Facultad 5

Trabajo de Diploma para Optar por el Título de
Ingeniero en Ciencias Informáticas

Título:

Aplicación Web para la Gestión de la Guardia Obrera en la Facultad 5.

Autor: Hendrik Guirado Fuentes
Tutores: MsC. Zoraida Fernández Guevara
Ing. Dagmar Mir Leyva

La Habana, Cuba
Junio, 2015

Declaración de autoría

Por este medio se declara que soy el único autor de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente.

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

_____	_____
Firma del Autor	Firma del Tutor
Hendrik Guirado Fuentes.	Ing. Dagmar Mir Leyva.

Síntesis del tutor

Dagmar Mir Leyva: graduado con título de oro de Ingeniero en Ciencias Informáticas en la Universidad de la Ciencias Informáticas (UCI), se desempeña como especialista en el Centro de Consultoría y Desarrollo de Arquitecturas Empresariales (CDAE). Cuenta con 2 años de experiencia vinculado a proyectos de desarrollo.

Correo electrónico: dmir@uci.cu

Síntesis del tutor

Zoraida Fernández Guevara: es Filóloga en Lengua Rusa, licenciada en Educación en la especialidad de Lengua Inglesa y Máster en Tecnología de los procesos educativos. Posee 30 años de experiencia en la enseñanza superior. Ha impartido asignaturas tales como Idioma Ruso e Inglés, Comunicación Profesional, Formación Pedagógica, Multimedia, para estudiantes no filólogos. Ha impartido un gran número de cursos de postgrado, varios de ellos preparados por ella. Ha participado en investigaciones pedagógicas. Ha presentado trabajos en eventos nacionales e internacionales. Ha publicado artículos y ponencias en memorias de eventos.

Correo electrónico: zorlis@uci.cu

AGRADECIMIENTOS

A mi madre y hermana por ser lo que me da fuerzas siempre para seguir adelante por difícil que parezca el camino.

A mi padre por estar siempre dispuesto a ayudarme y por esperar pacientemente por mis éxitos.

A toda mi familia sepan que tienen un lugar especial para mí, los quiero.

A mis amigos por considerarlos extensión de mi familia, siempre estaré para ustedes.

A mi tutor Dagmar Mir Leyva, por asumir el difícil reto, este es un éxito de ambos, nadie lo hubiese podido hacer mejor.

A mi tutora Zoraida Fernández Guevara, sus palabras de despedida fueron importantes para lograr este objetivo.

A la profesora Susej Beovides por su tiempo, sus consejos y su paciencia.

A toda persona que se alegre de mis éxitos, sobre todo a los profesores que tuvieron un papel importante en mi formación profesional.

DEDICATORIA

En ocasiones no sabemos o es imposible retribuir tanto amor, tanto sacrificio, tanta dedicación. A veces nos cuesta decir un simple te quiero, te extraño, te amo. Quiero aprovechar esta oportunidad para intentar agradecer todo lo que ha hecho por mí, pero estoy seguro que ni la vida me alcanzará. Este éxito y todos los demás que pudieran venir serán dedicados con todo el amor que exista a ella, mi mayor motivación, mi mayor inspiración, por darme esas lecciones de amor y sacrificio infinitos. Por ser esa persona que siempre incluso cuando creo que todo ha salido mal inconscientemente me hace pensar que solo es algo circunstancial porque la tengo a ella. Por ser el centro de mi vida dedico esta tesis a mi madre.

A mi hermanita Herlin, esa mujercita que por el solo hecho de pensar que pudiera ser un ejemplo para su vida me ha hecho aumentar mis ambiciones y con ellas mis sacrificios y ahora mismo me hace sentir la satisfacción del deber cumplido, decirle que la clave está en disfrutar el ascenso a cada meta y cada día como si fuese el último, por ser junto a mi madre la mayor motivación al éxito de mi vida.

A mi padre por darme su cariño siempre, por sus consejos que hasta en los momentos de rebeldía han tenido un fuerte impacto, por las lecciones de la vida que me ha dado con su forma de actuar, por ser mi héroe de la niñez y una gran inspiración para tratar de mantener o corregir el camino en mi adultez. Por su colaboración excepcional en pro de convertirme en una mejor persona, profesional y hombre.

ÍNDICE

INTRODUCCIÓN

La Universidad de las Ciencias Informáticas (UCI), tiene entre sus objetivos la informatización de todas sus áreas, con el propósito de mejorar el funcionamiento de los procesos que tienen lugar en el centro. El área de Extensión y Residencia, identificada como una de las áreas de resultados claves en el desarrollo creciente de la universidad, se sustenta sobre la base de la planificación, organización y control de todas las actividades no curriculares realizadas por los estudiantes y trabajadores internos que en ella conviven.

En nuestra casa de altos estudios existen disímiles medios y recursos materiales que permiten a los estudiantes y trabajadores mejorar sus condiciones de vida, por lo que debe ser interés de todos mantener una preservación constante de los mismos, una forma de evitar o minimizar probabilidades de existencia de algún tipo de hecho que ponga en riesgo dichos recursos o que interfiera con la tranquilidad ciudadana reinante en nuestro centro, es la realización de la Guardia Obrera (GO).

La Facultad 5 de la UCI y específicamente su Vicedecanato de Extensión mantiene un estricto seguimiento y control de las actividades relacionadas con (GO) realizándola de forma manual, es imprescindible mejorar la factibilidad y comodidad del sistema, tanto para los trabajadores que realizan la guardia, como del personal encargado de la planificación, distribución y control de la misma. Esta es una tarea importante a la que no se le ha dado aún una solución satisfactoria para todos.

Actualmente el vicedecano de extensión recepciona toda la documentación en copia dura (papel) o en formato digital (Excel, Word etc.) del cumplimiento de la (GO). Esta información es recibida a través de las personas asignadas como responsables en estos tipos de controles. Estos datos llegan a la secretaria vía correo, a través de dispositivos de almacenamiento portables o en las plantillas impresas creadas con estos fines o entregadas personalmente.

Por tanto, la **situación problemática** es la siguiente:

- No se tiene como visualizar el desempeño de la guardia en tiempo real o resulta demasiado engorroso el trabajo.
- Se dificulta el trabajo de conservación de las evaluaciones.
- El procesamiento de los datos se hace a medida que se van recepcionando, esto dificulta y produce atraso en la disponibilidad de la información obtenida.
- En ocasiones se creen ficheros duplicados por no llevar un correcto control de versiones.

- Se hace difícil la posibilidad de que varios usuarios puedan acceder a los datos almacenados simultáneamente.

Por lo anterior se define el siguiente **problema científico**: ¿Cómo organizar el proceso de la Guardia Obrera para mejorar la gestión de su planificación y control?

El **objeto de estudio** se enmarca en la gestión de la planificación y control de la guardia obrera.

Se define como **campo de acción** los sistemas para la gestión de la planificación y control de la guardia obrera.

Para solucionar el problema planteado se define como **objetivo general** del presente trabajo de diploma: Desarrollar una aplicación para la gestión de la planificación y control de la guardia obrera en la Facultad 5.

Para lograr el cumplimiento del objetivo general se definen las siguientes **tareas investigativas**:

- Elaboración del marco teórico de la investigación a partir del estado del arte actual referente al tema.
- Selección de las herramientas, patrones, arquitectura y metodología para el desarrollo de la aplicación.
- Realización de diagnóstico sobre la situación actual de la gestión de información y planificación de la Guardia obrera en la Facultad 5.
- Definición y modelación de los procesos relacionados con la (GO) en la Facultad 5.
- Diseño del modelo de datos para sustentar la persistencia de la solución informática.
- Selección y aplicación de pruebas a la solución con el objetivo de validar la calidad del diseño realizado, las funcionalidades del sistema y el cumplimiento de las necesidades del cliente corrigiendo los problemas detectados.

Resultado a obtener

Aplicación para la gestión de la planificación y control de la guardia obrera en la Facultad 5. Para la realización de este trabajo se contó con el apoyo de los siguientes métodos científicos de la investigación:

Métodos Teóricos

- Analítico – Sintético: este método permitió analizar bibliografía sobre los sistemas de gestión, así como profundizar en sus características y ventajas fundamentales, para apoyar el desarrollo de la investigación.
- Histórico – Lógico: se utilizó para realizar un análisis de los conceptos y sistemas relacionados con la gestión de las guardias obreras, analizando su evolución en el tiempo.
- La modelación: se utilizó para la creación de abstracciones que explican la realidad, ejemplo de esto se puede ver en los modelos y diagramas presentados en la fase de diseño.

Métodos Empíricos

- Consulta de la información en todo tipo de fuentes**:** para la elaboración del marco teórico de la investigación se utilizó este método en aras de obtener aspectos similares reutilizables en la propuesta de solución.
- Entrevista**:** este método jugó un papel fundamental en la validación del problema a resolver, permitiendo a un grupo de expertos examinar las facilidades que brinda la aplicación web para la gestión de la planificación y control de la guardia obrera en la Facultad 5.

Como **aporte práctico** se desarrollará una aplicación web para la gestión de la planificación y control de la Guardia Obrera en la Facultad 5.

De ahí que esta investigación se plantee la siguiente **idea a defender**: EL desarrollo de una aplicación web para la gestión de la Guardia Obrera de la Facultad 5 contribuirá a su planificación y control.

Estructura en capítulos del documento

Capítulo 1: Fundamentación teórica

Se presentan los elementos teóricos que sirven de base a la investigación del problema planteado. Se realiza un estudio del estado del arte, mediante un análisis de sistemas de software existentes para la gestión de la guardia obrera tanto en la Universidad de las Ciencias Informáticas, como fuera de ella. Se fundamenta la necesidad de desarrollar una aplicación web para la gestión de la planificación y control de la Guardia Obrera en la Facultad 5, y por último se realiza una breve descripción de las herramientas, tecnologías y metodología a utilizar para dar solución al problema planteado.

Capítulo 2: Características y diseño de la aplicación

Se abordan las fases de Exploración y Planificación, propias de la metodología de desarrollo utilizada en la implementación de la aplicación.

Capítulo 3: Implementación y prueba de la aplicación

Se aborda la fase de Producción, donde se detallan las iteraciones llevadas a cabo durante la etapa de implementación de la aplicación. Se describen las pruebas realizadas para comprobar su funcionamiento.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

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

En el presente capítulo se dará una explicación acerca de diferentes sistemas para planificar guardias existentes. De estos sistemas se abordarán sus principales características para poder identificar los puntos reutilizables y basarse en ellos para ayudar en la solución del problema en cuestión. Por lo que uno de los objetivos que se persigue en el capítulo es la realización de un estudio referente al estado del arte y a los principales conceptos que estarán presentes en la investigación.

1.2 Proceso de la planificación y el control de la guardia obrera

La planificación de la GO es un proceso clave en la UCI, de su realización y calidad depende prevenir, detectar, retardar y neutralizar la ocurrencia de amenazas, así como reducir los niveles de riesgos que puedan dar origen a hechos y actividades delictivas de carácter común. Esta tarea tiene lugar en diferentes áreas de la UCI y en ella intervienen tanto estudiantes como trabajadores. Para esto se realiza una planificación de guardia para un periodo de tiempo determinado.

El encargado de planificar la guardia tiene la misión de gestionar la planificación teniendo en cuenta personal disponible, horarios, áreas y postas distribuidas en diferentes áreas de la Facultad 5 de la UCI. Una vez realizado esto se realizan ajustes según se estimen convenientes y se da a conocer la información a los involucrados en el proceso. Para ello es necesario emplear gran cantidad de tiempo en distribuir correctamente a cada uno de los involucrados, teniendo en cuenta como elemento fundamental que no haya coincidencias en el otorgamiento de las planificaciones de las guardias.

Una vez planificada la guardia se lleva a cabo el proceso de control. Para esta tarea se destina a un grupo determinado de controladores jefes de la guardia. Después del chequeo se emite una evaluación la cual se evalúa de manera cualitativa con una ponderación de: Bien, Regular o Mal y los no evaluados (N/E), se pueden reportar incidencias en caso de detectar anomalías.

1.3 Análisis de soluciones existentes

Actualmente se pueden encontrar diversos sistemas que representan soluciones afines a programas para la gestión y planificación de la guardia obrera en una empresa o institución. En este epígrafe se realiza un análisis sobre algunas de las aplicaciones informáticas existentes dentro y fuera del país que realiza este tipo de procesos, con el objetivo de buscar características comunes que puedan ser tomadas como apoyo para el desarrollo de la solución que se propone.

1.3.1 Análisis de soluciones desarrolladas en la UCI

Gestor Web para el control de la Guardia Obrera

Aplicación desarrollada en la UCI, en la Facultad 4 la cual actualmente no está en funcionamiento. Este sistema tiene en cuenta características específicas de la facultad donde fue creado. Genera reportes referentes al cumplimiento o no de la guardia, aunque estos pueden ser generados en un periodo de tiempo específico (semanal, mensual), no son los únicos reportes necesarios en un proceso como este. El no poder acceder al código fuente de esta aplicación web imposibilita la reutilización o adaptación de la misma para poder dar solución a la problemática planteada (3).

Sistema de gestión de información del Área de Extensión Universitaria FAC2

Entre los procesos de este sistema se encuentra la gestión de la guardia estudiantil, esta está habilitada con las características específicas de la facultad 2 de la UCI y no planifica guardia para los trabajadores. La imposibilidad de acceder al código fuente de esta aplicación web hace que la reutilización o adaptación de la misma para poder dar solución a la problemática planteada sea imposible (4).

