Universidad de las Ciencias Informáticas

Facultad 6

Sistema para la gestión de reportes de la Dirección de Mantenimiento de la Universidad de Ciencias Informáticas v2.0

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

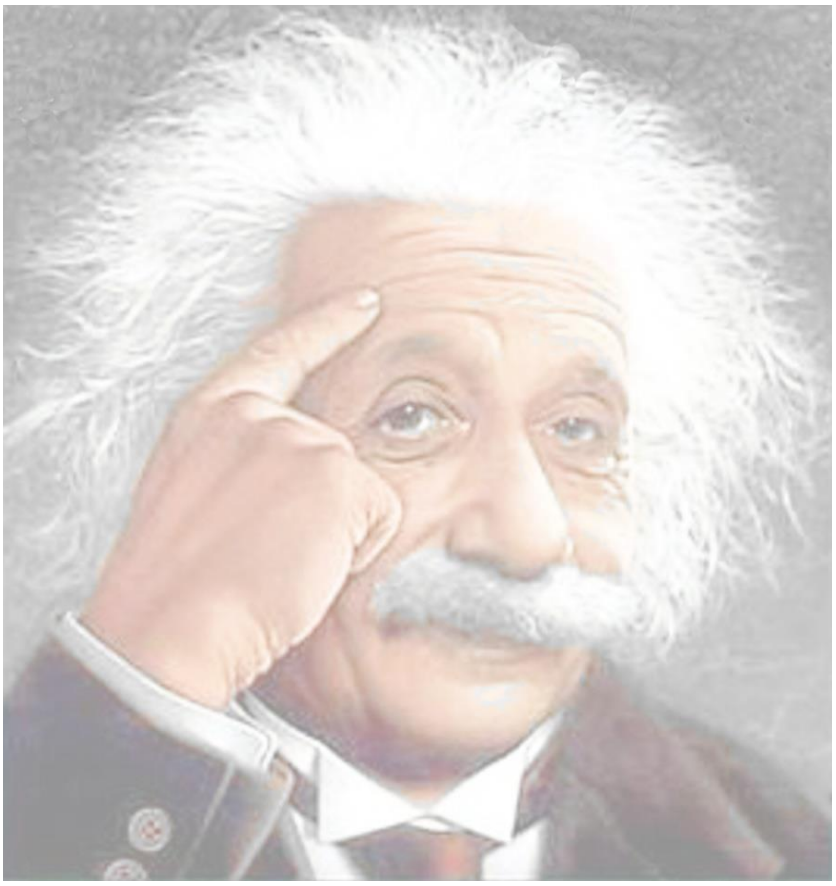
Autor: Ing.Yudelsi Rafael Fonseca Vargas

Tutor: MSc. Omar Mar Cornelio

La Habana, 2015

"Año 57 de la Revolución"

FRASE



"*Si buscas resultados distintos, no hagas siempre lo mismo.*"

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio.

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

Ing.Yudelsi Rafael Fonseca Vargas

_____

Firma del Autor

MSc. Omar Mar Cornelio

_____

Firma del Tutor

Datos de Contacto

Autor:

- ✓ Yudelsi Rafael Fonseca Vargas
- ✓ Universidad de las Ciencias Informáticas.
- ✓ e-mail: yrafael@uci.cu

Tutor:

- ✓ Omar Mar Cornelio
- ✓ Universidad de las Ciencias Informáticas.
- ✓ e-mail: omarmar@uci.cu

Estudios realizados: Licenciado en Educación en la Especialidad de Informática, Máster en Informática Aplicada. Ha recibido Curso de postgrado de Planificación de proyectos productivos, Auditores internos para proyectos, Arquitecturas de Redes y Comunicaciones, Sistemas inteligentes, Arquitectura de Bases de datos, Ingeniería de Software y Gestión de Software, Ciencia tecnología y Sociedad, Gestión de la Información, Gestión del Conocimiento, Gestión de Proyecto, Metodología de la Investigación, CPDs en cobre y fibra, Estadística Aplicada, Diseños de experimentos, Publicación de revistas con OJS, Arquitectura de máquinas computadoras.

Función: Vicedecano Administrativo, vinculado a la docencia, en el curso regular diurno en la disciplina de Sistemas Digitales. Certificado como especialista en calidad para revisiones de Software. Certificado como instalador en Centros de Datos por empresa española EQUINSA.

Experiencia como tribunal de tesis, tutor y oponte en pregrado. Actualmente realiza doctorado en Automática con la Universidad Central de Las Villas.

