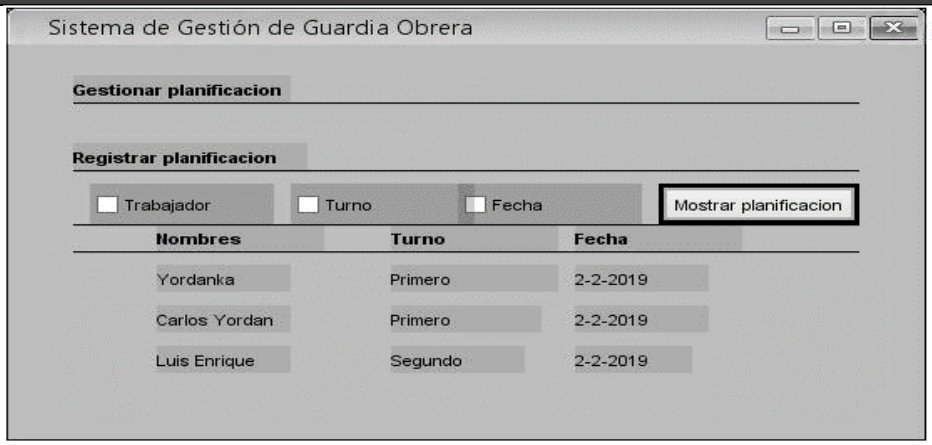
Tabla 3 Descripción del CU Gestionar planificación

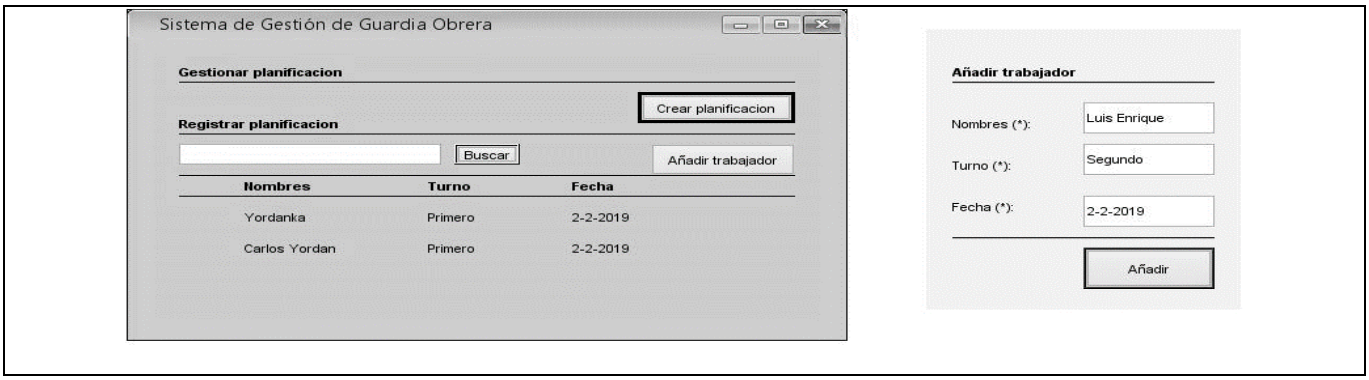
Objetivo	Permitir mostrar, actualizar y crear datos acerca de la planificación.	
Actores	Administrador.	
Resumen	El caso de uso se inicia cuando el administrador del sistema decide mostrar, actualizar o crear datos acerca de un usuario.	
Complejidad	Alta.	
Prioridad	Alto.	
Precondiciones	Administrador ya autenticado.	
Post condiciones	Se mostró, actualizó o se creó usuario(s).	
Flujo de eventos		
Flujo básico “Gestionar planificación”		
Actor		**Sistema**
1.	Selecciona de la página principal la opción “Gestionar planificación”.	
2.		Muestra una pantalla con un listado de usuarios y permite Mostrar, Actualizar o Crear la planificación.
3.	Desea mostrar, actualizar o crear la planificación.	
4.		Da la posibilidad de realizar alguna de las siguientes acciones: a) Si decide mostrar la planificación, ir a la sección “Mostrar Planificación”. b) Si decide actualizar los datos de la planificación, ir a la sección “Actualizar planificación”. c) Si decide crear la planificación, ir a la sección “Crear planificación”.
Flujos Alternos		
2a. Listado de usuarios vacío		
Actor		**Sistema**
1.		Carga una pantalla en blanco sin usuarios registrados.
Sección 1: “Actualizar planificación”		
Flujo básico Añadir trabajador		

Actor		Sistema
1.	Presiona el botón “Añadir trabajador”.	
2.		Muestra una ventana con los siguientes campos a introducir: • Nombre(s) • Turno • Fecha Y el botón “Añadir”.
3.	Introduce los datos y presiona el botón “Añadir trabajador”.	
4.		Verifica que todos los campos estén llenos.
5.		Verifica que los datos introducidos estén correctos.
6.		Verifica que este trabajador no exista.
7.		Almacena los datos del trabajador. *Finalizando así el caso de uso.*
Prototipo elemental de interfaz gráfica		



Flujos Alternos		
4a. Campos vacíos		
Actor		**Sistema**
1.		Muestra el mensaje “Campos vacíos”.
5a. Datos incorrectos		
Actor		**Sistema**
1.		Muestra el mensaje “Datos incorrectos”.
Sección 2: “Mostrar planificación”		
Flujo básico Mostrar planificación		
Actor		**Sistema**
1.	Marca la opción “Mostrar planificación.	
2.		Muestra la interfaz con los datos de la planificación: • Nombre • Turno

		• Fecha Y el botón “Mostrar Planificación”. Y los *check box* para seleccionar por el elemento que se desea mostrar: Trabajador, turno, fecha
Prototipo elemental de interfaz gráfica		
		
Sección 3: “Crear planificación”		
Flujo básico Crear planificación		
Actor		**Sistema**
1.	Marca la opción “Crear planificación”	
		Muestra una ventana con los siguientes campos a introducir: • Nombre(s) • Turno • Fecha Y el botón “Añadir”.
2.	Introduce los datos y presiona el botón “Añadir”.	
		Verifica que todos los campos estén llenos y crea la planificación.
Prototipo elemental de interfaz gráfica		



2.3. Análisis y diseño

Modelado del dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés (LARMAN, 2017). Para lograr un mejor entendimiento de los procesos que requieren informatización, se realizó un modelo conceptual (ver Figura 1), con un total de diecinueve (19) clases y veinticinco (25) relaciones, el cual representa y relaciona los conceptos más importantes dentro del contexto del negocio.

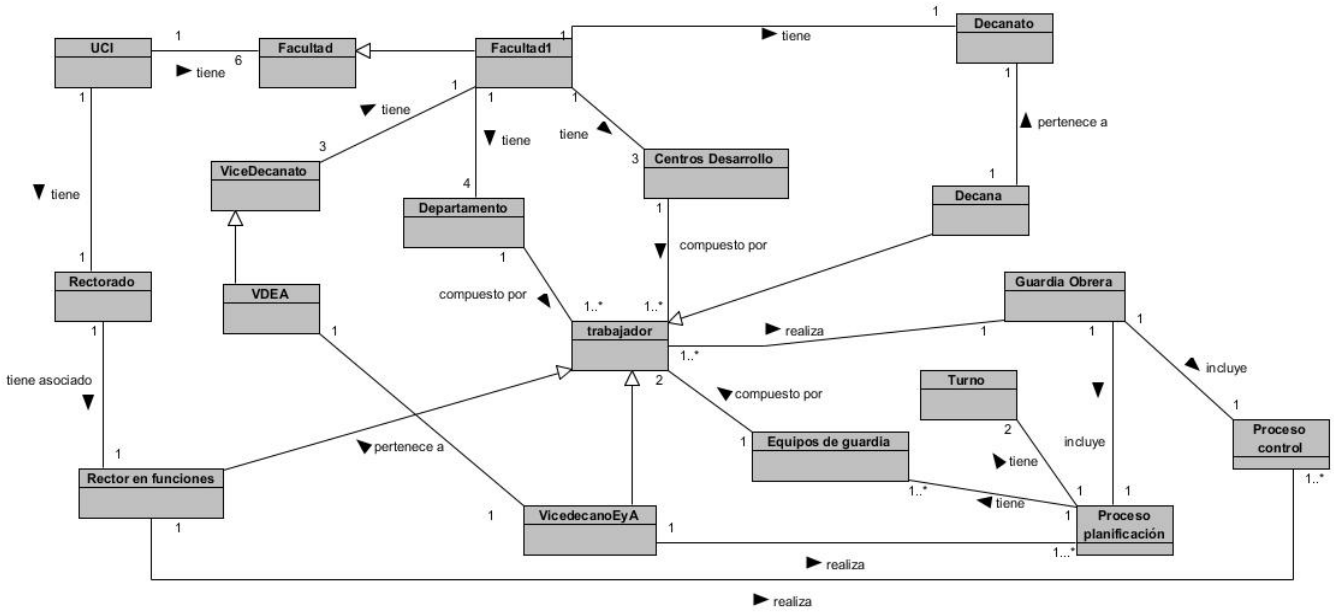


Figura 1. Modelo conceptual

Diagrama de caso de uso del sistema

Un caso de uso captura un contrato que describe el comportamiento del sistema en diferentes condiciones mientras este responde a la petición de uno de sus usuarios (Cockburn, 2008)**.**