Portal de la antigua Facultad 7

El sitio de la Facultad 7 de la Universidad de las Ciencias Informáticas tiene como objetivo proporcionarle a la dirección de la facultad la posibilidad de tener registrada toda la información necesaria referente a los procesos de la guardia estudiantil y cuartelaría.

Características:

- Proporciona que se pueda llevar un control personalizado de la evaluación de los estudiantes en la residencia estudiantil.
- Genera un conjunto de reportes, en formato digital, acerca de estas dos actividades, que pueden ser impresos.
- Facilita la planificación y el control de la guardia estudiantil y la cuartelería a los responsables de estas tareas.
- Posibilita realizar de forma más eficiente el manejo y gestión de la información de estas actividades.(5)

1.3.2 Análisis de otras soluciones existentes en Cuba

SIDATRAT

Analizando los datos que proporciona Internet se encuentra la propuesta de un producto Web que incorpora algunas de las funcionalidades que se propone para una versión final del producto.

El Sistema Automatizado para la Atención Médica Integral a Pacientes (“SIDATRAT”) desarrollado en el

Instituto de Medicina Tropical “Pedro Kourí” en Cuba, propone en uno de sus módulos llevar el plan y control de la guardia de los médicos del hospital en el Centro Receptor de Ingresos (CRI) y realizar reportes respecto a esta. El mismo ha sido desarrollado utilizando las ventajas que ofrecen los lenguajes de programación para Web, tales como: PHP, HTML, JavaScript y Visual Basic Script, y una base de datos en MySQL. (6)

Este sistema propone:

- El usuario debe introducir su identificación y clave para ser reconocido por el sistema. Una vez que ha entrado al mismo, toda la información relacionada con el acceso es almacenada.

- El menú de SIDATRAT se construye de manera dinámica, en dependencia de los privilegios otorgados al usuario que ha entrado al sistema.
- El sistema permite confeccionar cualquier listado que necesiten los usuarios sobre los pacientes registrados.
- En cuanto al manejo de los errores, los mismos son analizados antes de enviar los datos al servidor. La información será presentada de forma tal que los usuarios escojan el dato que quieren introducir y tecleen lo menos posible.
- El sistema ofrece un grupo de listados de errores que pueden ocurrir en diferentes momentos.
- Cada usuario es registrado con una clave encriptada y se le definen sus niveles de acceso y tareas específicas, no pudiendo acceder a información no autorizada.

1.3.3 Análisis de soluciones existentes fuera de Cuba

Tecno hospital.

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

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

Características:

- Genera automáticamente la planificación en base a unos parámetros definidos inicialmente como: área, categoría, número de horas máximas según los convenios de enfermería, número de horas máximas de guardia, o número mínimo y máximo de enfermeros necesarios por turno.
- Ofrece un portal de usuarios con servicios para los enfermeros y enfermeras.
- Permite intercambiar de forma sencilla los turnos entre empleados, modificar horarios, establecer turnos personalizados o trabajar con turnos predefinidos.

- Permite consultar en todo momento si un turno está cubierto de acuerdo a los parámetros predefinidos y, en caso de faltar personal, propone una lista de trabajadores para cubrir dicho turno.(7)

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

OPTIHPER.

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

Características:

- Permite adaptar, modificar y validar interactivamente una planificación previa ante incidencias o cambios posteriores (reactividad on-line).
- Incluye interfaces expresivas y amigables. Así mismo, puede adquirir los datos necesarios directamente desde bases de datos estándar. Genera automáticamente informes gráficos y textuales, con información general o particularizada.
- Facilita la reubicación y cambio de turnos, pudiendo ser auto-gestionado por los propios trabajadores, satisfaciendo las condiciones y restricciones existentes y bajo supervisión de la empresa. En caso de imposibilidad de atender los cambios propuestos, el sistema puede proveer de alternativas.
- Re-planifica la asignación de tareas u horarios en caso de incidencias.
- Está implementado en ANSI C, es multiplataforma (unix, linux, windows, etc.) y no hace uso de rutinas o código protegido por terceros.(8)
    - Este sistema incluye dentro de sus funcionalidades la planificación de horarios y/o turnos al personal, permitiendo escoger el periodo, ya sea semanal, mensual o anual. Realiza la reubicación

y cambio de turnos, notifica con antelación los turnos de trabajo y las tareas concretas a realizar, pero no permite la configuración de los turnos y las postas y la realización del control de la guardia.

1.3.4 Valoración de las soluciones existentes analizadas.

Después del análisis realizado, se arriba a la conclusión de que los sistemas estudiados representan una fuente de valor para la investigación y sirven de sostén para la elaboración de la misma, aportando conocimiento sobre las funcionalidades básicas a tener en cuenta. No obstante, no se toman como solución ya que no cumplen con los requisitos establecidos.

Por lo planteado anteriormente, se hace necesario desarrollar una aplicación web que responda acertadamente a las necesidades específicas de la gestión y planificación de la GO en la Facultad 5.

1.4 Herramientas y tecnologías

1.4.1 Lenguajes de programación

Desde los inicios de Internet, fueron surgiendo diferentes demandas por los usuarios y se dieron soluciones mediante lenguajes estáticos. Las tecnologías han ido desarrollándose y surgidos nuevos problemas a los que es necesario dar solución. Los lenguajes dinámicos de programación para la web -surgidos a partir de esta situación- permiten interactuar con los usuarios y utilizar sistemas de Bases de Datos.

JAVA

Es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de Lenguaje de Programación C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Las aplicaciones Java están típicamente compiladas en un bytecode, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el bytecode es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del bytecode por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrolladas por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del Java Community Process, otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre (9).

Python

Es un lenguaje de programación interpretado creado por Guido van Rossum en el año 1991. Se compara habitualmente con TCL, Perl, Scheme, Java y Ruby. En la actualidad Python se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation. La última versión estable del lenguaje es la 3.0.Python es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa. Los usuarios de Python consideran a éste mucho más limpio y elegante para programar.

Permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas (o como ejemplos para empezar a aprender Python). También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, sockets y hasta interfaces a GUI (interfaz gráfica con el usuario).

Por ser un lenguaje de programación interpretado ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa. (10)

PHP

Es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica. [11]

Ventajas

- Es un lenguaje multiplataforma.

- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad.
- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida

Lenguaje seleccionado

Luego del análisis de los lenguajes anteriores se decide utilizar para el desarrollo del sistema el lenguaje **JAVA**. Además de todas las ventajas expuestas con anterioridad una de las razones fundamentales por la que se escogió es por ser uno de los lenguajes más usados en la universidad y del que se posee una amplia documentación.

1.4.2 Otros lenguajes utilizados

CCS

Las hojas de estilo en cascada (Cascading Style Sheets, CSS) son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). Permite separar la estructura de un documento de su presentación. La información de estilo puede ser adjuntada tanto como un documento separado o en el mismo documento HTML. En este último podrían definirse estilos generales en la cabecera del documento o en cada etiqueta particular mediante el atributo "style". (12)

Las Ventajas de utilizar CSS son:

- Control centralizado de la presentación de un sitio web completo con lo que se agiliza de forma considerable la actualización del mismo.

- Los Navegadores permiten a los usuarios especificar su propia hoja de estilo local que será aplicada a un sitio web, con lo que aumenta considerablemente la accesibilidad.
- Una página puede disponer de diferentes hojas de estilo según el dispositivo que la muestre o incluso a elección del usuario.
- El documento HTML en sí mismo es más claro de entender y se consigue reducir considerablemente su tamaño.

HTML 5

HTML (Hyper Text Markup Language) en su versión 5, no sólo trata de incorporar nuevas etiquetas o eliminar otras, sino que supone mejoras en áreas que hasta ahora quedaban fuera del lenguaje y para las que se necesitaba utilizar otras tecnologías.

Los cambios comienzan añadiendo semántica y accesibilidad implícitas, especificando cada detalle y borrando cualquier ambigüedad. Se tiene en cuenta el dinamismo de muchos sitios webs, donde su aspecto y funcionalidad son más semejantes a aplicaciones webs que a documentos.

También cuenta con nuevas APIs para interfaz de usuario: temas tan utilizados como el "drag&drop" (arrastrar y soltar) en las interfaces de usuario de los programas convencionales, serán incorporadas al HTML 5 por medio de un API.

JavaScript

Es un lenguaje de programación que se puede utilizar para construir sitios web y para hacerlos más interactivos. Aunque comparte muchas de las características y de las estructuras del lenguaje Java, fue desarrollado independientemente. El lenguaje JavaScript puede interactuar con el código HTML, permitiendo a los programadores web utilizar contenido dinámico. Por ejemplo, hace fácil responder a los acontecimientos iniciados por usuarios (como introducción de datos en formularios) sin tener que utilizar interfaz de entrada común (CGI). El lenguaje JavaScript es de código abierto, por lo que cualquier persona puede utilizarlo sin comprar una licencia. (13)

1.4.3 Frameworks de desarrollo

Luego de haber seleccionado como lenguaje de programación java, se estudiaran algunos de los frameworks de desarrollo que se utilizan para este lenguaje, con el objetivo de determinar cuáles brindan mayores ventajas para el desarrollo de la solución que se propone.

JavaServer Faces

Es una tecnología y framework para aplicaciones Java basadas en web que simplifica el desarrollo de interfaces de usuarios en aplicaciones Java EE. **JSF** usa JavaServer Pages (JPS como la tecnología que permite hacer el despliegue de las páginas, pero también se puede acomodar a otras tecnologías como XUL (acrónimo de XML-based User-interface Language - lenguaje basado en XML para la interfaz de usuario)

JSF incluye:

- Un conjunto de APIs para representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Un conjunto por defecto de componentes para la interfaz de usuario.
- Dos bibliotecas de etiquetas personalizadas para JavaServer Pages que permiten expresar una interfaz JavaServer Faces dentro de una página JSP.
- Un modelo de eventos en el lado del servidor.
- Administración de estados.
- Beans administrados.

Existen diferentes versiones de Java Server Faces entre las que se encuentra PrimeFaces. Esta es una biblioteca de componentes para JSF de código abierto que cuenta con un conjunto de componentes enriquecidos que facilitan la creación de las aplicaciones web. PrimeFaces está bajo la licencia de Apache License V2.

Vaadin

- Es un framework para el desarrollo de la capa de presentación de aplicaciones web en Java, que ofrece una interfaz de escritorio tradicional, sin necesidad de usar JavaScript ni JSON ni XML ni siquiera HTML.
- Está orientado a componentes y eventos.
- Se programa en el servidor. La definición del interfaz (páginas) se realiza en el servidor utilizando Java (se elimina JavaScript). Todo se compila y se puede depurar.

- Trae una serie de componentes de interfaz definidos (parte servidor), y permite crear nuevos por composición de los que ya hay incluyendo llamadas al negocio.
- Permite extender un widget para hacer uno propio (parte cliente), así como crear uno nuevo, usando GWT (compilador de Google para elementos de la capa de presentación).
- Puede integrarse con cualquier otro framework Java, como JEE (CDI), Spring, Guice, etc.
- Permite exportar los componentes propios e importar componentes desarrollados por terceros mediante un sistema de Add-Ons.
- Ofrece un plugin para Eclipse para generar proyectos, compilar widgetset (GWT), etc.(14)

Hibernate

Es una herramienta de Mapeo Objeto-Relacional (ORM) para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) o anotaciones en los beans de las entidades que permiten establecer estas relaciones.

- Hibernate es de software libre.
- Permite al desarrollador detallar cómo es su modelo de datos, qué relaciones existen y qué forma tienen.
- Manipula los datos de la base operando sobre objetos, con todas las características de la POO.
- Convierte los datos entre los tipos utilizados por Java y los definidos por SQL.
- Genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias, manteniendo la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución. (15)

Spring

Spring es un framework open source que proporciona un marco de trabajo para el desarrollo de aplicaciones J2EE. Está basado en el uso de ficheros planos JavaBeans para la lógica de aplicación y archivos XML para la configuración. No obliga a usar un modelo de programación en particular. Se ha popularizado en la comunidad de programadores en Java. Considerado una alternativa y sustituto del modelo de Enterprise JavaBean. Por su diseño el framework ofrece mucha libertad a los desarrolladores

en Java y soluciones muy bien documentadas y fáciles de usar para las prácticas comunes en la industria. Las principales características de Spring son las siguientes:

- Usa implementaciones ORM de terceros como Hibernate, Ibatis, JDO, OJB.
- Provee servicios de aplicación en forma declarativa.
- Permite usar archivos XML de configuración, programación mediante la API y mediante un estándar JSR.
- Aporta integración con la solución de código abierto Acegi (soporta seguridad declarativa basada en el uso de IoC y AOP).
- Permite el uso de cualquier servicio usando un archivo XML de configuración.
- Todos los componentes pueden ser testeados fuera del contenedor.
- Soporta JTA, Hibernate, JDO, JDBC, ODBC.

Ventajas de Spring

- Spring recomienda el uso de Interfaces, si se decide no utilizar Spring, basta con re implementar las interfaces.
- Spring se puede ejecutar dentro de un contenedor Web o fuera de él en una aplicación Swing.

Desventajas de Spring

- No se puede evaluar si un objeto ha sido bien inyectado más que en tiempo de ejecución.
- El contenedor de Spring no es ligero (si se usan todos los módulos disponibles), no es recomendable su uso en aplicaciones de tiempo real o en aplicaciones para móviles. (16)

Struts

Struts es un framework de aplicación web open source desarrollado por Apache. Struts está basado en el patrón MVC. Se utiliza para construir aplicaciones Web basadas en servlets y JSP que pueden ejecutarse en cualquier contenedor de servlets incluyendo los servidores de aplicaciones J2EE. Mediante Struts se hace uso de patrones de diseño J2EE que son útiles en la construcción de aplicaciones J2EE. Y puesto que Struts sigue los patrones incluidos en el MVC, este framework es relativamente simple de usar y aprender.