Índice

Índice

Índice

Índice de Figuras

Índice de Tablas

Índice

Resumen

La Universidad de las ciencias Informáticas, presenta varias áreas que permiten en conjunto establecer tener un centro de altos estudios así como varios centros de desarrollo de software, que interactúan entre sí para lograr formar profesionales de la informática y contribuir a la automatización de la sociedad cubana. Para su buen funcionamiento, la Universidad cuenta con la Dirección de Mantenimiento, la cual tiene como misión fundamental, mantener la disponibilidad de las capacidades constructivas y tecnológicas a través de un sistema preventivo-correctivo. Esta dirección realiza sus funciones a partir de reportes emitidos por áreas de la universidad, siguiendo normas técnicas y ejecutando tareas de las cuales se emiten reportes de estado que van dirigido a los directivos de la entidad. Actualmente, la dirección cuenta con un Sistema de Gestión de Reportes, que, aunque satisface parte de las necesidades del proceso de mantenimiento, no permite tener en cuenta la disponibilidad de recursos, las asignaciones a brigadas y no brinda información útil para la toma de decisiones. En el presente trabajo se argumenta el desarrollo de la segunda versión del Sistema de Gestión de Reportes de la Dirección de Mantenimiento, describiendo sus requisitos, diseño y validación según la metodología OpenUP.

Palabras claves: gestión, mantenimiento, reporte, sistema, universidad.

Abstract: The University of Information Science, has several areas that allow a whole set have a higher education center and various software development centers, which interact to achieve train computer professionals and contribute to the automation of Cuban society. To function properly, the University has with the Maintenance, which primary mission, maintain availability of constructive and technological capabilities through a preventive and corrective system, achieving a high level of satisfaction in solving problems. This e performs its functions from reports issued by University areas, following technical standards and executing tasks which status reports that are addressed to the directors of the company are issued. Currently, the management has a Management System Reports, which, although part meets the needs of the maintenance process, fails to take into account the availability of resources, assignments brigades and does not provide useful information for decision-making. In this paper the development of the second version of System Management Reports of the Directorate of Maintenance is argued, describing their requirements, design and validation according to OpenUP methodology.

Introducción

Las tecnologías de la información facilitan el desarrollo de las actividades diarias de la sociedad, expanden las vías de comunicación entre los seres humanos y brindan formas de almacenamiento de datos. El avance de las Tecnologías de la Información y las Comunicaciones (TIC) contribuyó con la evolución del desarrollo científico técnico del ser humano, el cual posteriormente inventó las computadoras y con ellas surge la necesidad de informatizar procesos. Su uso se vuelve casi imprescindible para estar acorde al acelerado desarrollo en todos los sectores de la sociedad, haciendo especial énfasis en el sector empresarial.

El avance creciente en las tecnologías informáticas posibilita hoy, que las empresas sean cada vez más eficientes; convirtiendo las tareas engorrosas y repetitivas que normalmente consumen mucho tiempo y recursos en actividades que fluyen con rapidez y precisión, manejando mayor volumen de información. Estas ventajas elevan no sólo los niveles de productividad y competitividad de las empresas, sino también logran un amplio alcance en el control de sus recursos y que los niveles de información sean más precisos, útiles y actualizados.

Hoy día las instituciones requieren ser informatizadas debido a la cantidad de información que ellas procesan para su desempeño. Generalmente, estas instituciones necesitan un mecanismo que les permita proporcionar información útil de manera rápida. Para ello existen herramientas que permiten a los usuarios obtener con facilidad datos de archivos o bases de datos en forma de reportes. Estos son objetos que entregan información en un formato particular y que permiten realizar ciertas operaciones como: imprimirlos, enviarlos por correo electrónico y guardarlos a un archivo, a partir de los elementos almacenados en una base de datos.

La generación de reportes es una tarea necesaria en los sistemas que gestionan información. Generalmente los sistemas de gestión cuentan con un conjunto limitado de reportes y se ven limitados desde el punto de vista de la inserción de nuevos reportes asociados a cambios en los procesos informatizados.

En 2002 se crea la Universidad de las Ciencias Informáticas (UCI), centro que combina en sus campos universitarios la producción de software y la docencia, con un conjunto de centros y áreas que permiten el trabajo combinado en la Universidad.

Introducción

La UCI cuenta con varias vicerrectorías que contribuyen a la ejecución de prestación de servicios. Para su funcionamiento se estructura en diferentes áreas entre las que se encuentra la Dirección de Mantenimiento, la cual tiene como misión fundamental, mantener la disponibilidad de las capacidades constructivas y tecnológicas a través de un sistema preventivo-correctivo, logrando un alto nivel de satisfacción en la solución de los problemas. Para garantizar el cumplimiento de su misión tiene definidos como objetivos principales: la aplicación del mantenimiento preventivo planificado al total de las edificaciones e instalaciones de la Universidad; la aplicación del sistema de mantenimiento constructivo ligero, dándole solución a los reportes por averías imprevistas que se presenten en todas las áreas; las reparaciones constructivas, el aseguramiento de los servicios de climatización, electrodomésticos y otros.