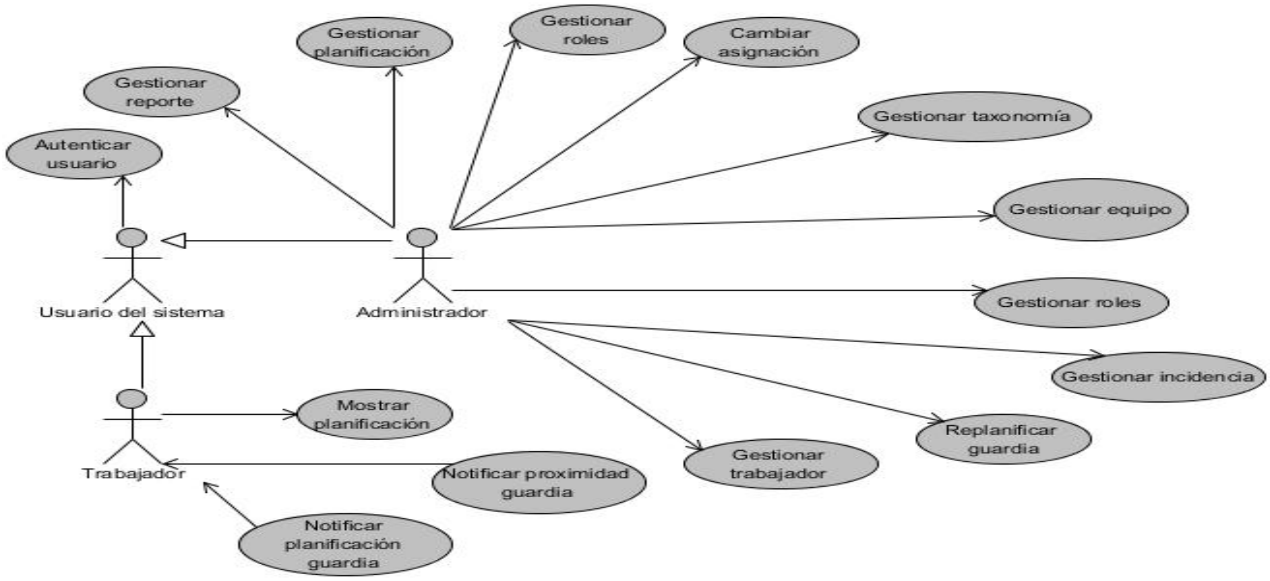


Figura 2. Diagrama de Caso de Uso del Sistema

Descripción de la arquitectura de software y los patrones de diseño

Arquitectura de software

Al utilizarse el *CMS Drupal* para el desarrollo de la propuesta de solución, la arquitectura de *software* a utilizar es la que este define, la cual es una arquitectura n-capas según (VANDYK, 2011), dividida específicamente en 5 capas que son descritas a continuación:

Plantillas (*templates*): Esta capa establece la apariencia gráfica que se le muestra al usuario. Esta separación entre la información y los estilos permite cambiar la apariencia del portal web sin necesidad de modificar los contenidos. En la propuesta de solución esta capa contiene el tema ***Sistema para la gestión de la guardia obrera***, así como su codificación en el lenguaje CSS, HTML y PHP.

Permisos de usuario: Es la capa que contiene los roles y permisos de los usuarios del sistema, delimitando en correspondencia con el permiso que tiene cada usuario la accesibilidad al sistema.

Bloques y menús: Representa la estructura de forma general de cómo está compuesto el sistema en cuanto a los bloques en los que está dividido y la cantidad de menús, para cada región contiene los elementos que componen cada bloque y la estructura de los menús del sistema.

Módulos (*modules*): Engloba los elementos que operan sobre los nodos otorgando funcionalidades a *Drupal*. Permiten incrementar sus capacidades o adaptarlas a las necesidades de cada portal web. Dentro de los módulos usados en la propuesta de solución, se encuentran gestionar planificación, gestionar grupo, los cuales fueron implementados para lograr una mejor gestión de la planificación.

Datos: Esta capa es la encargada de gestionar el acceso a la información almacenada en la base de datos referente al funcionamiento del sistema ("guardia_obrera") y a los contenidos que serán mostrados a través del tema activo, contiene además los nodos del sistema que están compuestos por la principal información del mismo.

Patrones de diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de *software*. Estos permiten diseñar sistemas seguros y que a su vez cumplan con los estándares de diseño establecidos por normas internacionales para el desarrollo de aplicaciones web (Buytaert, 2016).

Patrones *Gang of Four (*GOF)

El catálogo de patrones más famoso es el contenido en el libro "*Design Patterns: Elements of Reusable Object-Oriented Software*", también conocido como: El libro GOF (*Gang-Of-Four Book*). Según este documento (Gamma, y otros, 1997), estos patrones se clasifican por su propósito en creacionales, estructurales y de composición, mientras que respecto a su ámbito se clasifican en clases y objetos.

Instancia única: Este patrón está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un objeto único. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. Dentro del core de *Drupal* se utiliza este patrón de diseño en diversas tareas como la gestión de manejo de conexiones con la base de datos y pensando en los módulos y temas de *Drupal* como objetos para llevar a cabo la gestión de dichos elementos (Buytaert, 2016). Un ejemplo que evidenciará este patrón en el Sistema de gestión de la GO es el proceso de creación de tipos de contenidos, al cual se le asigna un identificador propio que evita la existencia en la base datos de elementos repetidos constituyendo así una instancia única.

Decorador: Este patrón de diseño permite añadir responsabilidades extra a objetos concretos de manera dinámica. Proporciona una alternativa flexible para extender funcionalidades. Brinda la flexibilidad de que nuevos módulos puedan modificar el comportamiento del núcleo en cuanto al tratamiento de los datos y en cada uno de los eventos del sistema (Buytaert, 2016). Se encontrará evidenciado en el sistema en el módulo que permite la gestión de la planificación de la GO.

Cadena de responsabilidades (*Chain of Responsibility):* El sistema de menús de *Drupal* sigue este patrón. En cada solicitud de la página, el menú del sistema determina si hay un módulo para gestionar la solicitud y si el usuario tiene acceso a los recursos solicitados. Para ello, el mensaje se pasa a la opción del menú correspondiente a la vía de la solicitud. Si el elemento de menú no puede manejar la petición, se pasa a otro. Esto continúa hasta que un módulo se encarga de la petición, un módulo niega el acceso para el usuario, o la cadena se ha agotado (Buytaert, 2016). La implementación del *hook_menu* es un ejemplo claro que genera un menú de configuración del módulo.

Diagramas de clases del diseño

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces *software* en una aplicación. A diferencia de las clases conceptuales del Modelo del Dominio, las clases de diseño de los DCD muestran las definiciones de las clases *software* en lugar de los conceptos del mundo real (Sommerville, 2011). A continuación, se muestra el DCD para el CU gestionar planificación.

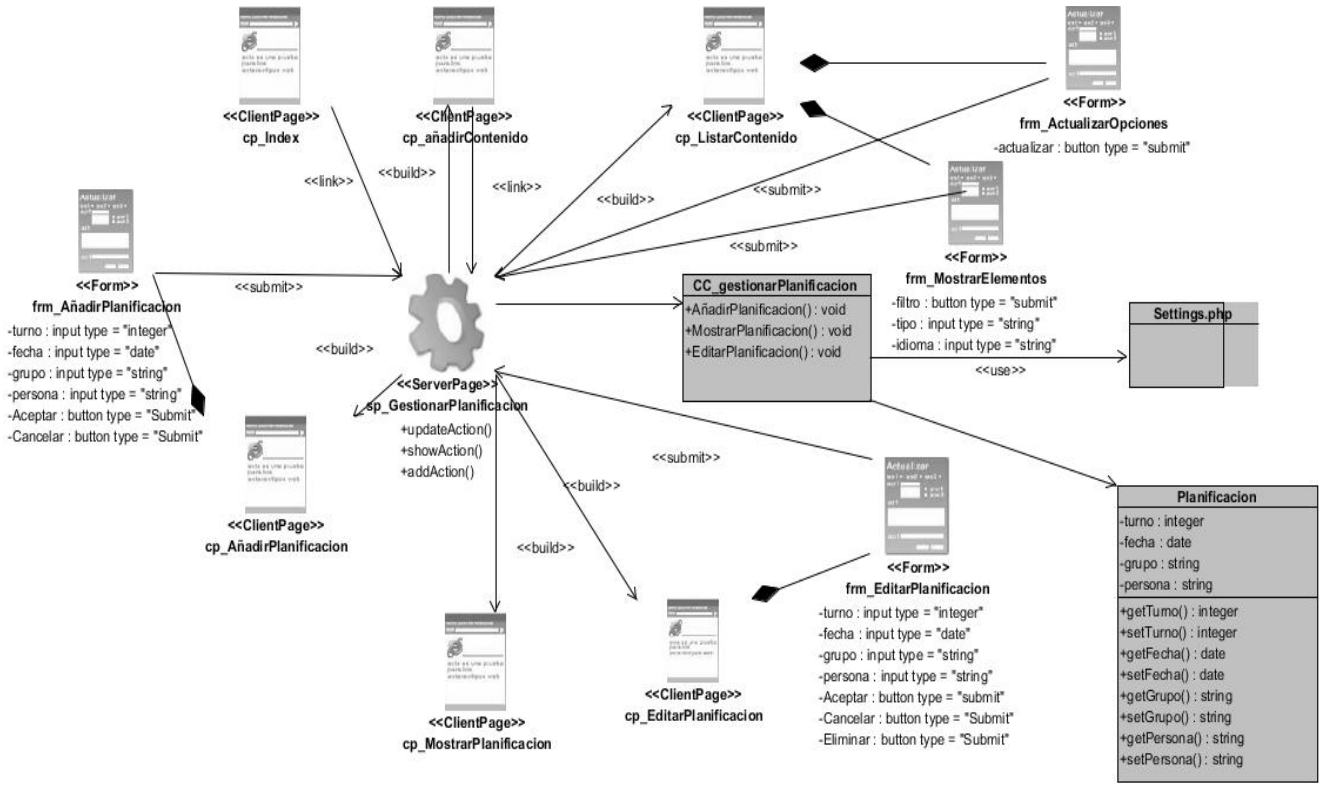


Figura 3. DCD con estereotipos web Gestionar Planificación

El DCD está compuesto por 1 *server_page* (página controladora/servidora) que se corresponde con la clase principal de gestión del contenido planificación, 6 *client_page* (página cliente/vista) que muestran las funcionalidades añadir y listar contenido, añadir, mostrar y editar planificación, y la página *index* correspondiente a la página principal del sistema, 3 formularios que contienen los campos botones y filtros asociados a los contenidos, la clase controladora que contiene todas las funcionalidades de la *server_page* (página controladora/servidora) , la clase que permite la conexión con la base de datos y la clase de la base de datos con los campos y funciones asociados al contenido planificación.