Struts proporciona al desarrollador un conjunto de etiquetas JSP personalizadas que facilitan la

integración del framework con las páginas JSP.

Las principales características de Struts son las siguientes:

- Modelo Correcto. Permite modelar las aplicaciones basadas en JSP y servlets mediante el uso del patrón de diseño MVC.
- Objetos “listos para usar”. Struts incorpora el concepto de objetos “listos para usar” a través de ficheros de configuración en XML.
- Patrones de Diseño pre-construidos. Struts incluye patrones de diseño internos en el framework. Por lo que no se ha de preocupar de crear los propios patrones de diseño.
- Características Extras. Incorpora características extras como validación, internacionalización, etc.

Ventajas de Struts

- Transporte automático de los datos introducidos en el cliente (JSP) hasta el controlador (Action) mediante formularios (ActionForm).
- Transporte automático de los datos enviados por el controlador (Action) a la parte de presentación (JSP) mediante formularios (ActionForm).
- Implementa la parte común a todas las aplicaciones en la parte de Controlador (ActionServlet); la parte particular de cada aplicación es fácilmente configurable (struts-config.xml).
- La separación de los componentes en capas (MVC) simplifica notablemente el desarrollo y su mantenimiento.

Desventajas de Struts

- El hecho de no abarcar todas las capas de la aplicación web (deja fuera la capa de negocio y la capa de persistencia) hace que el interfaz entre Struts y estas capas no esté tan automatizado, convirtiendo los accesos a los datos (DAO) en monótonos de desarrollar. (16)

Frameworks seleccionados

Luego del análisis de los frameworks anteriores atendiendo a su comodidad, integración y a la amplia documentación existente se decide utilizar para el desarrollo del sistema los siguientes:

- JavaServer Faces(JSF)
- Hibernate

- Spring

1.5 Metodologías de desarrollo de software

Metodología es un proceso de software detallado y completo. Las metodologías de desarrollo de software se han clasificado en: ágiles y tradicionales. Las metodologías ágiles enfatizan la comunicación con el cliente mientras que suelen ser señaladas por la falta de documentación técnica. Las tradicionales o pesadas son aquellas con mayor énfasis en la planificación y control del proyecto. ()
Entre las metodologías ágiles se destacan XP (eXtreme Programming), OpenUP (Open Unified Process), Scrum y Crystal. Así mismo se puede mencionar RUP (Rational Unified Process) y MSF (Microsoft Solution Framework) como las más destacadas dentro de las metodologías pesadas.

1.5.1 Metodología XP (*Extreme Programming*, Programación Extrema)

Los principios y prácticas son de sentido común pero llevadas al extremo, de ahí proviene su nombre. A continuación se presentan las características esenciales de XP organizadas en los tres apartados siguientes: historias de usuario, roles y proceso.

Historias de Usuario

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Se trata de tarjetas de papel en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales.
Roles XP: Aunque en varias fuentes de información aparecen algunas variaciones y extensiones de roles XP, en este apartado se describirán los roles de acuerdo con la propuesta original de Beck:

- Programador.
- Cliente.
- Encargado de pruebas (Tester).
- Encargado de seguimiento (Tracker).
- Entrenador (Coach).
- Consultor.
- Gestor (Big boss).

Proceso XP: Un proyecto XP tiene éxito cuando el cliente selecciona el valor de negocio a implementar basado en la habilidad del equipo para medir la funcionalidad que puede entregar a través del tiempo. El ciclo de desarrollo consiste a grandes rasgos en los siguientes pasos:

- El cliente define el valor de negocio a implementar.
- El programador estima el esfuerzo necesario para su implementación.
- El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
- El programador construye ese valor de negocio.
- Vuelve al paso 1.

El ciclo de vida ideal de XP concibe seis fases:

- Fase I Exploración.
- Fase II Planificación de la Entrega.
- Fase III Iteraciones.
- Fase IV Producción.
- Fase V Mantenimiento.
- Fase VI Muerte del Proyecto.

1.5.2 Desarrollo Adaptable de Software (ASD)

Metodología impulsada por Jim Highsmith, que posee como principales características ser iterativa, orientada a los componentes de software y tolerante a los cambios. Tiene un ciclo de vida dividido en tres fases principales: especulación, colaboración y aprendizaje. En la primera fase se inicia el proyecto y se planifican las características del software, en la segunda fase se desarrollan las características y finalmente en la tercera fase se revisa su calidad, y se entrega al cliente (17).

1.5.3 SCRUM

Es una metodología desarrollada por Ken Schwaber, Jeff Sutherland y Mike Beedle, se enfoca en el hecho de que procesos definidos y repetibles solo funcionan para atacar problemas definidos y repetibles con personas definidas y repetibles en ambientes definidos y repetibles. Además se caracteriza por el desarrollo de reuniones a lo largo del ciclo de vida del proyecto, así como un desarrollo basado en

iteraciones, denominadas sprints, con una duración de 30 días cada una. Cada iteración se entiende como un incremento ejecutable que se muestra al cliente (18).

1.5.4 Fundamentación del uso de la metodología XP

La metodología a utilizar en este trabajo será XP, después de haber realizado el estudio y análisis de algunas metodologías de desarrollo, la misma es adaptable al software a desarrollar, así como a las condiciones de trabajo, de forma general. Plantea la comunicación y satisfacción del cliente como requisito principal. Lo más importante en XP es definir los requerimientos y las pruebas de calidad.
Esta metodología es flexible cuando los requerimientos sufren cambios- algo que sucede a menudo - y permite administrarlos de forma óptima. Uno de sus objetivos principales es potenciar al máximo el trabajo en grupo, donde los jefes de proyecto, los clientes y desarrolladores son parte del equipo y están involucrados en el desarrollo del software.

1.6 Conclusiones parciales del capítulo

Después de realizar un análisis del marco teórico de la investigación, se puede arribar a las siguientes conclusiones:

- Se realizó un estudio del estado del arte de los diferentes sistemas existentes que realizan trabajos similares al que se desea desarrollar en el presente trabajo, y se concluye que representan una fuente de valor para la investigación y el desarrollo de nuestro sistema, aportando conocimiento sobre algunas de las funcionalidades básicas a tener en cuenta.
- El análisis de varias herramientas de trabajo arrojan como resultado la selección de un entorno de desarrollo altamente integrado y compatible entre sí, con el objetivo de facilitar el desarrollo de nuestro sistema.
- Para guiar el proceso de desarrollo de software se seleccionó la metodología XP, la cual posibilita la realimentación continua entre el cliente y el equipo de desarrollo. Además se seleccionaron las herramientas y las tecnologías a utilizar durante el desarrollo de la solución.

CAPÍTULO 2: DESCRIPCIÓN DEL SISTEMA

2.1 Introducción

En el presente capítulo se obtienen un conjunto de artefactos de la metodología XP que serán generados en cada fase. Se definen los requisitos funcionales con los cuales debe cumplir la solución propuesta. Se describen las historias de usuarios, se propone la arquitectura, los patrones a utilizar y el modelo de datos del sistema.

2.2 Exploración

La metodología de desarrollo XP comienza con la fase de exploración. Durante esta etapa se realiza el proceso de identificación de las historias de usuario, estas constituyen uno de los artefactos más importantes que se generan en la metodología, pues son la forma en que se especifican los requisitos del sistema. El ciclo de vida de XP enfatiza en el carácter iterativo e incremental del desarrollo. Una iteración de desarrollo es un período de tiempo en el que se realiza un conjunto de funcionalidades determinadas, que en el caso de XP corresponden a un conjunto de historias de usuarios.

Historias de Usuarios

La historia de usuario (HU), se escribe desde la perspectiva del cliente, aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido de estas debe ser concreto y sencillo. Se realiza una por cada característica principal del sistema, es utilizada para hacer estimaciones de tiempo y para el plan de lanzamientos, reemplazan un gran documento de requisitos. Las historias de usuarios quedan estructuradas de la siguiente manera:

- **Nombre**: nombre descriptivo de la HU.

- **Prioridad**: grado de prioridad que le asigna el cliente a la HU en dependencia de su importancia. Los valores que puede tomar son: alta, media o baja.
- **Complejidad**: grado de complejidad que le asigna el desarrollador a la HU luego de analizarla. Los valores que puede tomar son: alta, media o baja.
- **Estimación**: unidades de tiempo estimadas por el desarrollador para darle cumplimiento a la HU.

- **Iteración**: número de la iteración en la cual será implementada la HU.
- **Descripción**: descripción simple que brinda el cliente sobre lo que debe hacer la funcionalidad en cuestión.

Adicionalmente a cada HU se le asigna un número para facilitar su identificación por parte del equipo de desarrollo.

Durante esta fase se identificaron treinta historias de usuarios, cada una de ellas respondiendo a las diferentes funcionalidades solicitadas por el cliente. A continuación se describen algunas de las historias de usuarios identificadas, para consultar las restantes historias de usuario ver Anexo #1.

Tabla 1. Adicionar trabajador

Historia de Usuario	
Número: 1	**Nombre:** Adicionar trabajador
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Media
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador adicionar un trabajador especificando los siguientes datos asociados al mismo; usuario, nombre, apellido, solapín y centro.	
Observaciones: Hace referencia al RF1.	

Tabla 2. Eliminar trabajador.

Historia de Usuario	
Número: 2	**Nombre:** Eliminar trabajador
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Media
Puntos Estimados: 0,2	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador eliminar un trabajador.	
Observaciones: Hace referencia al RF2.	

Tabla 3. Obtener listado de los trabajadores existentes.

Historia de Usuario	
Número: 3	**Nombre:** Obtener listado de los trabajadores existentes
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Baja

Puntos Estimados: 0,2	**Iteración Asignada:** 1
Descripción: El sistema deberá permitir al usuario mostrar en una tabla un listado con todos los trabajadores registrados en el sistema.	
Observaciones: Hace referencia al RF3.	

Tabla 4. Modificar los datos de los trabajadores.

Historia de Usuario	
Número: 4	**Nombre:** Modificar los datos de los trabajadores.
Prioridad en Negocio: Baja	**Complejidad en Desarrollo:** Media
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: El sistema deberá permitir al usuario con permisos de planificador modificar los datos de un trabajador determinado; usuario, nombre, apellido, solapín y centro de un trabajador seleccionado.	
Observaciones: Hace referencia al RF4.	

Tabla 5. Adicionar la planificación de la guardia.

Historia de Usuario	
Número: 21	**Nombre:** Adicionar la planificación de la guardia.
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Alta
Puntos Estimados: 1	**Iteración Asignada:** 3
Descripción: Permite al usuario con privilegios de planificador adicionar una planificación de la guardia, con el trabajador responsable, turno de guardia, posta en la que debe estar el trabajador, la fecha en que se debe realizar la guardia y si esta fue realizada o no.	
Observaciones: Hace referencia al RF21.	

Tabla 6. Modificar la planificación de la guardia.

Historia de Usuario	
Número: 22	**Nombre:** Modificar la planificación de la guardia.
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,4	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario con permisos de planificador modificar una planificación de la guardia seleccionada, modificando los datos del trabajador responsable, turno de guardia, posta en la que debe estar el trabajador, la fecha en que se	

debe realizar la guardia y si esta fue realizada o no.
Observaciones: Hace referencia al RF22.

2.3 Planificación

Durante la fase de planificación se realiza una estimación del esfuerzo que costará implementar cada historia de usuario. Esta se expresa utilizando como medida el punto. Un punto se considera como una semana ideal de trabajo donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción.

Es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. La planificación no debe ser estricta puesto que hay muchas variables en juego, debe ser flexible para poder adaptarse a los cambios que puedan surgir. Una buena estrategia es hacer planificaciones detalladas para unas pocas semanas y planificaciones mucho más abiertas para unos pocos meses (19).

2.3.1 Estimación del tiempo por las historias de usuarios

Para el desarrollo del sistema propuesto se realizó una estimación del esfuerzo para cada una de las historias de usuarios identificadas, llegándose a las conclusiones que se muestran en la siguiente tabla.

Tabla 7. Estimación del tiempo por las historias de usuarios.

No	Historia de Usuario	Puntos de estimación
1	Adicionar trabajador.	0,3
2	Eliminar trabajador.	0,2
3	Obtener listado de los trabajadores existentes.	0,2
4	Modificar los datos de los trabajadores.	0,3
5	Adicionar posta de guardia.	0,3
6	Eliminar posta de guardia.	0,3
7	Obtener el listado de las postas	0,2

	existentes.	
8	Modificar los datos de las postas.	0,3
9	Adicionar turno de guardia.	0,3
10	Eliminar turno de guardia.	0,3
11	Obtener el listado de los turnos de guardias existentes.	0,2
12	Modificar los datos de los turnos de guardias.	0,3
13	Adicionar área de guardia.	0,2
14	Eliminar área de guardia.	0,2
15	Obtener el listado de las áreas de guardia existentes.	0,2
16	Modificar los datos de las áreas de guardia.	0,3
17	Exportar el listado de trabajadores.	0,5
18	Exportar el listado de turnos.	0,3
19	Exportar el listado de áreas.	0,3
20	Exportar el listado postas	0,3
21	Adicionar la planificación de la guardia.	1
22	Modificar la planificación de la guardia.	0,4
23	Eliminar la planificación de la guardia.	0,3
24	Obtener la planificación de la guardia.	0,3
25	Exportar la planificación de la guardia.	0,4

26	Realizar permutas de guardia entre trabajadores.	0,6
27	Obtener el listado de las guardias realizadas por el usuario autenticado.	1
28	Obtener un listado de los ausentes a la guardia.	0,3
29	Exportar el listado de los ausentes a la guardia.	0,2
30	Autenticar usuario.	0,4
Total		**11**

El tiempo total estimado para el desarrollo de la aplicación es de 11 puntos de estimación, que equivalen a 11 o dos meses y tres semanas ideales de trabajo.

2.4 Plan de Iteraciones

Una vez identificadas las historia de usuario del componente y estimado el esfuerzo dedicado a la realización de cada una de estas se procede a la planificación de la etapa de implementación del proyecto. En base a lo antes mencionado se decide realizar esta etapa en tres iteraciones, las cuales se detallan a continuación.

Iteración 1

Esta iteración tiene como objetivo la implementación de las historias de usuario referentes al proceso de adicionar, eliminar, modificar y obtener los listados de los trabajadores, las postas y los turnos de guardia. Al finalizar esta iteración estarán creadas las bases para poder implementar las funcionalidades principales del sistema.

Iteración 2

El objetivo de esta iteración es la implementación de las historias de usuario referentes a la adición, eliminación, modificación y obtención de los listados de las áreas de guardia. En el transcurso de esta

iteración se realizarán las funcionalidades de exportar los listados de trabajadores, turnos, áreas y postas a documentos. Además se implementará las funcionalidades adicionar y modificar la planificación de la guardia.

Iteración 3

En esta iteración serán implementadas funcionalidades importantes como autenticar usuario, realizar permutas entre trabajadores, obtener listados de las guardias realizadas por un usuario dado, o de los ausentes a la guardia, eliminar y obtener en una lista las planificaciones de las guardias, exportar la planificación de la guardia así como exportar el listado de los ausentes a la guardia. Al concluir esta iteración se habrá cumplido con todos los requisitos identificados inicialmente.

2.5 Plan de duración de las iteraciones

Como parte del ciclo de vida de un proyecto utilizando XP se crea el plan de duración de cada una de las iteraciones. Este plan se encarga de mostrar las historias de usuario que serán abordadas en cada una de las iteraciones, así como la duración estimada de estas últimas y el orden en que se implementarán las historias de usuario.

Tabla 8. Plan de duración de las iteraciones

Iteración	Orden de las historias de usuario a implementar	Duración de la iteración
Iteración 1	Adicionar trabajador.	21 días
	Eliminar trabajador.	
	Obtener listado de los trabajadores existentes.	
	Modificar los datos de los trabajadores.	
	Adicionar posta de guardia.	
	Eliminar posta de guardia.	
	Obtener el listado de las postas existentes.	

	Modificar los datos de las postas.	
	Adicionar turno de guardia.	
	Eliminar turno de guardia.	
	Obtener el listado de los turnos de guardias existentes.	
	Modificar los datos de los turnos de guardias.	
Iteración 2	Adicionar área de guardia.	27 días
	Eliminar área de guardia.	
	Obtener el listado de las áreas de guardia existentes.	
	Modificar los datos de las áreas de guardia.	
	Exportar el listado de trabajadores.	
	Exportar el listado de turnos.	
	Exportar el listado de áreas.	
	Exportar el listado postas	
	Adicionar la planificación de la guardia.	
	Modificar la planificación de la guardia.	
Iteración 3	Eliminar la planificación de la guardia.	25 días
	Obtener la planificación de la guardia.	
	Exportar la planificación de la guardia.	
	Realizar permutas de guardia entre trabajadores.	

	Obtener el listado de las guardias realizadas por el usuario autenticado.	
	Obtener un listado de los ausentes a la guardia.	
	Exportar el listado de los ausentes a la guardia.	
	Autenticar usuario.	

2.6 Plan de Entregas

A continuación se presenta el plan de entregas ideado para la fase de implementación. Como resultado del mismo se obtendrán versiones del sistema al finalizar cada iteración en la fecha aproximada que se indica en la siguiente tabla.

Tabla 9. Plan de entrega

Aplicación	**Fin 1ra Iteración 3ra semana de Marzo 2015**	**Fin 2da Iteración 3ra semana de abril**	**Fin 3ra Iteración 2da semana de mayo**
Aplicación Web para la Gestión de la Guardia Obrera en la Facultad 5	v 0.1	v 0.2	1.0

2.7 Arquitectura del sistema

La arquitectura es necesaria para comprender el sistema, organizar el desarrollo, fomentar la reutilización y hacer evolucionar el sistema. Para definirla es necesario seleccionar y combinar patrones.

Un patrón es una solución a un problema en un contexto, codifica conocimiento específico acumulado por la experiencia en un dominio. Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente, y luego describe el núcleo de la solución a ese problema, de tal manera que puedes usar esa solución un millón de veces más, sin hacer jamás la misma cosa dos veces. (20)

Según la escala o nivel de abstracción presentan tres categorías (20):

- Patrones arquitecturales: aquellos que expresan un esquema organizativo estructural fundamental para sistemas software, no son otra cosa que los estilos.
- Patrones de diseño: aquellos que expresan esquemas para definir estructuras de diseño o sus relaciones con las que construir sistemas software.
- Idiomas: patrones de bajo nivel específicos para un lenguaje de programación o entorno concreto.

Para la definición de la arquitectura de la aplicación a desarrollar, se seleccionaron los patrones arquitecturales y los de diseño, los cuales son abordados a continuación:

Patrones arquitecturales

Los patrones arquitecturales definen las reglas generales de organización y las restricciones en la forma y la estructura de un grupo numeroso y variado de sistemas de software. Entre las clasificaciones establecidas por este patrón, se encuentra el Estilo de llamada y retorno, que a su vez define la arquitectura en capas. (21)

Arquitectura en capas

Para la realización del sistema se utilizó una arquitectura dividida en capas, específicamente en tres capas, debido a que nos permite separar los componentes de la aplicación de acuerdo al tipo de trabajo que realizan, desacoplándola en distintos niveles especializados en la presentación de información, la gestión de la lógica del negocio y el acceso a las fuentes de datos. Esto permite manejar posibles cambios y mejoras a la aplicación de forma más ordenada y afectar la menor cantidad de componentes posibles. En la siguiente figura se muestra una representación de las capas de la arquitectura con que contará la aplicación.

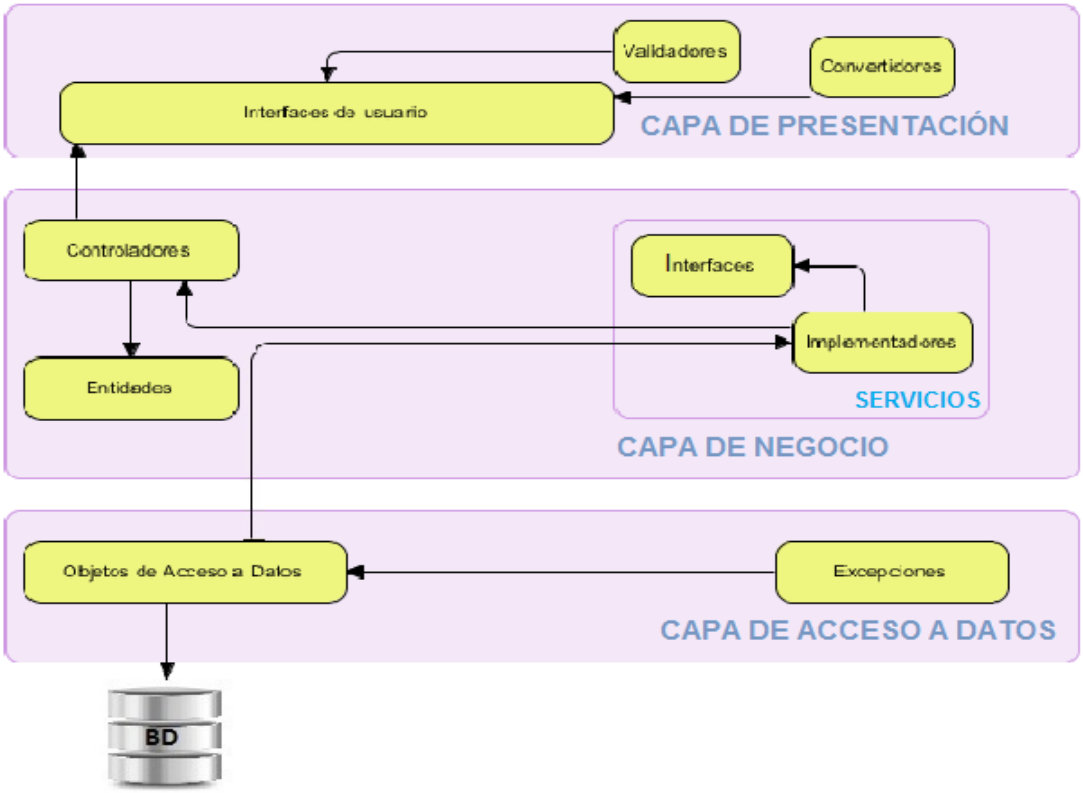


Figura 1. Arquitectura del sistema.

Capa de Presentación: Es la encargada de mostrar la información al usuario e interpretar las acciones realizadas por los mismos, realiza un filtrado previo para comprobar que no hay errores de formato. Se comunica únicamente con la capa de negocio.

Capa de Negocio: Es la encargada de la lógica del negocio, es la que recibe las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina además, capa de lógica del negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa es la intermediaria entre la capa de presentación y la capa de acceso a datos.

Capa de Acceso a Datos: Permite la persistencia de los objetos del negocio. Es la encargada de acceder a los datos.

2.7.2 Patrones de diseño

Un patrón de diseño es una solución repetible a un problema recurrente en el diseño de software. Esta solución no es un diseño terminado que puede traducirse directamente a código, sino más bien una descripción sobre cómo resolver el problema, la cual puede ser utilizada en diversas situaciones. Los patrones de diseño reflejan todo el rediseño y re modificación que los desarrolladores han ido haciendo a medida que intentaban conseguir mayor reutilización y flexibilidad en su software.

Los patrones documentan y explican problemas de diseño, y luego discuten una buena solución a dicho problema. Con el tiempo, los patrones comienzan a incorporarse al conocimiento y experiencia colectiva de la industria del software, lo que demuestra que el origen de los mismos radica en la práctica misma más que en la teoría. (22)

Patrones de Software para la Asignación General de Responsabilidades (GRASP)

Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. (23)

Patrones GRASP aplicados (23):

- **Creador**: el patrón creador ayuda a identificar quién debe ser el responsable de la creación o instanciación de nuevos objetos o clases. Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creador. Una ventaja es el bajo acoplamiento, lo cual supone facilidad de mantenimiento y reutilización.
- **Controlador**: es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto es para aumentar la reutilización de código y a la vez tener un mayor control.
- **Bajo acoplamiento**: es el patrón que mantiene las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.
- **Alta cohesión**: Asigna responsabilidad de modo que se mantenga alta cohesión. Una de las características principales de Java es la organización del trabajo en él mismo en cuanto a la

estructura del proyecto, lo cual permite crear y trabajar con clases con una alta cohesión. Por ejemplo, las clases con sufijo Controller en su nombre contienen varias funcionalidades estrechamente relacionadas entre ellas, teniendo un sentido común y un propósito único, siendo las encargadas de controlar las acciones de las plantillas y por lo tanto pertenecen a la capa del Controlador dentro de la arquitectura Modelo-Vista-Controlador.

2.7.3 Tarjeta CRC

Las tarjetas CRC (Clase, Responsabilidad, Colaborador) son una herramienta de reflexión en el diseño de software orientado a objetos. Fueron propuestas por Ward Cunningham y Kent Beck. Se utilizan normalmente cuando primero se determinan las clases que se necesitan y cómo van a interactuar y después se implementa la solución.

La técnica de las tarjetas CRC se utilizan para guiar al sistema a través de análisis basados en la responsabilidad. Las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema, y las clases con las que necesitan colaborar para completar sus responsabilidades. A continuación se muestran algunas de las tarjetas CRC que fueron confeccionadas, para consultar las restantes ver **Anexo #2**.

Tarjeta CRC	
Clase: AreaController	
Responsabilidades	**Colaboraciones**
adicionarArea eliminarArea actualizarArea	Area AreaServiceInterfaceImpl

Tarjeta CRC	
Clase: AreaServiceInterfaceImpl	
Responsabilidades	**Colaboraciones**
adicionarArea eliminarArea actualizarArea obtenerAreas	AreaDAO Area

Tarjeta CRC	
Clase: AreaDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**

Tarjeta CRC	
Clase: AreaServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarArea eliminarArea actualizarArea obtenerAreas	Area

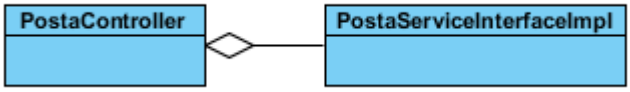
Tarjeta CRC	
Clase: Área	
Responsabilidades	**Colaboraciones**
Almacenar los datos correspondientes a un área determinada.	

Tarjeta CRC	
Clase: AreaConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Área a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes a las áreas.	AreaController

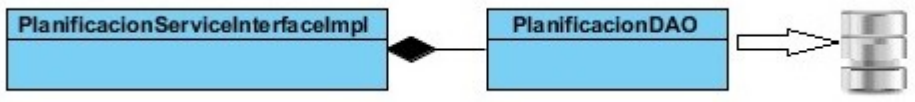
Representación de los patrones GRASP utilizados en la elaboración de las clases.

Patrón Creador: el uso de este patrón ayudó a identificar a la clase **`AreaController`** como la responsable de la creación e instanciación del objeto de la clase **`AreaServiceInterfaceImpl`**.