Diagrama de secuencia

Los diagramas de secuencia (DS) en el UML se usan principalmente para modelar las interacciones entre los actores y los objetos en un sistema, así como las interacciones entre los objetos en sí (Sommerville, 2011). Para la presente investigación se generó el diagrama de secuencia correspondiente al requisito funcional añadir planificación como se muestra a continuación.

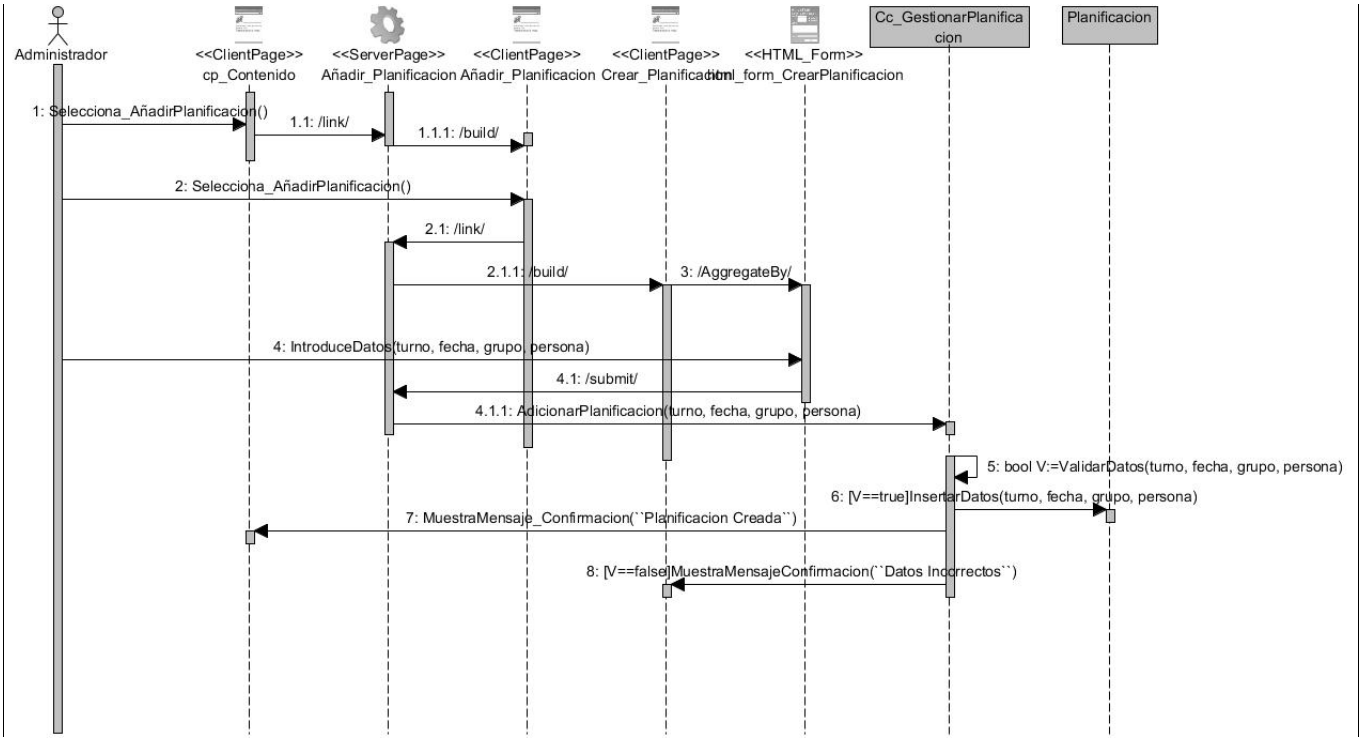


Figura 4. DS Añadir planificación

El flujo representado en el DS comienza cuando el actor (administrador) del sistema selecciona en el mismo la opción de añadir planificación. La página cliente intermediaria hace la solicitud a la *server_page* (página controladora/servidora). La página controladora construye la *client_page* (página cliente/vista) que permite añadir la planificación, esta hace link a la controladora que crea la página cliente con el formulario con los campos asociados a la planificación. El actor introduce los datos que se envían a la *server_page* (página controladora/servidora) y automáticamente se adiciona la planificación en la clase controladora donde se validan que los datos estén correctos. En caso de que los datos no hayan sido insertados de forma correcta se muestra un mensaje de error en la página cliente desde donde se creó la planificación, en el caso contrario se muestra un mensaje de confirmación en la clase desde donde el actor realizo la solicitud inicial.

2.4. Modelo de Despliegue

Un modelo de despliegue consiste en una representación estructural de la arquitectura del sistema desde el punto de vista de la distribución de los artefactos del software en los destinos de despliegue; definiendo

a los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo (Sarmiento, 2016).

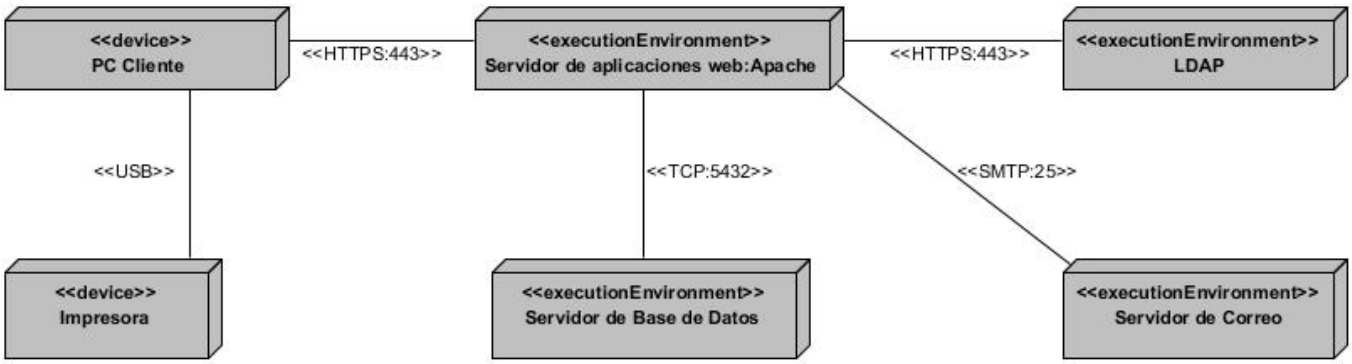


Figura 5 Modelo de Despliegue

Descripción de elementos e interfaces de comunicación

Dispositivo PC_Cliente: La estación de trabajo necesita un navegador web para conectarse al sistema hospedado en el servidor de aplicaciones utilizando el protocolo de comunicación HTTP/HTTPS.

Servidor de aplicaciones: Es la estación de trabajo que hospeda el código fuente de la aplicación y que le brinda al usuario las interfaces para realizar los procesos del sistema. Esta estación se comunica con el servidor de base de datos donde se almacenan los datos de la aplicación realizando la comunicación mediante el protocolo TCP/IP (*Transmission Control Protocol/Internet Protocol).*

Servidor de BD: Este servidor es el encargado del almacenamiento de los datos del sistema. Se comunica con el servidor de aplicaciones del sistema, posibilitando el acceso mediante el usuario con privilegios para las operaciones determinadas a realizarse en el mismo.

Servidor de correo: Este servidor es el encargado del envío de correos electrónicos comunicándose a través del protocolo SMTP (*Simple Mail Transfer Protocol*).

LDAP: (Protocolo Ligero/Simplificado de Acceso a Directorio) es un protocolo de tipo cliente-servidor para acceder a un servicio de directorio. Los servicios de directorio facilitan el acceso a información organizada a una gran variedad de aplicaciones. Estos servicios le permiten a los usuarios y aplicaciones autorizadas buscar información de personas, computadoras, aplicaciones y dispositivos red.

Impresora: Es un dispositivo periférico del ordenador que permite producir una gama permanente de textos o gráficos de documentos almacenados en un formato electrónico, imprimiéndolos en medios físicos

2.5. Conclusiones del capítulo 2

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

1. El análisis de las características del sistema y la modelación del dominio permitió identificar los principales requisitos funcionales y no funcionales del sistema de gestión de la guardia obrera, los cuales fueron agrupados y categorizados por casos de uso.
2. El diseño de los diagramas de clases y de secuencia facilitó la visión en cuanto a composición física y lógica del sistema.
3. La generación de todos los artefactos requeridos por el modelo de desarrollo documentaron la solución propuesta, lo cual facilitó su posterior mantenimiento (actualización o adición de funcionalidades).
4. La construcción del modelo de despliegue y la descripción del mismo brindó una mayor comprensión del sistema a desarrollar y de los elementos que lo componen.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBAS DEL SISTEMA PARA LA GESTIÓN DE LA GUARDIA OBRERA EN LA FACULTAD 1 DE LA UCI

3.1. Introducción

Una vez que se sabe qué funciones debe desempeñar el sistema y se ha decidido cómo organizar sus distintos componentes (diseño), es el momento de pasar a la etapa de implementación. En esta fase, según (Russo, 2011), se toma como punto de partida el modelo de la fase anterior y se procede a programar los diseños especificados, los cuales son implementados en términos de componentes, ficheros de código fuente y ejecutables.