Patrón Controlador: el uso de este patrón se evidencia al asignarle la responsabilidad a la clase **`PostaControler`** de recibir los datos del usuario y enviarlos a las distintas clases según el método llamado.



Patrón Bajo Acoplamiento: el uso de este patrón se evidencia con la creación de las clases que se encargan del acceso y persistencia de los datos, esto desacopla a la clases principales de la complejidad de cómo acceder y persistir los datos. En este caso se evidencia con la creación de la clase **`PlanificacionDAO`**, que es la encargada de gestionar los datos referentes a las planificaciones.



2.8 Conclusiones parciales del capítulo

Con la elaboración del capítulo se transitaron las fases de Exploración y Planificación de la metodología de desarrollo utilizada para el desarrollo de la aplicación y se obtuvieron las conclusiones siguientes:

- El trabajo en conjunto, del cliente y el desarrollador posibilitaron la identificación de treinta requisitos funcionales, descritos en treinta historias de usuario.
- Para el diseño de la aplicación se utilizó una arquitectura en 3 capas y se aplicaron los patrones de diseño GRASP, posibilitando visualizar el sistema desde diferentes puntos de vistas y mediante la correcta aplicación de los patrones de diseño, se facilitó la reutilización de código y una mejora considerable del nivel de complejidad de las clases.
- Para la implementación de la aplicación, se definieron tres iteraciones, agrupando en cada una las historias de usuarios de acuerdo a su importancia. Dejando listo el diseño necesario para dar paso a la implementación y prueba del sistema a desarrollar.
- La correcta aplicación de la metodología seleccionada acorto el tiempo de desarrollo estimado.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA DE LA APLICACIÓN

3.1 Introducción

Durante el desarrollo de este capítulo se presentarán los resultados obtenidos durante la fase de prueba, se estarán viendo los resultados de las pruebas de aceptación que fueron hechas con el objetivo de aumentar la calidad de los sistemas reduciendo el número de errores no detectados y disminuyendo el tiempo transcurrido entre la aparición de un error y su detección. También se realizan las pruebas unitarias encargadas de verificar el código.

3.2 Producción

En esta fase se genera todo el código fuente necesario para satisfacer las historias de usuario definidas para la solución y se describen todas las tareas realizadas en cada iteración. Al inicio de cada historia de usuario, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Todo el trabajo de la iteración es expresado en tareas de programación, cada una de ellas es asignada a un programador como responsable.

Iteraciones en el desarrollo de la aplicación

En esta fase se genera todo el código fuente necesario para satisfacer las historias de usuario definidas para la solución y se describen todas las tareas realizadas en cada iteración.
Una vez realizada la planificación, se determinaron tres iteraciones de desarrollo sobre la aplicación, obteniendo como resultado un producto con todas las restricciones y características deseadas por el cliente para posteriormente ser utilizado. A continuación se detallan cada una de las iteraciones, mostrando las tareas más importantes para cada iteración.

Tabla 10. Historias de usuarios de la primera iteración

No	Historia de Usuario	Tiempo de implementación	
		Estimación	Real
1	Adicionar trabajador.	0,3	0,3
2	Eliminar trabajador.	0,2	0,2

3	Obtener listado de los trabajadores existentes.	0,2	0,2
4	Modificar los datos de los trabajadores.	0,3	0,3
5	Adicionar posta de guardia.	0,3	0,3
6	Eliminar posta de guardia.	0,3	0,3
7	Obtener el listado de las postas existentes.	0,2	0,2
8	Modificar los datos de las postas.	0,3	0,3
9	Adicionar turno de guardia.	0,3	0,3
10	Eliminar turno de guardia.	0,3	0,3
11	Obtener el listado de los turnos de guardias existentes.	0,2	0,2
12	Modificar los datos de los turnos de guardias.	0,3	0,3

Para consultar las tareas por historias de usuario restantes ver **Anexo #3**.

Tareas de las Historias de usuario. Primera iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 1
Nombre de la tarea: Adicionar un trabajador.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:27/2/2015	**Fecha de fin:**1/3/2015
Descripción: La aplicación tendrá la posibilidad de adicionar un trabajador especificando los atributos del mismo.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 2
Nombre de la tarea: Eliminar un trabajador.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:1/3/2015	**Fecha de fin:**3/3/2015

Descripción: El sistema tendrá la posibilidad de eliminar un trabajador que haya sido adicionado previamente.

Tabla 11. Historias de usuarios de la segunda iteración

No	Historia de Usuario	Tiempo de implementación	
		Estimación	Real
13	Adicionar área de guardia.	0,2	0,2
14	Eliminar área de guardia.	0,2	0,2
15	Obtener el listado de las áreas de guardia existentes.	0,2	0,2
16	Modificar los datos de las áreas de guardia.	0,3	0,3
17	Exportar el listado de trabajadores.	0,5	0,5
18	Exportar el listado de turnos.	0,3	0,3
19	Exportar el listado de áreas.	0,3	0,3
20	Exportar el listado postas	0,3	0,3
21	Adicionar la planificación de la guardia.	1	1
22	Modificar la planificación de la guardia.	0,4	0,4

Tareas de las Historias de usuario. Segunda iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 21
Nombre de la tarea: Adicionar la planificación de la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio:6/4/2015	**Fecha de fin:**13/4/2015
Descripción: La planificación tendrá la posibilidad de adicionar la planificación de la guardia obrera al sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 22
Nombre de la tarea: Modificar la planificación de la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,4
Fecha de inicio:13/4/2015	**Fecha de fin:**16/4/2015
Descripción: El sistema podrá realizar modificaciones sobre una planificación de guardia previamente realizada.	

Tabla 12. Historias de usuarios de la tercera iteración

No	Historia de Usuario	Tiempo de implementación	
		Estimación	Real
23	Eliminar la planificación de la guardia.	0,3	0,3
24	Obtener la planificación de la guardia.	0,3	0,3
25	Exportar la planificación de la guardia.	0,4	0,4
26	Realizar permutas de guardia entre trabajadores.	1	1
27	Obtener el listado de las guardias realizadas por el usuario autenticado.	0.6	0.6
28	Obtener un listado de los ausentes a la guardia.	0,3	0,3
29	Exportar el listado de los ausentes a la guardia.	0,2	0,2
30	Autenticar usuario.	0,4	0,4

Tareas de las Historias de usuario. Tercera iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 26
Nombre de la tarea: Realizar permutas de guardia entre trabajadores.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio:22/4/2015	**Fecha de fin:**29/4/2015

Descripción: El sistema tendrá la posibilidad de solicitar permutas a los trabajadores con planificaciones de guardia existentes.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 28
Nombre de la tarea: Obtener un listado de los ausentes a la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:3/5/2015	**Fecha de fin:**5/5/2015
Descripción: El sistema mostrará en una tabla un listado con los trabajadores ausentes a la guardia.	

3.3 Pruebas

Las pruebas de software constituyen un elemento indispensable para la garantía de la calidad del mismo. Las mismas permiten realizar una revisión final de las especificaciones con que debe cumplir el software y verificar el comportamiento del mismo ante determinadas situaciones.

La metodología XP divide las pruebas en dos grupos: pruebas unitarias, desarrolladas por los programadores, encargadas de verificar el código de forma automática, y las pruebas de aceptación, destinadas a evaluar si al final de una iteración se obtuvo el resultado esperado por el cliente.

Pruebas de aceptación

A continuación se muestran los casos de pruebas diseñados a un conjunto de historias de usuario que debido a que las mismas incluyen un conjunto de funcionalidades críticas para el desarrollo exitoso del sistema informático propuesto en la investigación.

Dichos casos de pruebas se describirán en tablas que contendrán los siguientes campos:

Código: Identificador de la prueba realizada, a su vez será sugerente al nombre de la prueba a la que hace referencia.

Historia de usuario: Nombre de la HU a la que hace referencia la prueba a realizar.

Nombre: Nombre de la prueba.

Descripción de la prueba: Breve descripción de la prueba.

Condiciones de Ejecución: Muestra las condiciones que deben cumplirse para poder llevar a cabo el caso de prueba, estas condiciones deben ser satisfechas antes de la ejecución del caso de prueba para que se puedan obtener los resultados esperados.

Entradas / Pasos de Ejecución: Descripción de cada uno de los pasos seguidos durante el desarrollo de la prueba, se tendrá en cuenta cada una de las entradas que hace el usuario con el objetivo de ver si se obtiene el resultado esperado.

Resultados de la prueba: Descripción breve del resultado que se espera obtener con la prueba realizada.

La evaluación de la prueba realizada se hará según el resultado de la misma, la tendrá uno de los tres resultados que a continuación se describen:

Satisfactoria: Cuando el resultado de la prueba es exactamente el esperado por el usuario.
Parcialmente bien: Cuando el resultado no es completamente el esperado por el cliente o usuario de la aplicación y muestra resultados erróneos o fuera de contexto.

Mal: Cuando el resultado de la prueba realizada genera un error de codificación en la aplicación o muestra como resultado elementos no deseados o fuera de contexto, trayendo como consecuencia que la funcionalidad requerida por el cliente no tenga resultado, lo que invalida también la HU.

El siguiente caso de prueba corresponde a la funcionalidad “Adicionar planificación” encargada de realizar una distribución de recursos para obtener así una planificación de la guardia:

Para consultar los casos de prueba de aceptación restantes ver **Anexo #4**

Caso de Prueba de Aceptación	
Código: **HU21_P1**	**Historia de Usuario**: Adicionar la planificación de la guardia.
Nombre: Adicionar planificación.	
Descripción: Prueba para la funcionalidad de adicionar planificación.	
Condiciones de Ejecución: El usuario con rol de planificador debe estar autenticado.	

Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Planificación” la opción “Adicionar”. El planificador introduce datos validos en los campos correspondientes a la planificación y presiona el botón “Adicionar”.
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La planificación ha sido adicionada correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU23_P1**	**Historia de Usuario**: Eliminar la planificación de la guardia
Nombre: Eliminar planificación.	
Descripción: Prueba para la funcionalidad de eliminar planificación.	
Condiciones de Ejecución: El usuario con rol de planificador debe estar autenticado.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Planificación” la opción “Listado”. Selecciona la planificación que desea eliminar haciendo click sobre ella y presiona el botón “Eliminar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La planificación ha sido eliminada correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU22_P1**	**Historia de Usuario**: Modificar la planificación de la guardia
Nombre: Modificar planificación.	
Descripción: Prueba para la funcionalidad de modificar planificación.	

Condiciones de Ejecución: El usuario con rol de planificador debe estar autenticado.
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Planificación” la opción “Listado”. Selecciona la planificación que desea modificar haciendo click sobre ella. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La planificación ha sido modificada correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU16_P1**	**Historia de Usuario**: Modificar datos de las áreas de guardia
Nombre: Modificar área.	
Descripción: Prueba para la funcionalidad de modificar área.	
Condiciones de Ejecución: El usuario con rol de planificador debe estar autenticado.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Area” la opción “Listado”. Selecciona el área que desea modificar haciendo click sobre ella. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El área ha sido modificada correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU17_P1**	**Historia de Usuario**: Exportar el listado de trabajadores
Nombre: Exportar listado de trabajadores.	
Descripción: Prueba para la funcionalidad de exportar el listado de los trabajadores adicionados previamente.	

Condiciones de Ejecución: El trabajador debe estar autenticado.
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Trabajador” la opción “Listado”. Presiona sobre el botón del formato en el que desea exportar el listado de trabajadores, Excel o Word.
Resultado Esperado: Se descarga el listado de trabajadores adicionados en el formato solicitado.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU18_P1**	**Historia de Usuario**: Exportar el listado de turnos.
Nombre: Exportar el listado de turnos.	
Descripción: Prueba para la funcionalidad de exportar el listado de los turnos adicionados previamente.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Turno” la opción “Listado”. Presiona sobre el botón del formato en el que desea exportar el listado de turnos, Excel o Word.	
Resultado Esperado: Se descarga el listado de turnos adicionados en el formato solicitado.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU19_P1**	**Historia de Usuario**: Exportar el listado de áreas.
Nombre: Exportar el listado de áreas.	
Descripción: Prueba para la funcionalidad de exportar el listado de las áreas adicionados previamente.	

Condiciones de Ejecución: El trabajador debe estar autenticado.
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Area” la opción “Listado”. Presiona sobre el botón del formato en el que desea exportar el listado de áreas, Excel o Word.
Resultado Esperado: Se descarga el listado de áreas adicionadas en el formato solicitado.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU20_P1**	**Historia de Usuario**: Exportar el listado postas.
Nombre: Exportar el listado postas.	
Descripción: Prueba para la funcionalidad de exportar el listado de las postas adicionados previamente.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Posta” la opción “Listado”. Presiona sobre el botón del formato en el que desea exportar el listado de postas, Excel o Word.	
Resultado Esperado: Se descarga el listado de postas adicionadas en el formato solicitado.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU25_P1**	**Historia de Usuario**: Exportar la planificación de la guardia.
Nombre: Exportar la planificación de la guardia.	
Descripción: Prueba para la funcionalidad de exportar el listado de las planificaciones adicionados previamente.	

Condiciones de Ejecución: El trabajador debe estar autenticado.
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Planificación” la opción “Listado”. Presiona sobre el botón del formato en el que desea exportar el listado de planificaciones, Excel o Word.
Resultado Esperado: Se descarga el listado de las planificaciones adicionadas en el formato solicitado.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU29_P1**	**Historia de Usuario**: Exportar el listado de los ausentes a la guardia.
Nombre: Exportar el listado de los ausentes a la guardia.	
Descripción: Prueba para la funcionalidad exportar el listado de los ausentes a la guardia.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona dentro del menú “Reportes” la opción “Ausentes a Guardias”. Presiona sobre el botón del formato en el que desea exportar el listado de los ausentes a la guardia, Excel o Word.	
Resultado Esperado: Se descarga el listado de las planificaciones adicionadas en el formato solicitado.	
Evaluación de la Prueba: Satisfactoria.	