Por otra parte, el desarrollo de un software es algo complejo y son innumerables las posibilidades de cometer errores. Por esta razón todo proceso de implementación debe ir acompañado de alguna actividad que garantice la calidad. Las pruebas de validación constituyen una base para garantizar la aceptación favorable de una aplicación informática por parte del usuario. Con la realización de las mismas se pretende encontrar y documentar los errores que tiene un sistema, validar los requisitos y comprobar que estos fueron implementados correctamente. Este capítulo tiene como objetivo, documentar los resultados de las fases de implementación del sistema y de la estrategia de pruebas desarrollada.

3.2. Diagrama de componentes

El diagrama de componentes muestra los componentes de un sistema de *software* conectados por las relaciones de dependencias lógicas entre cada uno de ellos. Provee una vista arquitectónica de alto nivel del sistema, ayudando a los desarrolladores a visualizar el camino de la implementación. Cada componente representa una unidad del código (fuente, binario o ejecutable), que permite mostrar las dependencias en tiempo de compilación y ejecución. La realización del diagrama posibilita tomar decisiones respecto a las tareas de implementación y los requisitos (RIVERA ALVA, 2017).

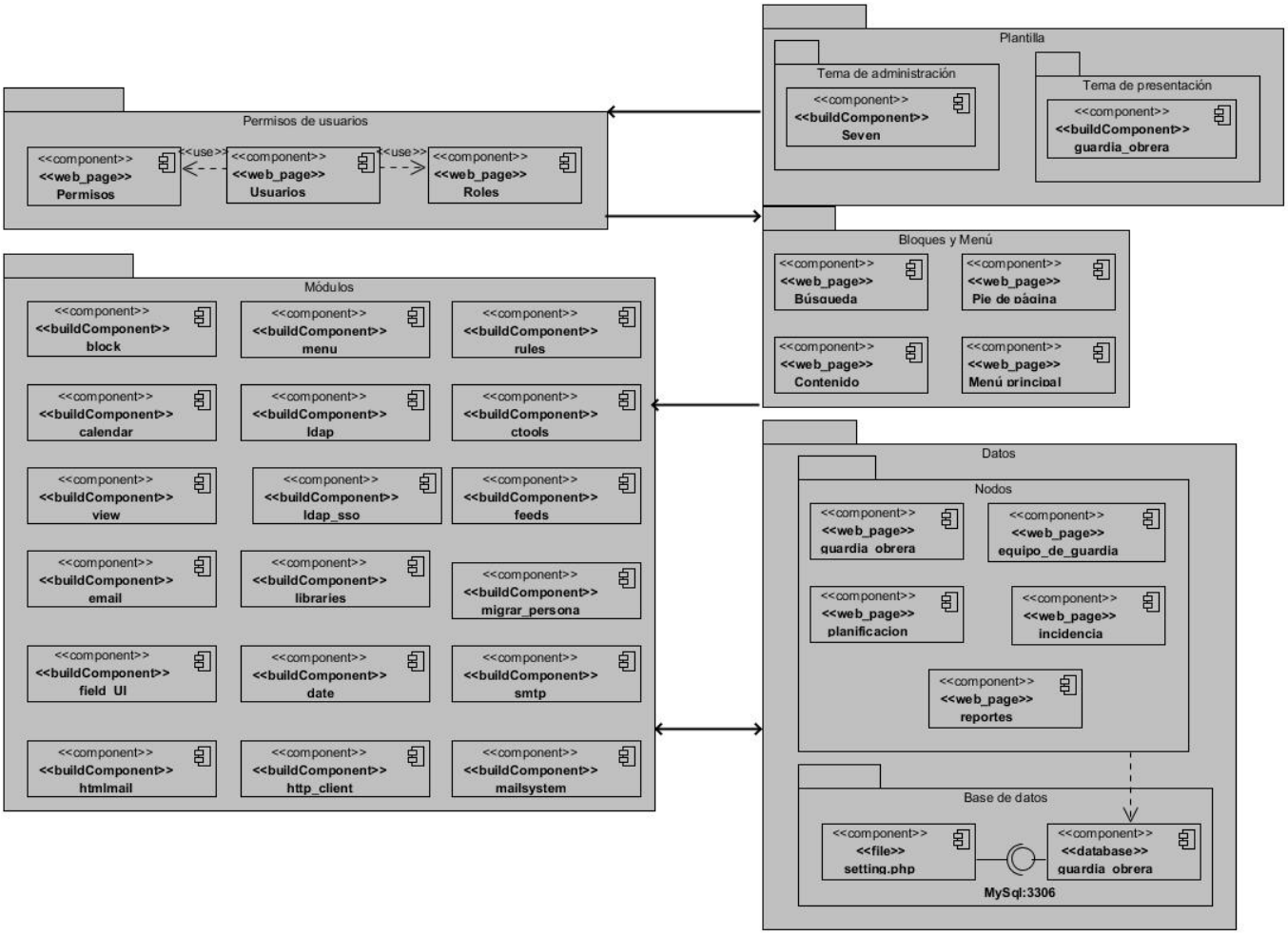


Figura 6. Diagrama de componentes

El diagrama de componentes asemeja cada uno de los paquetes con el contenido que tiene cada capa de la arquitectura de software utilizada por *Drupal* (n-capas), para la capa de plantilla contiene el tema utilizado en el sistema, ”*Sistema para la gestión de la GO”,* en la capa de datos se encuentran los nodos del sistema y la conexión con la base de datos, en la capa de bloques y menús se muestran la estructura del sistema, en la capa de los módulos los principales instalados y utilizados por el sistema, y en la capa de los permisos de usuario se encuentra como está establecida la jerarquía que representa la relación existente entre los permisos, roles y usuarios que existen en el sistema.

3.3. Estándares de codificación

Los estándares de codificación constituyen un principio esencial en el desarrollo de *software*. Garantizan que el código obtenido sea fácil de leer, entendido y modificado independientemente de quien haya sido el desarrollador del producto. Son una guía para el equipo de desarrollo, permiten asegurar que el código presente calidad y no contenga errores. *Drupal* proporciona a sus desarrolladores un conjunto de normas para fomentar el código de una forma uniforme para todos (Buytaert, 2016). A continuación, se detallan los estándares de codificación utilizados en la implementación de la solución propuesta.

Etiquetas de apertura y cierre

Cuando se escribe en PHP, siempre se deben utilizar las etiquetas **<?php** y **?>**, y en ningún caso la versión corta **<?** y **?>**. En general se omite la etiqueta de cierre de PHP (**?>**) al final de los archivos .module y .inc. Esta convención evita que se puedan quedar olvidados espacios no deseados al final del archivo (después de la etiqueta de cierre ?**>**), que serían identificados como salida HTML y podrían provocar un error muy típico, "*Cannot modify header information-headers alrealy sent by…*". Por tanto, la etiqueta de cierre final del archivo (**?>**) es opcional en *Drupal* (Gil, 2012). En la (Figura 7) se puede apreciar un fragmento de código el cual evidencia el uso de las etiquetas de apertura de PHP.

```php
<?php

function list_funcionalidades() {


    $foto = '';
    try {
        if (module_exists( module: 'ws')) {
            module_load_include( type: 'inc', module: 'ws', name: 'ws');
            $obj = new wsdl("http://assets.uci.cu/servicios/v4/AssetsWS.php?wsdl");
            $data = $obj->__call_pasarela1('ObtenerAreaDadoIdExpediente', 'T18013');
            $foto = $data->NombreArea; print $foto; die();
        }
```

Figura 7. Ejemplo de uso de apertura de etiqueta

Identación

La identación consiste en insertar espacios en blanco o tabuladores en determinadas líneas de código para facilitar su comprensión. En programación se emplea identación para anidar elementos. En *Drupal* se debe identar con 2 espacios, nunca con tabuladores. Además, no se debe dejar espacios en blanco al final de

cada línea (Gil, 2012). En el siguiente ejemplo (Figura 8) se presenta un fragmento de código que demuestra la identación del código.

```
function migrate_import_usuario_form($form, &$form_state, $arg = NULL) {
    $form['from'] = array(
        '#type' => 'item',
        '#title' => t( string: 'Importar todas las personas '),
        '#markup' => 'Los datos actuales de las personas serán actualizados. Esta operaci&oacute;n no se puede deshacer.',
    );
```

Figura 8. Ejemplo de uso de identación

Operadores

Los operadores binarios, que se utilizan entre dos valores, deben separarse de estos valores, a ambos lados del operador, por un espacio. Por ejemplo, $max_age = 452, en el lugar de $max_age=452. Esto se aplica a operadores como +,-,*,/,=,==,!=,>,<,. (Concatenación de cadenas), .=, +=, -=. Los operadores unarios como ++, -- no deben tener separación (Gil, 2012). En la (Figura 9) se puede apreciar lo antes explicado en un ejemplo en el código de la solución.

```
    // Eliminando todas las personas
    $eliminar = $form['eliminar']['#value'];
    if($eliminar==1){
        $node_type = 'trabajador';
        $result = db_select( table: 'node', alias: 'n')
            ->fields( table_alias: 'n', array('nid'))
            ->condition('type', $node_type, '=')
            ->execute();
        $deleted_count = 0;
        foreach ($result as $record) {
            node_delete($record->nid);
```

Figura 9. Ejemplo de uso de operadores

Uso de comillas

Se utilizan tanto comillas simples como la (‘cadena’) como las comillas dobles (“cadena”) para delimitar las cadenas de caracteres. Las comillas dobles son necesarias si se desean incluir variables dentro de las

cadenas de texto (Gil, 2012). En la (Figura 9) se evidencia el uso de las comillas en la codificación de la propuesta de solución.

```
$foto = '';
try {
    if (module_exists( module: 'ws')) {
        module_load_include( type: 'inc', module: 'ws', name: 'ws');
        $obj = new wsdl("http://assets.uci.cu/servicios/v4/AssetsWS.php?wsdl");
        $data = $obj->__call_pasarela1('ObtenerAreaDadoIdExpediente', 'T18013');
        $foto = $data->NombreArea; print $foto; die();
    }
```

Figura 10. Ejemplo del uso de comillas

Uso de punto y coma (;) en código PHP

Aunque PHP permite escribir líneas de código individuales sin el terminador de línea (;), como por ejemplo <?php print $title ?>. En *Drupal* es siempre obligatorio: <?php print $title; ?>. Es importante señalar que el cierre de la etiqueta php es opcional (Gil, 2012).