Análisis de los resultados:

Para validar que la salida emitida por el sistema informático concordara con el resultado esperado por el cliente se diseñaron 3 iteraciones para los casos de prueba de aceptación en conjunto cliente-desarrolladores, en una primera iteración de 30 pruebas de aceptación realizadas se obtuvo como resultado que 20 salidas coincidieron con los resultados esperados representando un 66,7%. En una

segunda iteración se corrigieron las no conformidades detectadas en la iteración anterior y luego de realizadas nuevamente las 30 pruebas de aceptación se obtuvo como resultado que 25 pruebas resultaron satisfactorias representando el 83,3% lo que significó un logro para el equipo, mientras que una tercera iteración arrojó como resultado que las 30 pruebas de aceptación realizadas tuvieron un resultado satisfactorio, para un 100% de pruebas satisfactorias.

A continuación se muestra como quedan reflejados los resultados por cada una de las iteraciones de pruebas funcionales realizadas el sistema:

Iteración	Satisfactoria	Insatisfactoria	Total
1	20	10	30
2	25	5	30
3	30	0	30

3.4 Conclusiones parciales del capítulo

Después de realizar la implementación, prueba y validación del componente desarrollado, se puede arribar a las siguientes conclusiones:

- El plan de entrega fue cumplido acorde al tiempo planificado para el desarrollo de las tareas por cada historia de usuario.
- Con la realización de las pruebas de aceptación, se validó el cumplimiento de los requisitos y se comprobó la aceptación de la aplicación para gestionar la guardia obrera en la Facultad 5.

CONCLUSIONES GENERALES

Como resultado del presente trabajo, se concluye que:

- La elaboración del marco teórico de la investigación concerniente a los principales conceptos de la planificación de la guardia obrera y su gestión, permitió la selección adecuada de las herramientas, tecnologías y metodología utilizadas en el desarrollo de la presente investigación.
- La realización de las fases de planificación y producción permitieron definir las funcionalidades necesarias para llevar a cabo la propuesta de solución, y la obtención de una arquitectura y un diseño ajustado a las características de la aplicación web desarrollada, de manera que facilita su posible reutilización y mejora.
- La propuesta de solución desarrollada permite que se realicen permutas, de las planificaciones de guardia, entre los trabajadores.
- Con la utilización de la aplicación web desarrollada, se facilitan los procesos de planificación y control de la guardia obrera.
- Las pruebas de aceptación realizadas permitieron evaluar que con la propuesta de solución desarrollada, se obtuvo el resultado esperado por el cliente.

RECOMENDACIONES

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

- Implementar las funcionalidades necesarias para realizar la planificación de la guardia automáticamente.
- Desarrollar un módulo que permita la planificación y control de la guardia estudiantil.

BIBLIOGRAFÍA

1. **Pressman, R.S., *Software engineering : a practitioner's approach*. An introduction to UML. Vol. 7. 2001, New York: McGraw-Hill Companies.**
2. **Ken Schwaber, M.B., *Agile Software Development with Scrum*. 1 ed. 2001.**
3. **Sitio oficial de SCRUM: Control Chaos, Disponible en: https://www.scrum.org/resources**
4. **Oracle Technology Network. [Online] [Cited: mayo 4, 2015.] www.oracle.com/technetwork/java/index/html.**
5. **10. Python. [Online] [Cited: mayo 25, 2015]. https://www.python.org/.**
6. **Letelier P., Penadés C., Metodologías ágiles para el desarrollo de Software: eXtreme Programming (XP), Universidad Politécnica de Valencia, Disponible en: http://www.willydev.net/descargas/masyxp.pdf.**
7. **Almeida, Humberto y Torrez Aguilera, Grettell. Gestor Web para el control de la Guardia Obrera de la Universidad de las Ciencias Informáticas (UCI). 2009.**
8. **Rosabal Sanchez, Yisel. Sistema de gestión de información del Área de Extensión Universitaria.**
9. **Karel Bejerano González Y Félix Miguel Montes De Oca Barrios. Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7. S.l.: Universidad de las Ciencias Informáticas, 2009.**
10. **Ing. Carlos Aragonés López, Dr. L. Jorge Pérez Ávila. Sistema Automatizado para la Atención Médica Integral a Pacientes vih/sida "SIDATRAT". [Cited 12 diciembre 2014]. Available fromWorld Wide Web: <http://74.125.47.132/search?q=cache:ixi26YaElBYJ:www.informatica2007.sld.cu>.**
11. **TecnoHospital. [Online]. S.l.: s.n. [Cited 11 diciembre 2014]. Available from: http://www.coprava.com/prensa/Nota%20de%20prensa%20Gestion%20Turnos%20Abril08.pdf.**

12. **OPTIHPER. Asignación Optimizada de Personal y Tareas. In: [online]. [Cited 14 diciembre 2014]. Available from: http://users.dsic.upv.es/grupos/gps/optihper/aplicacion.html.**

13. **Acebal F.César, Cueva L. Juan, eXtreme Programming (XP): un nuevo método de desarrollo de software, Articulo de Novática. Disponible en: http://www.ati.es/novatica/2002/156/156-8.pdf.**
14. **Larman, Craig. Uml y Patrones: Introducción al análisis y diseño orientado a objetos. 2 ed. Prentice Hall, 2005.**
15. **Sommeville, Ian., Ingeniería de programas informáticos. 8 ed. ISBN-13:9780321313798. 2007.**

16. **Facultad de Ciencias Exactas (UNICEN). Facultad de Ciencias Exactas (UNICEN). sitio web de la Facultad de Ciencias Exactas (UNICEN). [En línea] 2003. [Citado el: 24 enero 2015.] http://www.exa.unicen.edu.ar/catedras/modemp/03_Introduccion_Integracion_Modelado.pdf.**

REFERENCIAS BIBLIOGRÁFICAS

1. Susana LLanusa Ruiz, Nereida Rojo Pérez, Magali Cabral Hernández. *TIC. In.* [Online] [Cited: enero 20, 2015.] http://bvs.sld.cu/revistas/spu/vol31_3_05/spu08.

2. Impacto TIC. In: [online]. [Accessed febrero 2, 2015]. Available from: http://www.utvm.edu.mx/OrganoInformativo/OrgFeb07/paginas/impacto.htm.

3. Almeida, Humberto y Torrez Aguilera, Grettell. Gestor Web para el control de la Guardia Obrera de la Universidad de las Ciencias Informáticas (UCI). 2009.

4. Rosabal Sanchez, Yisel. Sistema de gestión de información del Área de Extensión Universitaria.

5. Karel Bejerano González Y Félix Miguel Montes De Oca Barrios. Aplicación Web para el control de la Guardia Estudiantil y la Cuartelería de la Facultad 7. S.l.: Universidad de las Ciencias Informáticas, 2009.

6. Ing. Carlos Aragonés López, Dr. L. Jorge Pérez Ávila. Sistema Automatizado para la Atención Médica Integral a Pacientes vih/sida "SIDATRAT". [Cited 12 diciembre 2014]. Available from World Wide Web: <http://74.125.47.132/search?q=cache:ixi26YaElBYJ:www.informatica2007.sld.cu>.

7. TecnoHospital. [Online]. S.l.: s.n. [Cited 11 diciembre 2014]. Available from: http://www.coprava.com/prensa/Nota%20de%20prensa%20Gestion%20Turnos%20Abril08.pdf.

8. OPTIHPER. Asignación Optimizada de Personal y Tareas. In: [online]. [Accessed 14 diciembre 2014]. Available from: http://users.dsic.upv.es/grupos/gps/optihper/aplicacion.html.

9. Oracle Technology Network. [Online] [Cited: mayo 4, 2015.] www.oracle.com/technetwork/java/index/html.

10. Python. [Online] [Cited: mayo 25, 2015]. https://www.python.org/.

11. eFaber. [Online] EFABER SOLUCIONES INTELIGENTES. [Cited: mayo 21, 2015.] http://www.efaber.net:8881/formacion/fp/curso_php/index.html.

12. LIBROSWEB. [Online] [Cited: abril 12, 2015.] http://librosweb.es/libro/css/capitulo_1/breve_historia_de_css.html.

13. https://www.javascript.com/. [Online] [Cited: marzo 23, 2015.] https://www.javascript.com/.

14. Bilbomatica. [Online] [Cited: enero 30, 2015.] http://www.bilbomatica.es/es/content/vaadin.

15. HIVERNATE. [Online] [Cited: enero 30, 2015.] http://hibernate.org/tools/.

16. Frameworks Modelo Vista Controlador (MVC). [Online] [Cited: febrero 22, 2015.] http://www.nosolounix.com/2010/03/frameworks-modelo-vista-controlador-mvc.html.

17. Highsmith, J.A. *Adaptive Software Development: An Evolutionary Approach to Managing Complex Systems.* 2000.

18. scrum.org. [Online] [Cited: marzo 3, 2015.] www.scrum.org.

19. Larman, Craig. UML y Patrones - Una introducción al análisis y diseño orientado a objetos y el proceso unificado. S.l. Prentice Hall, 1999.

20. García Lira, Keidy, Granda Dihigo, Ailec y Tejera Hernández, Dayana Caridad. Arquitectura y Patrones de Diseño, 2009.

21. Ralph Johnson, John Vlissides, Richard Helm, Erich Gamma. Design Patterns. 1994.

22. Larman, Craig. UML y Patrones - Una introducción al análisis y diseño orientado a objetos y el proceso unificado. s.l. : Prentice Hall, 1999.

23. García Lira, Keidy, Granda Dihigo, Ailec y Tejera Hernández, Dayana Caridad. Arquitectura y Patrones de Diseño. [Conferencia]. La Habana, Cuba : s.n.

24. Pall, Gabriel A. Quality Process Management. México : Prentice Hall, 1987.

25. Juran, J. M. Jurán y la planificación para la calidad. s.l: Diaz de Santos, 1990.

26. Heras, Miguel A. Gestión. Barcelona: ESADE, 1996.

27. Concepto de gestión . [Online] [Cited: novimbre 24, 2014.] http://definicion.de/gestion/.

28. Facultad de Ciencias Exactas (UNICEN). Facultad de Ciencias Exactas (UNICEN). sitio web de la Facultad de Ciencias Exactas (UNICEN). [En línea] 2003. [Citado el: 24 de noviembre de 2014.] http://www.exa.unicen.edu.ar/catedras/modemp/03_Introduccion_Integracion_Modelado.pdf.

29. Concepto de planificación . [Online] [Cited: noviembre 24, 2014.] http://definicion.de/planificacion/#ixzz3UcXzY1Us.

ANEXOS

Anexo 1. Historias de usuario.

Tabla 11. Adicionar posta de guardia.

Historia de Usuario	
Número: 5	**Nombre:** Adicionar posta de guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador adicionar una posta de guardia, con los datos del nombre de la posta y el nombre del área a la cual pertenece.	
Observaciones: Hace referencia al RF5.	

Tabla 12. Eliminar posta de guardia.

Historia de Usuario	
Número: 6	**Nombre:** Eliminar posta de guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador eliminar una posta de guardia existente en el sistema.	
Observaciones: Hace referencia al RF6.	

Tabla 13. Obtener el listado de las postas existentes.

Historia de Usuario	
Número: 7	**Nombre:** Obtener el listado de las postas existentes.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,2	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario mostrar en una tabla un listado con todas las postas registradas en el sistema.	
Observaciones: Hace referencia al RF7.	

Tabla 14. Modificar los datos de las postas.

Historia de Usuario

Número: 8	**Nombre:** Modificar los datos de las postas.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Media
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador modificar los datos de una posta seleccionada, especificando el nuevo nombre de la posta y el nombre del área a la cual pertenece.	
Observaciones: Hace referencia al RF8.	

Tabla 15. Adicionar turno de guardia.

Historia de Usuario	
Número: 9	**Nombre:** Adicionar turno de guardia.
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Alta
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador crear un turno de guardia añadiendo la hora de comienzo, hora fin del turno y nombre del mismo.	
Observaciones: Hace referencia al RF9.	

Tabla 16. Eliminar turno de guardia.

Historia de Usuario	
Número: 10	**Nombre:** Eliminar turno de guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: El sistema deberá permitir al usuario con permisos de planificador eliminar un turno de guardia.	
Observaciones: Hace referencia al RF10.	

Tabla 17. Obtener el listado de los turnos de guardias existentes.

Historia de Usuario	
Número: 11	**Nombre:** Obtener el listado de los turnos de guardias existentes.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,2	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario mostrar en una tabla un listado con todos los turnos de guardia registrados en el sistema.	

Observaciones: Hace referencia al RF11.

Tabla 18. Modificar los datos de los turnos de guardias.

Historia de Usuario	
Número: 12	**Nombre:** Modificar los datos de los turnos de guardias.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,3	**Iteración Asignada:** 1
Descripción: La aplicación deberá permitir al usuario con permisos de planificador modificar los datos de un turno de guardia seleccionado, especificando la nueva hora de comienzo, hora fin del turno y nombre del mismo.	
Observaciones: Hace referencia al RF12.	

Tabla 19. Adicionar área de guardia.