Estructuras de control

Según (Gil, 2012) con respecto a las estructuras de control, hay que tener en cuenta las siguientes normas:

- Debe haber un espacio entre el comando que define la estructura (*if, while, for*) y el paréntesis de apertura. Esto es así para no confundir las estructuras de control con la nomenclatura de las funciones.

```
$foto = '';
try {
    if (module_exists( module: 'ws')) {
        module_load_include( type: 'inc', module: 'ws', name: 'ws');
        $obj = new wsdl("http://assets.uci.cu/servicios/v4/AssetsWS.php?wsdl");
        $data = $obj->__call_pasarela1('ObtenerAreaDadoIdExpediente', 'T18013');
        $foto = $data->NombreArea; print $foto; die();
    }
```

Figura 11. Ejemplo del uso de la estructura IF

- La llave de apertura ({) se situará en la misma línea que la definición de la estructura, separada por un espacio.

```
if ($fp = fopen($user_file->uri, mode: 'r')) {
    while (!feof($fp)) {
        $row = fgetcsv($fp, length: NULL, delimiter: ",");
        $data['users'][] = $row;
```

Figura 12. Ejemplo del uso de la llave de apertura

- Se recomienda usar siempre las llaves {} aun en los casos en que no sea obligatorio su uso (una sola "línea" de código dentro de la estructura de control).

```
if($sexo=="Masculino"){
    $node->field_sexo['und'][0]['value'] = Masculino;
}else{
    $node->field_sexo['und'][0]['value'] = Femenino;
}
```

Figura 13. Ejemplo de uso de llave de apertura y cierre

- Las estructuras *else* y *elseif* se escribirán en la línea siguiente al cierre de la sentencia anterior.

```
}elseif (variable_get( name: 'maintenance_mode', default: 0)) {
  watchdog( type: 'cron', message: 'Cron could not run because the site is in maintenance mode.', array(), severity: WATCHDOG_NOTICE);
  drupal_access_denied();
}
else {
  drupal_cron_run();
}
```

Figura 14. Ejemplo de uso de las estructuras else y elseif

Funciones

Los nombres de las funciones deben estar escritos en minúsculas y las palabras separadas por guión bajo. Además, se debe incluir siempre como prefijo el nombre del módulo o tema, para evitar así duplicidad de funciones. En su declaración, después del nombre de la función, el paréntesis de inicio de los argumentos debe ir sin espacio. Cada argumento debe ir separado por un espacio, después de la coma del argumento anterior (Gil, 2012).

```
function migrate_import_usuario_form_validate($form, &$form_state) {

    $file = $form_state['values']['users'];

    $validators = array('file_validate_extensions' => array('csv'));
    $user_file = file_save_upload( form_field_name: 'user_upload', $validators,
```

Figura 15. Ejemplo del nombre de una función

En la llamada a la función se aplican las mismas reglas anteriores con respecto a los parámetros, como se muestra a continuación:

```
function hook_node_limit_save($lid, $applies, $element) {
  if ($applies) {
    // $element contains the username of the user
    // user_load based on the name to get the uid
    $user = user_load_by_name($element);
```

Figura 16. Ejemplo de llamada a una función

Arreglos (*Arrays*)

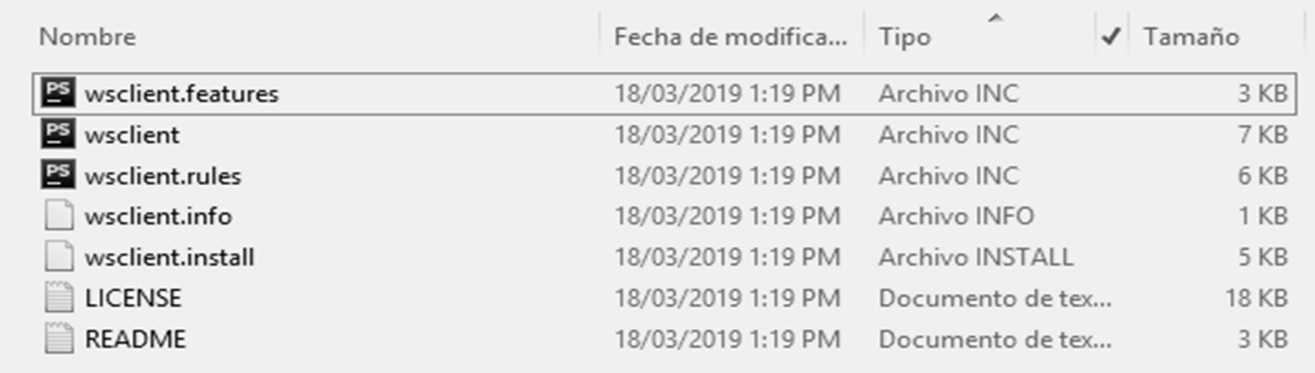
Los valores dentro de un *array* (o matriz) se deben separar por un espacio (después de la coma que los separa). El operador => debe separarse por un espacio a ambos lados. Cuando la línea de declaración del *array* supera los 80 caracteres, cada elemento se debe escribir en una única línea. En este último caso, la coma de separación del último elemento también se escribirá, aunque no existan más elementos. De esta forma se evitan errores al añadir nuevos elementos al vector (Gil, 2012).

```
    $form['user_upload'] = array(
        '#type' => 'file',
        /*'#required' => TRUE,*/
        '#title' => 'Archivo',
        '#description' => t( string: 'Cargar el archivo desde su equipo.'),
    );
```

Figura 17. Ejemplo de arreglo

Nombres de archivos

Los nombres de archivos deben escribirse siempre en minúscula. La única excepción son los archivos de documentación, que tendrán extensión .txt y el nombre en mayúscula. Por ejemplo README.txt.



Nombre	Fecha de modifica...	Tipo	Tamaño
wsclient.features	18/03/2019 1:19 PM	Archivo INC	3 KB
wsclient	18/03/2019 1:19 PM	Archivo INC	7 KB
wsclient.rules	18/03/2019 1:19 PM	Archivo INC	6 KB
wsclient.info	18/03/2019 1:19 PM	Archivo INFO	1 KB
wsclient.install	18/03/2019 1:19 PM	Archivo INSTALL	5 KB
LICENSE	18/03/2019 1:19 PM	Documento de tex...	18 KB
README	18/03/2019 1:19 PM	Documento de tex...	3 KB

Figura 18. Ejemplo de nombre de los archivos

Comentar el código

Para la realización de comentarios suelen emplear ***/*/*** para comentarios en varias líneas y ***//*** para comentarios de una única línea. Se deben escribir frases completas, comenzándolas con mayúscula y terminándolas con un punto. En caso de que en el comentario se haga referencia a una constante, esta deberá escribirse en mayúscula (por ejemplo: TRUE o FALSE) (Gil, 2012).

```
function migrate_import_usuario_form_submit($form, &$form_state) {

    // Eliminando todas las personas
    $eliminar = $form['eliminar']['#value'];
    if($eliminar==1){
        $node_type = 'trabajador';
```

Figura 19. Ejemplo de comentario de código

3.4. Aplicación de la estrategia de prueba

Para garantizar el completo funcionamiento de toda aplicación informática es necesario realizar pruebas en todas sus categorías y de esta forma evitar cualquier problema o insatisfacción por parte de los clientes. Estas pruebas se trazan después de estrategias que permitan evaluar todos los aspectos de determinado producto teniendo en cuenta su característica y así de una forma u otra estar completamente confiados y poder garantizar el éxito del mismo. (Rosabal, 2005). La estrategia de pruebas realizadas está dirigida a componentes del software, analizando la usabilidad, disponibilidad, respuesta, las pruebas de seguridad y la aceptación por parte del cliente.

Pruebas de rendimiento (carga y estrés)

Las pruebas de rendimiento se diseñan para asegurar que el sistema pueda procesar su carga esperada. Éstas se ocupan tanto de demostrar que el sistema satisface sus requerimientos, como de descubrir problemas y defectos en el sistema (Sommerville, 2011).

Las pruebas de carga consisten en simular una carga de trabajo similar y superior a la que tendrá cuando el sistema esté funcionando, con el fin de detectar si el *software* instalado (programas y aplicaciones) cumple con los requerimientos de muchos usuarios simultáneos y también si el *hardware* (servidor y el equipamiento computacional de redes y enlace que lo conecta a Internet) es capaz de soportar la cantidad de visitas esperadas (PRESSMAN, 2002).

Las pruebas de estrés evalúan la robustez y la confiabilidad del *software* sometiéndolo a condiciones de uso extremas. Entre estas condiciones se incluyen el envío excesivo de peticiones y la ejecución en condiciones de hardware limitadas. El objetivo es saturar el programa hasta un punto de quiebre donde aparezcan defectos potencialmente peligrosos (PRESSMAN, 2002).