Historia de Usuario	
Número: 13	**Nombre:** Adicionar área de guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,2	**Iteración Asignada:** 2
Descripción: La aplicación deberá permitir al usuario con permisos de planificador adicionar un área de guardia especificando su nombre.	

Tabla 20. Exportar el listado postas.

Historia de Usuario	
Número: 20	**Nombre:** Exportar el listado postas.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,3	**Iteración Asignada:** 2
Descripción: La aplicación deberá permitir al usuario exportar el listado de postas a un documento.	
Observaciones: Hace referencia al RF20.	

Tabla 21. Eliminar la planificación de la guardia.

Historia de Usuario	
Número: 23	**Nombre:** Eliminar la planificación de la guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja

Puntos Estimados: 0,3	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario con permisos de planificador eliminar una planificación de la guardia seleccionada.	
Observaciones: Hace referencia al RF23.	

Tabla 22. Obtener la planificación de la guardia.

Historia de Usuario	
Número: 24	**Nombre:** Obtener la planificación de la guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Media
Puntos Estimados: 0,3	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario mostrar en una tabla un listado con todas las planificaciones registradas en el sistema.	
Observaciones: Hace referencia al RF24.	

Tabla 23. Exportar la planificación de la guardia.

Historia de Usuario	
Número: 25	**Nombre:** Exportar la planificación de la guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,4	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario exportar la planificación de la guardia a un documento.	
Observaciones: Hace referencia al RF25.	

Tabla 24. Realizar permutas de guardia entre trabajadores.

Historia de Usuario	
Número: 26	**Nombre:** Realizar permutas de guardia entre trabajadores.
Prioridad en Negocio: Alta	**Complejidad en Desarrollo:** Alta
Puntos Estimados: 0,4	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario realizar intercambios de planificaciones de guardia entre dos trabajadores.	
Observaciones: Hace referencia al RF26.	

Tabla 25. Obtener el listado de las guardias realizadas por el usuario autenticado.

Historia de Usuario	
Número: 27	**Nombre:** Obtener el listado de las guardias realizadas por el usuario autenticado.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 1	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario mostrar un listado de las guardias que ha realizado.	
Observaciones: Hace referencia al RF27.	

Tabla 26. Obtener un listado de los ausentes a la guardia.

Historia de Usuario	
Número: 28	**Nombre:** Obtener un listado de los ausentes a la guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,2	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario con permisos de planificador mostrar un listado de los ausentes a una guardia seleccionada.	
Observaciones: Hace referencia al RF28.	

Tabla 27. Exportar el listado de los ausentes a la guardia.

Historia de Usuario	
Número: 29	**Nombre:** Exportar el listado de los ausentes a la guardia.
Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,2	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario con permisos de planificador exportar el listado de los ausentes a la guardia a un documento.	
Observaciones: Hace referencia al RF29.	

Tabla 28. Autenticar usuario.

Historia de Usuario	
Número: 30	**Nombre:** Autenticar usuario.

Prioridad en Negocio: Media	**Complejidad en Desarrollo:** Baja
Puntos Estimados: 0,4	**Iteración Asignada:** 3
Descripción: La aplicación deberá permitir al usuario autenticarse en el sistema insertando el nombre de usuario y contraseña.	
Observaciones: Hace referencia al RF30.	

Anexo 2. Tarjetas CRC.

Tarjeta CRC	
Clase: PostaController	
Responsabilidades	**Colaboraciones**
adicionarPosta eliminarPosta actualizarPosta	Posta PostaServiceInterfaceImpl

Tarjeta CRC	
Clase: PostaServiceInterfaceImpl	
Responsabilidades	**Colaboraciones**
adicionarPosta eliminarPosta actualizarPosta obtenerPostas	PostaDAO

Tarjeta CRC	
Clase: PostaDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**
Adicionar, eliminar, actualizar las postas. Obtener el listado de las postas.	

Tarjeta CRC	
Clase: PostaServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarPosta eliminarPosta actualizarPosta obtenerPostas	

Tarjeta CRC	
Clase: Posta	
Responsabilidades	**Colaboraciones**
Almacenar los datos correspondientes a una posta determinada.	Area

Tarjeta CRC	
Clase: PostaConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Posta a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes a las postas.	PostaController

Tarjeta CRC	
Clase: TurnoController	
Responsabilidades	**Colaboraciones**
adicionarTurno eliminarTurno actualizarTurno	Turno TurnoServiceInterfaceImpl

Tarjeta CRC
Clase: TurnoServiceInterfaceImpl

Responsabilidades	Colaboraciones
adicionarTurno eliminarTurno actualizarTurno obtenerTurnos	TurnoDAO

Tarjeta CRC	
Clase: TurnoDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**
Adicionar, eliminar, actualizar los turnos. Obtener el listado de los turnos.	

Tarjeta CRC	
Clase: TurnoServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarTurno eliminarTurno actualizarTurno obtenerTurnos	

Tarjeta CRC	
Clase: Turno	
Responsabilidades	**Colaboraciones**
Almacenar los datos correspondientes a un turno determinado.	

Tarjeta CRC	
Clase: TurnoConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Turno a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes	TurnoController

a los turnos.	

Tarjeta CRC	
Clase: PlanificacionController	
Responsabilidades	**Colaboraciones**
adicionarPlanificacion eliminarPlanificacion actualizarPlanificacion	Planificacion PlanificacionServiceInterfaceImpl

Tarjeta CRC	
Clase: PlanificacionServiceInterfaceImpl	
Responsabilidades	**Colaboraciones**
adicionarPlanificacion eliminarPlanificacion actualizarPlanificacion obtenerPlanificaciones	PlanificacionDAO

Tarjeta CRC	
Clase: PlanificacionDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**
Adicionar, eliminar, actualizar las planificaciones. Obtener el listado de las planificaciones.	

Tarjeta CRC	
Clase: PlanificacionServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarPlanificacion eliminarPlanificacion	

actualizarPlanificacion obtenerPlanificaciones	

Tarjeta CRC	
Clase: Planificacion	
Responsabilidades	**Colaboraciones**
Almacenar los datos correspondientes a una planificación determinada.	Turno Posta Trabajador

Tarjeta CRC	
Clase: PlanificacionConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Planificacion a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes a las planificaciones.	PlanificacionController

Tarjeta CRC	
Clase: TrabajadorController	
Responsabilidades	**Colaboraciones**
adicionarTrabajador eliminarTrabajador actualizarTrabajador	Trabajador TrabajadorServiceInterfaceImpl

Tarjeta CRC	
Clase: TrabajadorServiceInterfaceImpl	
Responsabilidades	**Colaboraciones**
adicionarTrabajador eliminarTrabajador actualizarTrabajador obtenerTrabajadores	TrabajadorDAO

Tarjeta CRC	
Clase: TrabajadorDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**
Adicionar, eliminar, actualizar los trabajadores. Obtener el listado de los trabajadores.	

Tarjeta CRC	
Clase: TrabajadorServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarTrabajador eliminarTrabajador actualizarTrabajador obtenerTrabajadores	

Tarjeta CRC	
Clase: Trabajador	
Responsabilidades	**Colaboraciones**
Almacenar los datos correspondientes a un trabajador determinado.	

Tarjeta CRC	
Clase: TrabajadorConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Trabajador a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes a los trabajadores.	TrabajadorController

Tarjeta CRC	
Clase: PermutaController	
Responsabilidades	**Colaboraciones**
adicionarPermuta	Permuta PermutaServiceInterfaceImpl

Tarjeta CRC	
Clase: PermutaServiceInterfaceImpl	
Responsabilidades	**Colaboraciones**
adicionarPermuta eliminarPermuta actualizarPermuta obtenerPermutas	PermutaDAO

Tarjeta CRC	
Clase: PermutaDAO **Súper clase:** GenericDao	
Responsabilidades	**Colaboraciones**
Adicionar, eliminar, actualizar las permutas. Obtener el listado de las permutas.	

Tarjeta CRC	
Clase: PermutaServiceInterface	
Responsabilidades	**Colaboraciones**
adicionarPermuta eliminarPermuta actualizarPermuta obtenerPermuta	

Tarjeta CRC	
Clase: Permuta	
Responsabilidades	**Colaboraciones**

Almacenar los datos correspondientes a una permuta determinada.	Planificacion

Tarjeta CRC	
Clase: PermutaConverter	
Responsabilidades	**Colaboraciones**
Convertir los objetos de tipo Permuta a cadenas y viceversa, para su utilización en las interfaces de usuarios correspondientes a las permutas.	PermutaController

Anexo 3. Tareas de las historias de usuario.

Primera iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 3
Nombre de la tarea: Obtener listado de los trabajadores existentes	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:3/3/2015	**Fecha de fin:**4/3/2015
Descripción: El sistema mostrará en una tabla un listado con todos los trabajadores existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 4
Nombre de la tarea: Modificar los datos de los trabajadores	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:4/3/2015	**Fecha de fin:**5/3/2015
Descripción: El sistema podrá realizar modificaciones sobre los trabajadores insertados previamente.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 5
Nombre de la tarea: Adicionar posta de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:5/3/2015	**Fecha de fin:**7/3/2015

Descripción: El sistema tendrá la posibilidad de adicionar una posta de guardia especificando los atributos del mismo.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 6
Nombre de la tarea: Eliminar posta de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:7/3/2015	**Fecha de fin:**8/3/2015
Descripción: El usuario tendrá la posibilidad de eliminar una posta de guardia que haya sido adicionada previamente.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 7
Nombre de la tarea: Obtener el listado de las postas existentes.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:8/3/2015	**Fecha de fin:**9/3/2015
Descripción: La aplicación mostrará en una tabla un listado con todas las postas existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 8
Nombre de la tarea: Modificar los datos de las postas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:9/3/2015	**Fecha de fin:**11/3/2015
Descripción: Esta tarea permitirá modificar los datos de las postas.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 9
Nombre de la tarea: Adicionar turno de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:11/3/2015	**Fecha de fin:**13/3/2015
Descripción: El sistema tendrá la posibilidad de adicionar un turno de guardia especificando los atributos del mismo.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 10
Nombre de la tarea: Eliminar turno de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3

Fecha de inicio:13/3/2015	**Fecha de fin:** 15/3/2015
Descripción: El sistema tendrá la posibilidad de eliminar un turno de guardia que haya sido adicionado previamente.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 11
Nombre de la tarea: Obtener el listado de los turnos de guardias existentes.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:15/3/2015	**Fecha de fin:**17/3/2015
Descripción: La aplicación mostrará en una tabla un listado con todos los turnos de guardia existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 12
Nombre de la tarea: Modificar los datos de los turnos de guardias.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:17/3/2015	**Fecha de fin:**19/3/2015
Descripción: El sistema podrá realizar modificaciones sobre los turnos de guardia previamente insertados.	

Segunda iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 13
Nombre de la tarea: Adicionar área de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:19/3/2015	**Fecha de fin:**21/3/2015
Descripción: El sistema tendrá la posibilidad de adicionar un área de guardia especificando los atributos de la misma.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 14
Nombre de la tarea: Eliminar área de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:21/3/2015	**Fecha de fin:**22/3/2015
Descripción: El sistema tendrá la posibilidad de eliminar un área de guardia que haya sido adicionada previamente.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 15
Nombre de la tarea: Obtener el listado de las áreas de guardia existentes.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:22/3/2015	**Fecha de fin:**24/3/2015
Descripción: La aplicación mostrará en una tabla un listado con todas las áreas de guardia existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 16
Nombre de la tarea: Modificar los datos de las áreas de guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:24/3/2015	**Fecha de fin:**27/3/2015
Descripción: La aplicación podrá realizar modificaciones sobre las áreas de guardia existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 17
Nombre de la tarea: Exportar el listado de trabajadores	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,5
Fecha de inicio:27/3/2015	**Fecha de fin:**31/3/2015
Descripción: La aplicación permitirá exportar el listado de trabajadores en formato Excel o Word.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 18
Nombre de la tarea: Exportar el listado de turnos.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:31/3/2015	**Fecha de fin:**2/4/2015
Descripción: La aplicación permitirá exportar el listado de turnos en formato Excel o Word.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 19
Nombre de la tarea: Exportar el listado de áreas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:2/4/2015	**Fecha de fin:**4/4/2015
Descripción: La aplicación permitirá exportar el listado de áreas en formato Excel o Word.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 20
Nombre de la tarea: Exportar el listado postas.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:4/4/2015	**Fecha de fin:**6/4/2015
Descripción: La aplicación permitirá exportar el listado de postas en formato Excel o Word.	

Tercera iteración.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 23
Nombre de la tarea: Eliminar la planificación de la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:16/4/2015	**Fecha de fin:**18/4/2015
Descripción: El sistema tendrá la posibilidad de eliminar una planificación de guardia que haya sido adicionada previamente.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 24
Nombre de la tarea: Obtener la planificación de la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,3
Fecha de inicio:18/4/2015	**Fecha de fin:**20/4/2015
Descripción: La aplicacion mostrará en una tabla un listado con todas las planificaciones existentes en el sistema.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 25
Nombre de la tarea: Exportar la planificación de la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,4
Fecha de inicio:20/4/2015	**Fecha de fin:**22/4/2015
Descripción: La aplicación permitirá exportar el listado de planificaciones de guardia en formato Excel o Word.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 27
Nombre de la tarea: Obtener el listado de las guardias realizadas por el usuario autenticado.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,6
Fecha de inicio:29/4/2015	**Fecha de fin:**3/5/2015

Descripción: La aplicación mostrará en una tabla un listado de las guardias realizadas por el usuario autenticado en el sistema.

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 29
Nombre de la tarea: Exportar el listado de los ausentes a la guardia.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,2
Fecha de inicio:5/5/2015	**Fecha de fin:**6/5/2015
Descripción: La aplicación permitirá exportar el listado de los ausentes a la guardia en formato Excel o Word.	

Tarea por historia de usuario	
Número de la tarea: 1	**Número de historia de usuario:** 30
Nombre de la tarea: Autenticar usuario.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0,4
Fecha de inicio:6/5/2015	**Fecha de fin:**10/5/2015
Descripción: La aplicación permitirá al usuario ingresar en el sistema.	

Anexo 4. Casos de prueba de aceptación.

Caso de Prueba de Aceptación	
Código: HU1_P1	**Historia de Usuario**: Adicionar trabajador.
Nombre: Adicionar trabajador.	
Descripción: Prueba para la funcionalidad de adicionar trabajador.	
Condiciones de Ejecución: El usuario debe estar autenticado con el rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Trabajador” la opción “Adicionar”. El planificador introduce datos validos en los campos correspondientes al trabajador y presiona el botón “Adicionar”.	

Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El trabajador ha sido adicionado correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: HU2_P1	**Historia de Usuario**: Eliminar trabajador.
Nombre: Eliminar trabajador.	
Descripción: Prueba para la funcionalidad de Eliminar trabajador.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Trabajador” la opción “Listado”. Selecciona el trabajador que desea eliminar haciendo click sobre ella y presiona el botón “Eliminar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El trabajador ha sido eliminado correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU3_P1	**Historia de Usuario**: Obtener listado de los trabajadores existentes.
Nombre: Obtener listado de los trabajadores.	
Descripción: Prueba para la funcionalidad de listar los trabajadores existentes.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El usuario selecciona en la vista trabajador la opción Listado.	
Resultado Esperado: El trabajador selecciona la opción “Listado” dentro del menú correspondiente a “Trabajador”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU4_P1	**Historia de Usuario**: Modificar los datos de los trabajadores.
Nombre: Modificar los datos de los trabajadores.	
Descripción: Prueba para la funcionalidad de Modificar los datos de los trabajadores.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Trabajador” la opción “Listado”. Selecciona el trabajador que desea modificar haciendo click sobre ella. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El trabajador ha sido modificado correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU5_P1	**Historia de Usuario**: Adicionar posta de guardia.
Nombre: Adicionar posta de guardia.	
Descripción: Prueba para la funcionalidad de Adicionar posta de guardia.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Posta” la opción “Adicionar”. El planificador introduce datos validos en los campos correspondientes a las postas y presiona el botón “Adicionar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La posta ha sido adicionada correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU6_P1	**Historia de Usuario**: Eliminar posta de guardia.
Nombre: Eliminar posta de guardia.	

Descripción: Prueba para la funcionalidad de Eliminar posta de guardia.
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.
Entrada / Pasos de Ejecución El planificador selecciona dentro del menú “Posta” la opción “Listado”. Selecciona la posta que desea eliminar haciendo click sobre ella y presiona el botón “Eliminar”.
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La posta ha sido eliminada correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: HU7_P1	**Historia de Usuario**: Obtener el listado de las postas existentes.
Nombre: Obtener el listado de las postas existentes.	
Descripción: Prueba para la funcionalidad de Obtener el listado de las postas existentes.	
Condiciones de Ejecución: El usuario debe estar autenticado. Deben existir postas adicionadas previamente.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Listado” dentro del menú correspondiente a “Posta”.	
Resultado Esperado: El sistema muestra el listado de todas las postas de guardia existentes.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU8_P1	**Historia de Usuario**: Modificar los datos de las postas.
Nombre: Modificar los datos de las postas.	
Descripción: Prueba para la funcionalidad de Modificar los datos de las postas.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	

Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Posta” la opción “Listado”. Selecciona la posta que desea modificar haciendo click sobre ella. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “La posta ha sido modificada correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: HU9_P1	**Historia de Usuario**: Adicionar turno de guardia.
Nombre: Adicionar turno de guardia.	
Descripción: Prueba para la funcionalidad de Adicionar turno de guardia.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Turno” la opción “Adicionar”. El planificador introduce datos validos en los campos correspondientes a la planificación y presiona el botón “Adicionar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El turno ha sido adicionado correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU10_P1	**Historia de Usuario**: Eliminar turno de guardia.
Nombre: Eliminar turno de guardia.	
Descripción: Prueba para la funcionalidad de Eliminar turno de guardia.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Turno” la opción “Listado”. Selecciona el turno que desea eliminar haciendo click sobre ella y presiona el botón “Eliminar”.	

Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El turno de guardia ha sido eliminada correctamente”.
Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU11_P1**	**Historia de Usuario**: Obtener el listado de los turnos de guardias existentes.
Nombre: Obtener el listado de los turnos de guardias existentes.	
Descripción: Prueba para la funcionalidad obtener el listado los turnos de guardias existentes.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Listado” dentro del menú correspondiente a “Turno”.	
Resultado Esperado: El sistema muestra el listado de todos los turnos de guardia existentes.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU12_P1**	**Historia de Usuario**: Modificar los datos de los turnos de guardias.
Nombre: Modificar los datos de los turnos de guardias.	
Descripción: Prueba para la funcionalidad modificar los datos de los turnos de guardias.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Turno” la opción “Listado”. Selecciona el turno que desea modificar haciendo click sobre el. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El turno ha sido modificado correctamente”.	

Evaluación de la Prueba: Satisfactoria.

Caso de Prueba de Aceptación	
Código: **HU13_P1**	**Historia de Usuario**: Adicionar área de guardia.
Nombre: Adicionar área de guardia	
Descripción: Prueba para la funcionalidad adicionar área de guardia	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Area” la opción “Adicionar”. El planificador introduce datos validos en los campos correspondientes al área y presiona el botón “Adicionar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El área de guardia ha sido adicionada correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU14_P1	**Historia de Usuario**: Eliminar área de guardia.
Nombre: Eliminar área de guardia.	
Descripción: Prueba para la funcionalidad eliminar área de guardia.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Area” la opción “Listado”. Selecciona el área que desea eliminar haciendo click sobre ella y presiona el botón “Eliminar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El área de guardia ha sido eliminada correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU15_P1**	**Historia de Usuario**: Obtener el listado de las áreas de guardia existentes.
Nombre Obtener el listado de las áreas de guardia existentes.	
Descripción: Prueba para la funcionalidad obtener el listado de las áreas de guardia existentes.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Listado” dentro del menú correspondiente a “Area”.	
Resultado Esperado: El sistema muestra el listado de todas las áreas de guardia existentes.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU16_P1**	**Historia de Usuario**: Modificar los datos de las áreas de guardia.
Nombre: Modificar los datos de las áreas de guardia.	
Descripción: Prueba para la funcionalidad modificar los datos de las áreas de guardia.	
Condiciones de Ejecución: El usuario debe estar autenticado con rol de planificador.	
Entrada / Pasos de Ejecución: El planificador selecciona dentro del menú “Area” la opción “Listado”. Selecciona el área que desea modificar haciendo click sobre ella. Introduce o mantiene datos válidos para todos los campos y presiona el botón “Modificar”.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El área ha sido modificado correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código:	**Historia de Usuario**: Realizar permutas de guardia entre trabajadores.

HU16_P1	
Nombre: Realizar permutas de guardia entre trabajadores.	
Descripción: Prueba para la funcionalidad realizar permutas de guardia entre trabajadores.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El usuario selecciona en la vista permuta la opción “Enviar solicitud de permuta”, posteriormente selecciona la persona con quien desea permutar y le envía la solicitud.	
Resultado Esperado: El sistema muestra el siguiente mensaje de notificación: “El trabajador ha sido modificado correctamente”.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: HU24_P1	**Historia de Usuario**: Obtener el listado de las planificaciones de las guardias.
Nombre: Listar planificaciones.	
Descripción: Prueba para la funcionalidad listar las planificaciones.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Listado” dentro del menú correspondiente a “Planificación”.	
Resultado Esperado: El sistema muestra una tabla con las planificaciones existentes.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU24_P1**	**Historia de Usuario**: Autenticar usuario.
Nombre: Listar planificaciones.	
Descripción: Prueba para la funcionalidad autenticar usuario.	
Condiciones de Ejecución:	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Mi cuenta” dentro del menú correspondiente a “Usuario”. Ingresa su usuario y contraseña.	
Resultado Esperado: El sistema le permite acceder a las funcionalidades admitidas por su rol.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU24_P1**	**Historia de Usuario**: Obtener el listado de los ausentes a la guardia.
Nombre: Listar planificaciones.	
Descripción: Prueba para la funcionalidad obtener el listado de los ausentes a la guardia..	
Condiciones de Ejecución: El usuario debe estar autenticado con el rol de planificador.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Ausentes a Guardias” dentro del menú correspondiente a “Reportes”. Selecciona la fecha de la guardia que se quiera conocer el listado de los ausentes.	
Resultado Esperado: El sistema le permite obtener un listado con.	
Evaluación de la Prueba: Satisfactoria.	

Caso de Prueba de Aceptación	
Código: **HU24_P1**	**Historia de Usuario**: Obtener el listado de las guardias realizadas por el usuario autenticado.
Nombre: Listar planificaciones.	
Descripción: Prueba para la funcionalidad obtener el listado de las guardias realizadas por el usuario autenticado.	
Condiciones de Ejecución: El trabajador debe estar autenticado.	
Entrada / Pasos de Ejecución: El trabajador selecciona la opción “Guardias Realizadas” dentro del menú correspondiente a “Reportes”.	
Resultado Esperado: El sistema le permite obtener un listado con.	
Evaluación de la Prueba: Satisfactoria.	

GLOSARIO DE TÉRMINOS

Procesos

Varios autores han definido lo que es un proceso, por lo que esto ha contribuido a una evolución del concepto.
Gabriel Pall, define un proceso como “la organización lógica de personas, materiales, energía, equipamiento e información en actividades de trabajo diseñadas para producir un resultado final requerido (24).
Juran lo describe como “Una serie de acciones sistemáticas dirigidas al logro de un objetivo previamente definido” (25). También veremos que Miguel Heras se refiere al proceso como “el conjunto de actividades secuenciales que realizan una transformación de una serie de inputs (material, mano de obra, capital, información) en los outputs deseados (bienes y/o servicios) añadiendo valor” (26).

Gestión

Indica, que se trata de realización de diligencias enfocadas a la obtención de algún beneficio, tomando a las personas que trabajan en la compañía como recursos activos para el logro de los objetivos. Por otro lado también se debe tener en cuenta que la gestión en este caso requiere definir por su parte todas las políticas utilizadas por el personal para que así el mismo pueda ponerse en funcionamiento articulando las funciones sociales teniendo en cuenta las metas que posee la empresa.(27)

Importante subrayar que la gestión tiene como objetivo primordial conseguir aumentar los resultados óptimos de una industria o compañía y depende fundamentalmente de cuatro pilares básicos gracias a los cuales puede conseguir que se cumplan las metas marcadas. Dichos pilares anteriormente mencionados son: (27)

- **Estrategia:** es el conjunto de líneas y de trazados de los pasos que se deben llevar a cabo, teniendo en cuenta factores como el mercado o el consumidor, para consolidar las acciones y hacerlas efectivas.

- **Cultura:** es el grupo de acciones para promover los valores de la empresa en cuestión, para fortalecer la misma, para recompensar los logros alcanzados y para poder realizar las decisiones adecuadas.
- **Estructura:** son las actuaciones para promover la cooperación, para diseñar las formas para compartir el conocimiento y para situar al frente de las iniciativas a las personas mejores cualificadas.
- **Ejecución:** consiste en tomar las decisiones adecuadas y oportunas, fomentar la mejora de la productividad y satisfacer las necesidades de los consumidores.

Gestión de procesos

La gestión por sí misma, no es más que la coordinación organizada de los recursos disponibles con el fin de conseguir un objetivo. Teniendo esto en cuenta podemos decir que la gestión de procesos es la forma de gestionar organizadamente las relaciones de los procesos entre ellos y con las demás áreas y el entorno.

Un Sistema de gestión, por tanto, ayuda a una organización a establecer las metodologías, las responsabilidades, los recursos, las actividades que le permitan una gestión orientada hacia la obtención de esos “buenos resultados” que desea, o lo que es lo mismo, la obtención de los objetivos establecidos.

Dentro de la gestión de procesos también podemos encontrar los **indicadores**, los cuales permiten determinar en qué medida de ejecución se encuentra el plan; estos constituyen una herramienta para controlar los resultados por lo que es importante su correcta definición y determinar los que ejercen mayor influencia en los resultados de la actividad a analizar. Estos por definición deben de ser medibles (28)

Planificación

Los esfuerzos que se realizan a fin de cumplir objetivos y hacer realidad diversos propósitos se enmarcan dentro de una planificación. Este proceso exige respetar una serie de pasos que se fijan en un primer momento, para lo cual aquellos que elaboran una planificación emplean diferentes herramientas y expresiones.

La planificación supone trabajar en una misma línea desde el comienzo de un proyecto, ya que se requieren múltiples acciones cuando se organiza cada uno de los proyectos. Su primer paso, dicen los expertos, es trazar el plan que luego será concretado.

En otras palabras, la planificación es un método que permite ejecutar planes de forma directa, los cuales serán realizados y supervisados en función del planeamiento (29)**.**

Framework

Se emplea unos muchos ámbitos del desarrollo de sistemas software, no solo en el ámbito de aplicaciones Web sino también en aplicaciones médicas, de visión por computadoras, para el desarrollo de juegos, entre muchos otros.

El término framework, se refiere a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que se le puede añadir piezas para construir una aplicación concreta.