Resultados de las pruebas de rendimiento

Para las pruebas de rendimiento se utiliza el *software* Apache Jmeter v2.10. Para ello se definen las propiedades de las PC implicadas.

***Hardware* de prueba (PC servidor)**

- Sistema Operativo: Linux Mint v.18
- Microprocesador: Intel(R) Core(TM) i3-7100U CPU @2.40GHz 2.40GHz
- Memoria RAM: 8.00 GB
- Disco Duro: 1024 GB

Luego de definido el hardware se configuran los parámetros del Apache JMeter logrando un ambiente de simulación con un total de 100 usuarios conectados concurrentemente, se realizan peticiones a diferentes páginas del Sistema para la Gestión de guardia obrera de la Facultad 1. En la (Figura 20) se pueden observar los resultados obtenidos por el sistema.

A continuación, se describen las variables que miden el resultado de las pruebas de carga y estrés realizadas al módulo:

- **Usuarios:** total de usuarios.

- **# Muestras:** El número de peticiones.
- **Media:** El tiempo medio transcurrido en milisegundos para un conjunto de resultados.
- **Mín:** El mínimo tiempo transcurrido en milisegundos para las muestras de la URL dada.
- **Máx:** El máximo tiempo transcurrido en un milisegundo para las muestras de la URL dada.
- **% Error:** Porcentaje de las peticiones con errores.
- **Rendimiento:** Rendimiento medido en base a peticiones por segundo/minuto/hora.
- **Kb/s Recibidos:** Rendimiento medido en Kbytes por segundos.

Tabla 4. Resultados de las pruebas de rendimiento

Usuarios	# Muestras	Media	Mín	Máx	% Error	Rendimiento (peticiones/segundos)	Kb/s Recibidos
100	200	29963	2904	31869	0 %	1.6	1.7

El sistema desarrollado, para un total de 100 usuarios conectados de forma concurrente respondió 200 peticiones al servidor en un promedio de 29.963 segundos, lo que equivale a 1.6 peticiones por segundo. Atendiendo a la cantidad de peticiones por cada segundo que se enviaron y las prestaciones del *hardware* donde se realizaron las pruebas se considera que constituye un resultado satisfactorio.

Pruebas funcionales

Se denominan pruebas funcionales a las pruebas de *software* que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados (Rivasi, 2009).

Tabla 5. Caso de prueba: Añadir planificación

Caso de prueba Añadir planificación
Nombre de requisito: Añadir planificación
Nombre de la persona que realiza la prueba: Daniela Herrera González
Descripción de la prueba: Prueba a la funcionalidad añadir planificación.
Entrada/Pasos de ejecución: Se seleccionan los siguientes datos para crear la planificación.
Mes **Año** **Trabajador**

El administrador presiona el botón aplicar y si los datos están correctos se crea la planificación que podrá ser visualizado desde la vista Planificación para los usuarios autenticados en el sistema, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.
Evaluación de la prueba: Satisfactoria.

Tabla 6. Caso de prueba: Editar equipo de guardia

Caso de prueba Editar equipo de guardia
Nombre de requisito: Editar equipo de guardia
Nombre de la persona que realiza la prueba: Daniela Herrera González
Descripción de la prueba: Prueba a la funcionalidad editar equipo de guardia.
Entrada/Pasos de ejecución: Se modifican los siguientes datos para editar un contenido de tipo equipo de guardia.
Datos actuales **Nombre:** resd **Período:** Lectivo **Horario:** Diurno **Integrante 1:** Yordanka Fuentes Castillo **Integrante 2:** Carlos Yordan González Herrera **Datos después de editados** **Nombre:** Equipo 1 **Período:** Lectivo **Horario:** Diurno **Integrante 1:** Yordanka Fuentes Castillo **Integrante 2:** Carlos Yordan González Herrera El administrador presiona el botón guardar y si los datos están correctos actualiza el equipo de guardia que podrá ser visualizado desde la vista Equipo de guardia, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.
Evaluación de la prueba: Satisfactoria.

Resultado de las pruebas funcionales

Para validar que el sistema cumpla con las funciones específicas para las cuales ha sido creado se realizaron las pruebas funcionales arrojando como resultados en una primera iteración 20 no conformidades

de las cuales 4 fueron resueltas. En la segunda iteración quedaron pendientes 16 de las cuales se resolvieron 7 y en la última iteración, de las 9 restantes, se resolvieron todas. En la figura 21 se muestran los resultados obtenidos en la iteración de pruebas realizadas al Sistema para la gestión de la guardia obrera en la Facultad 1.

Figura 20. Resultado de la primera iteración de pruebas funcionales

Las no conformidades de funcionalidad, estuvieron relacionadas con diversas opciones en el sistema como por ejemplo crear una planificación en el sistema e importar los datos desde el documento *excel* conjuntamente con el servicio web. En el primer caso, no se permitía crear la planificación sin asociarlo a un equipo de guardia, no debiendo ocurrir así, debido a que puede existir la situación de que se necesite planificar a un solo trabajador y no a un equipo de guardia. Esta no conformidad fue resuelta con nuevas validaciones en el sistema que posibilitaron planificar tanto para equipo como para un trabajador de forma individual. En el segundo caso, al importar los datos de los trabajadores desde ambos lugares estrictamente se detectó que cuando el servicio web no está habilitado podía existir problemas con dicha información y fue necesario modificar y especificar que fuera suficiente con la información contenida en el documento *excel* previamente elaborado.

Las no conformidades de validación fueron encontradas en el formulario para crear un equipo de guardia donde inicialmente el sistema permitía crearlo con el mismo trabajador de forma repetida, y en el caso del correo donde se permitía introducir caracteres extraños y no la estructura que el mismo debe seguir. La solución para los caracteres extraños fue validar la estructura del correo y para el caso de los equipos incluir la restricción de que ambos trabajadores fuesen diferentes. El sistema muestra un mensaje de error si se intenta realizar esta acción de manera incorrecta.

Pruebas de seguridad

Las pruebas de seguridad se realizan para comprobar que los mecanismos de protección integrados en el sistema realmente lo protejan de irrupciones inapropiadas (PRESSMAN, 2002). Además, se encargan de certificar que los datos y las funciones del sistema solo son accesibles por los actores debidamente autorizados (Toll, 2007).

Resultados de las pruebas de seguridad

Con el objetivo de evaluar la seguridad de la solución propuesta se emplea la herramienta Acunetix WVS la cual arrojó los siguientes resultados.

Tabla 7. Resultados de la prueba de seguridad

Categorías de vulnerabilidades	Cantidad de errores
Vínculos rotos	1
Ataque de adivinación de contraseñas en la página de inicio de sesión	1
Las credenciales de usuario se envían en texto claro	5
Secuencias de comandos entre sitios	1
Total	8

La prueba realizada mediante la herramienta Acunetix WVS, arrojó que en la primera iteración se detectan 8 no conformidades, de ellas 2 alertas de riesgo alto, 5 alertas de riesgo medio y 1 de riesgo bajo. Para proporcionar una mayor seguridad se configuró el servidor de forma tal que solo pudieran acceder al sistema los usuarios con permiso, así como la instalación de módulos para mayor seguridad, tales como *secure_login*, *seckit*, *session_limit*.

Pruebas de aceptación

En ingeniería de software y pruebas de software, las pruebas de aceptación pertenecen a las últimas etapas previas a la liberación en firme de versiones nuevas a fin de determinar si cumplen con las necesidades y/o requerimientos de las empresas y sus usuarios (PRESSMAN, 2002).

Por su parte, la Junta Internacional de Cualificaciones de Pruebas de Software (*ISTQB* por sus siglas en inglés) define la "Aceptación" aplicado a la rama de la informática como: Pruebas formales que se realizan atendiendo las necesidades del cliente, requerimientos y procesos de negocio, realizadas para determinar si un sistema satisface los requerimientos de aceptación que permitan que el cliente pueda determinar si acepta o no el sistema (Herreros, 2015).

Para realizar la prueba de aceptación, se entregó la aplicación al cliente, el cual emitió su criterio a través de una carta de aceptación, a partir de sus consideraciones respecto a las ventajas que ofrecen el sistema y las necesidades que resuelve.

Pruebas de usabilidad

Según diversos estándares de la Ingeniería de Software, se puede definir la usabilidad como el grado en el que un producto puede ser utilizado por usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción, en un determinado contexto de uso. Como se puede apreciar, la usabilidad de un sistema está ligada a usuarios, necesidades y condiciones específicas (Carcasés, 2016). De manera general, el término usabilidad es empleado para referirse a la capacidad que posee un producto de ser utilizado por los usuarios de forma fácil, eficiente y con satisfacción, en un determinado contexto de uso.

Se puede decir que el proceso de prueba de usabilidad se enfoca en satisfacer las necesidades de los usuarios finales basados en métricas definidas durante la planeación. Para la realización de las pruebas de usabilidad, se hace uso de la "Lista de Chequeo de Usabilidad para sitios web", desarrollada por los especialistas del grupo de Seguridad del Departamento de Evaluación de Productos de Software (DEPSW), perteneciente al Centro Nacional de Calidad de Software (CALISOFT). A continuación, se muestran los resultados de dichas pruebas.

Tabla 8. Resultados de las pruebas de usabilidad

Categoría de los indicadores	Indicadores	Proceden	Correctos	Incorrectos
Visibilidad del sistema	17	14	12	2
Lenguaje común sistema/usuario	12	10	9	1

Libertad y control por parte del usuario	29	24	18	6
Consistencia y estándares	33	27	22	5
Estética y diseño minimalista	17	15	15	0
Prevención de errores	8	7	6	1
Ayuda y documentación	11	11	7	4
Flexibilidad y eficiencia	6	4	3	1
Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de los errores	11	11	8	3
Total	144	123	100	23

Como se observa en la tabla, de los 144 parámetros originales de la lista de chequeo, solo proceden 123. En la primera iteración se evaluaron como correctos 100 parámetros, identificando 23 no conformidades, para un 81.30 % de usabilidad. Algunos de los problemas detectados en primeras iteraciones de las pruebas de usabilidad fue que el sistema luego de una acción relevante relacionada con los contenidos del mismo no había vuelta atrás a la página donde se encontraba inicialmente el administrador. Siguiendo el principio de la realización de un sistema intuitivo para todo usuario habiendo o no trabajado antes con la tecnología se validó luego de la modificación del contenido desde las páginas internas del mismo al realizar una acción relevante el sistema volvía a la vista principal de dicho contenido, otra de las no conformidades detectadas y resueltas estuvo relacionada con las resoluciones de pantalla, en un inicio el sistema solo era visible de forma correcta para determinadas resoluciones y luego de reajustado lo cumple para todas.

3.5. Interfaces principales del Sistema de gestión de guardia obrera

Una vez desarrollado el Sistema de gestión de guardia obrera, es posible visualizar las pantallas principales del mismo, donde se observa el resultado obtenido durante la implementación de los requisitos funcionales descritos en el capítulo 2.

La pantalla actual es la vista de incidencias. Dicha vista solo es accesible para el administrador del sistema que es el encargado de gestionar las mismas mediante los botones que contiene la vista que la permiten trabajar con las incidencias.

Figura 21. Captura de pantalla del sistema Crear Incidencias

La pantalla actual es la vista principal del sistema relacionada con el contenido planificación, para el día en cuestión se puede observar los trabajadores a los cuales les corresponde realizar la guardia ese día agrupados en equipos de guardia y por turnos. La vista contiene igualmente los botones con las funcionalidades que permiten buscar algún trabajador así como aplicar algún filtro.

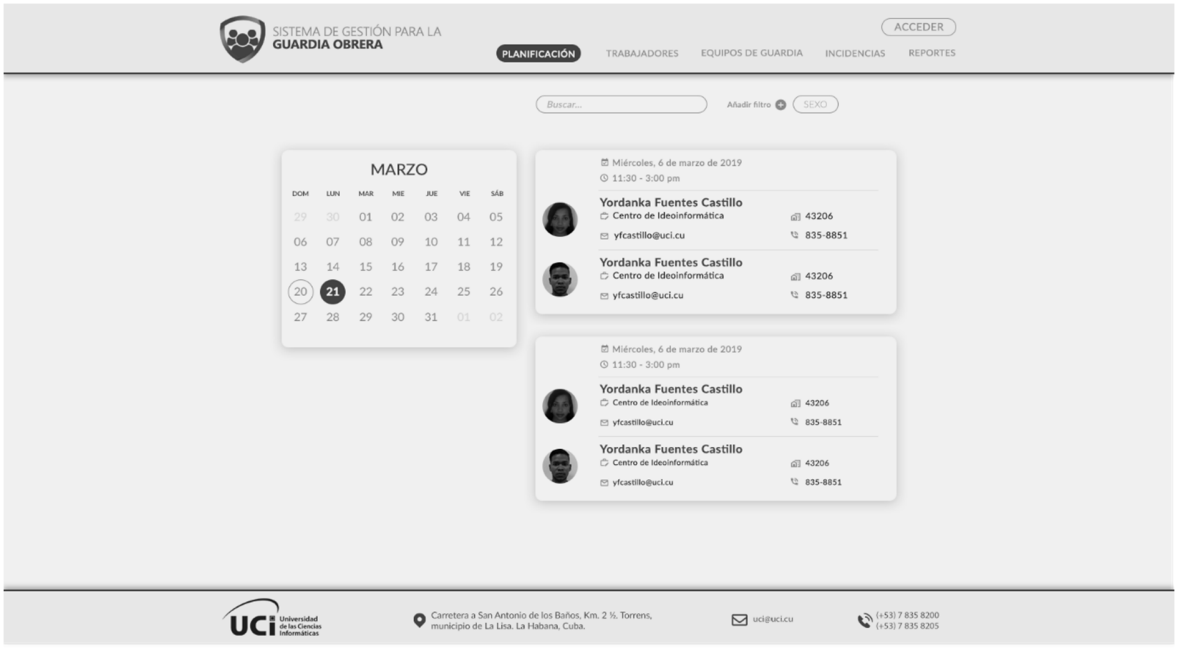


Figura 22. Captura de pantalla del sistema Mostrar planificación

3.6. Conclusiones parciales

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

1. La confección del diagrama de componentes permitió observar la integración de los componentes de *software*.
2. Aplicar los estándares de codificación permitió obtener en el sistema un código legible, estándar y fácil de comprender lo que asegura la calidad y facilita un futuro mantenimiento.
3. El proceso de validación de la solución propuesta a través de las pruebas de carga y estrés, funcionalidad, seguridad, usabilidad y aceptación permitió identificar y corregir las no conformidades detectadas para obtener un producto de mayor calidad.

CONCLUSIONES GENERALES

1. El estudio de los referentes teóricos y el análisis de las diferentes herramientas y tendencias para la gestión de contenidos permitió determinar la no existencia de un sistema informático que responda a las necesidades requeridas por el cliente.
2. El diseño de la propuesta de solución permitió generar los artefactos más significativos de acuerdo con la metodología de desarrollo de *software* AUP-UCI tomándose como referencia los requisitos detectados.
3. La implementación del sistema a través de las herramientas y lenguajes seleccionados permitió obtener una aplicación web capaz de manejar datos referentes a los trabajadores y la planificación del proceso de guardia que realizan los mismos.
4. Las técnicas de validación aplicadas a la propuesta de solución permitieron la detección y corrección de las no conformidades detectadas y evidenciaron que el sistema constituye una solución funcional.
5. La validación del problema de investigación mediante la carta de aceptación emitida por el cliente demostró que el Sistema para la gestión de la Guardia Obrera de la Facultad 1 en la UCI contribuye a informatizar el proceso de planificación de la guardia para cada uno de los trabajadores que radican en la Facultad.

RECOMENDACIONES

Al concluir el presente trabajo investigativo, se recomienda para investigaciones futuras:

- Implementar un algoritmo inteligente que permita generar la planificación automática de la guardia obrera estudiantil.
- Diseñar patrones que puedan ser asignados a las personas en el momento de planificar la guardia. Un ejemplo de patrón sería: Realizar la guardia todos los días primeros de cada mes.
- Antes de realizar la planificación importar los datos de los trabajadores desde alguna fuente más actualizada o disponible para el caso del servicio web desde donde la aplicación actual consume los datos para completar el contenido de los trabajadores.

REFERENCIAS BIBLIOGRÁFICAS

ApacheJMeeter. 2016. [Online] 2016. [Citado em: 5 de enero de 2019.] http://jmeter.apache.org.

Betancourt, Alberto Nuñez. 2018. Trabajadores. Órgano de la Central de Trabajadores de Cuba. [Online] 2018. [Citado em: 20 de noviembre de 2019.] digital@trabajadores.cu.

Bravo, Juan. 2016. AIPEMBlog. [Online] 2016. [Citado em: 25 de noviembre de 2018.]

Buytaert, Dries. 2016. Drupal.org. [Online] 2016. [Citado em: 13 de enero de 2019.]

CAMPO, CÉSAR MAURICIO. 2013. programming site. [Online] 2013. [Citado em: 2 de marzo de 2019.] http://repositorio.uchile.cl.

Carcasés, Arianne Ferrer. 2016. *Guía para los procesos de Usabilidad que se desarrollan en el Laboratorio de Pruebas de Software (LPS) de la Dirección de Calidad-UCI.* 2016.

Castillo, Alvaro del. 2006. *Web dinamicas con PHP.* 2006.

Cockburn, Alistair. 2008. *Agile Software Development.* 2008.

2008. Diccionario de la Real Academia Española. Información [en línea]. [Online] 2008. [Citado em: 16 de diciembre de 2018.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=control%F6%A2g.

2008. Diccionario de la Real Academia Española. Información [en línea]. [Online] 2008. [Citado em: 13 de diciembre de 2018.] http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=informaci%C3%B3n.

Ferran, Salvador. 2016. www.DSA.com. [Online] 2016. [Citado em: 18 de abril de 2019.] http://dsav.net/wp-content/uploads/2013/02/Acunetix-WVS-castellano.pdf.

Flores, Eduardo Machado. 2011. Foro de seguridad. [Online] 2011. [Citado em: 20 de abril de 2019.]

Gamma, Erich e otros. 1997. *Design Patterns: Elements of Reusable Object-Oriented Software.* 1997.

Gervacio, Luis Olarte. 2018. *Ingeniería y tecnología.Ciencias computacionales y de la información.* 2018.

Gil, Frank. 2012. *Experto en drupal.Curso de creación y gestión de portales web en drupal 7.Nivel Experto.* 2012.

Gonzalez Guadalupe, Exxon Anyer e León Pérez, Miguel Ángel. 2015. *Sistema de Gestión de la Guardia Obrera Estudiantil en la Facultad 1.* 2015.

Gonzalez, Guillermo Valles. 2014. *Una introduccion a Apache.* 2014.

Gutiérrez, Emmanuel. 2009. *JavaScript: Conceptos básicos y avanzados. Barcelona : Informática Técnica.* 2009. ISBN: 978-2-7460-4341-1. .

Heredia Hanza, Xiomara Beatriz e Guerrero Vera , Judith Santa. 2010. *Estudio de PHP y MySQL para el desarrollo del portal web de Esmeraldas.* 2010.

Herreros, Rafael Garcia. 2015. PMOinformatica.com. [Online] 2015. [Citado em: 18 de abril de 2019.]

Jones Pérez, Liadelis e Bach Eng, Raúl . 2016. Componente para la Gestión de la Guardia Obrera Estudiantil en la Universidad de las Ciencias Informáticas. 2016.

Jose, Maria e Auquilla, Guapi. 2018. *Diseño metodologico para el desarrollo de interfaces graficas en paginas web utilizando los lenguajes HTML 5 y CSS 3.* s.l. : Riobamba, 2018.

Kolbe, Massimiliano. 2015. Tecno Hospital. [Online] 2015. [Citado em: 2018 de octubre de 27.]

Kriesi, Frank. 2015. Visual Time.com. [Online] 2015. [Citado em: 10 de febrero de 2019.]

LARMAN, C. UML y Patrones. 2017. *Lenguaje UML y patrones.* 2017.

Mata Sanchez, Dailin e Navarro Quintero, Aroldo. 2013. Sistema para la gestion de la guardia obrero estudiantil de la Facultad 3, UCI. [Online] 2013. [Citado em: 2018 de octubre de 3.]

Mauricio, Maldonado &. 2017. *Estudio de la integración de los frameworks bootstrap.* 2017.

Morell Álex, Garcia Jordi. 2014. www.departamentodeinternet.com. [Online] 2014. [Citado em: 30 de enero de 2019.] https://www.departamentodeinternet.com/que-es-un-cms-y-que-ventajas-tiene/.

Mulet, Jose Miguel. 2018. Revista Digital INESEM. [Online] 2018. [Citado em: 2019 de enero de 19.]

Olivares, Antonio Molina. 2016. *Implementacion de una aplicacion web usando el sistema gestor de contenidos Drupal.* 2016.

Patrice Lamothe, Francois Rocaboy, Alain Cohen, Samuel Tissier, Nicolas Cynober. 2009. www.pearltrees.com. [Online] 2009. [Citado em: 21 de enero de 2019.] https://www.opensourcecms.com/content-management-systems-vs-frameworks/.

PINZÓN, S. y GUEVARA BOLAÑOS, J.C., . 2006. *La gestión, los procesos y las metodologías de desarrollo de software. .* 2006.

PRESSMAN, R. 2002. *Ingeniería del software, un enfoque práctico.* 2002.

—. 2002. *Ingeniería del software, un enfoque práctico. S.l.: s.n.* 2002.

Rivasi, Javier. 2009. Pruebas funcionales. [Online] 2009. [Citado em: 2 de abril de 2017.] http://www.calidadysoftware.com/testing/pruebas_funcionales.php..

RIVERA ALVA, E. 2017. *Arquitectura-de-Software-II-Diagrama-de-Componentes-y-Despliegue.* 2017.

RODRÍGUEZ SÁNCHEZ, T.. 2014. *Metodología de desarrollo para la Actividad productiva de la UCI.* 2014.

Romero, Alex Méndez. 2016. *El framewrok Bootstrap sus usos y ventajas.* 2016.

Rosabal, Rodisbel R. Barzaga. 2005. Pruebas de software. [Online] 2005. [Citado em: 7 de abril de 2019.] www.informatica-juridica.com/wp-content/uploads/2014/01/Estrategia_pruebas_aplicaciones_web.pdf.

Russo, Patricia. 2011. *Gestion documental en las Organizaciones.* Barcelona : s.n., 2011.

Sarmiento, J. 2016. *UML: Diagrama de despliegue. Obtenido de Visión general de los diagrama de despliegue. .* 2016.

—. 2016. UML: Diagrama de despliegue. Obtenido de Visión general de los diagrama de. [Online] 2016. [Citado em: 4 de febrero de 2019.]

Schaferhoff, Nick. 2019. websitesetup.org. [Online] 25 de abril de 2019. [Citado em: 14 de enero de 2019.]

Sommerville, Ian. 2011. *INGENIERIA DEL SOFTWARE. UN ENFOQUE PRACTICO.* 2011.

Toll, Yuniet y Mendoza, Yilennis. 2007. *Propuesta de manual de procedimiento de Pruebas de Sistema y su aplicación en el Proyecto CICPC.* 2007.

Valdés, Damián Pérez. 2007. Maestros del web. [Online] 3 de julio de 2007. [Citado em: 25 de noviembre de 2018.] http://www.maestrosdelweb.com/que-es-javascript/.

VANDYK, J. 2011. *An Introduction to Drupal Architecture.* Iowa : s.n., 2011.

Visual Paradigm. 2015. VisualParadigm. Visual Paradigm. *Sitio web oficial de Visual Paradigm.* [Online] 2015. [Citado em: 27 de enero de 2019.]

Yanover, David Alejandro. 2016. www.mastermagazine.info. [Online] 2016. [Citado em: 13 de enero de 2019.] https://www.mastermagazine.com.br/.

ANEXOS

Anexo 1. Entrevista al cliente para conocer la necesidad del desarrollo de la propuesta de solución y definir los requisitos funcionales y no funcionales.

Estimado profesor: Se necesita de su cooperación en una investigación para una tesis de pregrado. Por ello, sería de gran ayuda que respondiera lo siguiente:

1. ¿Se preparan guardias especiales o diferenciadas para algunos trabajadores? ¿Atendiendo a que aspectos?
2. Cuál es la frecuencia con la que un trabajador cualquiera del centro realiza su guardia?
3. ¿Se planifica la guardia los fines de semana? ¿Es de la misma forma la planificación para los fines de semana (especificar si es todo el día o por horarios porque entonces más de un trabajador tuviera la guardia el mismo día)?
4. ¿Solo hacen guardia los trabajadores internos?
5. ¿En qué áreas se lleva a cabo la guardia obrera?
6. ¿Se planifica por turnos la guardia obrera?
7. ¿Existe algún trabajador que este eximido de la guardia obrera?
8. ¿Qué afectación trae para la planificación que se altere el proceso de la guardia?
9. ¿Cómo se planifica la guardia?
10. ¿Cómo se controla la guardia?
11. ¿Cuáles son las principales funcionalidades que se desea informatizar?

Anexo 2. Casos de prueba.

Tabla 9 Caso de prueba: Añadir equipo de guardia

Caso de prueba Añadir equipo de guardia
Nombre de requisito: Añadir equipo de guardia
Nombre de la persona que realiza la prueba: Daniela Herrera González
Descripción de la prueba: Prueba a la funcionalidad añadir equipo de guardia.
Entrada/Pasos de ejecución: Se seleccionan los siguientes datos para crear la planificación.
Nombre **Periodo-Horario** **Integrante 1** **Integrante 2** El administrador presiona el botón aplicar y si los datos están correctos se crea el equipo de guardia que podrá ser visualizado desde la vista Equipo de guardia para los usuarios autenticados en el sistema, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.
Evaluación de la prueba: Satisfactoria.

Tabla 10 Caso de prueba: Añadir incidencia

Caso de prueba Añadir incidencia
Nombre de requisito: Añadir incidencia
Nombre de la persona que realiza la prueba: Daniela Herrera González
Descripción de la prueba: Prueba a la funcionalidad añadir incidencia.
Entrada/Pasos de ejecución: Se seleccionan los siguientes datos para crear la incidencia.
Titulo **Fecha**

Descripción **Trabajador/Equipo de guardia** **Turno** El administrador presiona el botón aplicar y si los datos están correctos se crea el incidencia que podrá ser visualizado desde la vista Incidencias para el administrador, si existe algún dato incorrecto el sistema mostrará un mensaje de error y señalará el campo erróneo en rojo para su posterior corrección.
Evaluación de la prueba: Satisfactoria.

Anexo 3. Capturas de pantalla del sistema.

Figura 23. Vista equipo de guardia

Figura 24. Vista trabajadores

Anexo 4. Carta de aceptación emitida por el cliente donde se expresa la conformidad por parte del mismo con el sistema desarrollado y las funciones que este realiza.

UCI
Universidad de las Ciencias Informáticas

FACULTAD 1

DICTAMEN CRÍTICO

La Universidad de las Ciencias Informáticas (UCI) es un centro cubano de altos estudios con la misión de: formar profesionales comprometidos con su Patria y altamente calificados en la rama de la Informática, producir aplicaciones y servicios informáticos, a partir de la vinculación estudio – trabajo como modelo de formación y servir de soporte a la industria cubana de la informática.

En función de cumplir con la misión encomendada, la UCI cuenta con una planificación de la guardia obrera para cada uno de los trabajadores de cada facultad. Con el objetivo de informatizar dicho proceso y facilitar el trabajo a la persona encargada de realizarlo se desarrolló el Sistema de gestión de la Guardia Obrera (GO) para la Facultad 1 de la UCI. Dicho sistema permite planificar la guardia de forma automática, registrar incidencias, así como generar reportes y notificar vía correo a los trabajadores en una fecha próxima al día de guardia para evitar que incurra en incidencias.

Con el desarrollo de este sistema, se brindó al VDEA una mejora para realizar el proceso de forma óptima. El sistema representa de igual forma una ventaja para que los trabajadores puedan interactuar con el mismo, permitiéndoles conocer información importante referente al proceso, pueden tener conocimiento con quien forman equipo de guardia en que turno y que día le corresponde realizar la misma.

Para que así conste, firma la presente a los 3 días del mes de junio del año 2019.

Nombre y apellidos: Leovan Pérez Serrano

Responsabilidad: Vicedecana de Economía y Admón

Firma:

UCI
FACULTAD 1

Anexo 5. Reconocimiento obtenido en la Jornada del Ingeniero en Ciencias Informáticas con la ponencia del presente trabajo de diploma.