En la actualidad la gestión y control de los procesos que se llevan a cabo en la dirección anteriormente mencionada, se realiza a través de la vía telefónica; medio por el cual se reciben todos los datos asociados a las roturas, problemas o servicios solicitados por los estudiantes o trabajadores de la institución anteriormente mencionada. Actualmente el Sistema de Reportes v1.0 cuenta con una serie de reportes que no funcionan por lo que les dificulta tener un control de los recursos. Esta dirección tiene un local donde existe una persona encargada de recibir y compilar toda la información, a partir de la cual se generan reportes que son registrados de forma manual en documentos Excel, lo cual atenta contra su perdurabilidad, integridad, normalización y seguridad. Actualmente el sistema de gestión de reporte no realiza las siguientes funcionalidades:

- No tiene relación con el Sistema de Control de Activos (ASSET), provocando la falta de constancia de las cantidades disponibles de recursos para la ejecución de tareas de mantenimiento.
- Presenta dificultades de seguridad al permitir que cualquier usuario emita reportes.
- La información que brinda no es suficiente para la correcta toma de decisiones afectando la eficacia y eficiencia del proceso de reportes.
- No permite diferenciar las brigadas tercerizadas de las brigadas de la dirección de mantenimiento durante la asignación de reportes.
- No incluye la gestión de prioridades en relación a la disponibilidad de recursos y los reportes emitidos por instancias superiores.
- No realiza la trazabilidad de los reportes en relación a la capacidad de respuesta.
- No permite clasificar los reportes en cuanto a sus prioridades.
- No realiza la asignación directa de reportes a estados según la disponibilidad de recursos.

Todo esto provoca:

- Incapacidad de análisis de la gestión laboral por cada brigada de resolución de problemas de mantenimiento.
- Dificultad en la realización de búsquedas específicas que posibiliten filtrar la información, imposibilitando muchas veces la realización de estimaciones acerca de los recursos que más o menos se utilizan.
- Desconocimiento del usuario de la fase en que se encuentra el proceso del reporte realizado
- Imposibilidad para llevar el control estadístico del consumo de recursos en todas las áreas.
- Inaccesibilidad y pocos detalles acerca del estado actual de los problemas reportados y en el caso que se pueda acceder debe imprimirse en papel y ser entregados personalmente, incrementando el consumo de materiales y disminuyendo la capacidad productiva de los trabajadores.

A partir de la situación problemática anteriormente expuesta, se plantea como **Problema a resolver:** ¿Cómo contribuir a la toma de decisiones para la gestión de reportes en la Dirección de Mantenimiento de la UCI en su Versión 2.0?

Para enmarcar los límites del presente trabajo se define como **Objeto de estudio:** sistemas para la gestión de información.

Definiéndose como **Campo de acción:** los sistemas de información para la gestión de reportes de mantenimiento.

Para dar solución al problema planteado se define como **Objetivo general**: Desarrollar una nueva versión del sistema de gestión de reportes que contribuya a la toma de decisiones en la Dirección de Mantenimiento de la UCI.

Idea a defender:

Con el desarrollo de una aplicación informática, se contribuirá a la toma de decisiones para la gestión de reportes en la Dirección de Mantenimiento de la Universidad de las Ciencias Informáticas en su Versión 2.0.

Para dar solución al objetivo planteado se definen las siguientes **tareas de investigación:**

- Conformación del marco teórico a partir del estudio del proceso de gestión de reportes en la Dirección de Mantenimiento de la Universidad.

- Selección de las herramientas, tecnologías, lenguajes y metodologías a utilizar, para el desarrollo del sistema.
- Análisis y diseño del Sistema Web para la Gestión de Reportes en la Dirección de Mantenimiento de la UCI.
- Implementación de los requisitos funcionales definidos para el Sistema Web para la Gestión de Reportes en la Dirección de Mantenimiento de la UCI.
- Validación de la aplicación propuesta**.**

Con el objetivo de dar un correcto cumplimiento a las tareas propuestas, para una mejor comprensión de los procesos y el desarrollo de un estudio profundo de técnicas, herramientas y metodologías de desarrollo se utilizaron como **Métodos Científicos de Investigación:**

Métodos Teóricos:

- **Método Histórico - Lógico**: Permitirá el análisis del comportamiento que ha tenido la gestión de reportes en la Dirección de Mantenimiento identificándose las mejoras necesarias en el proceso actual, además del análisis del proceso de gestión de reportes y su evolución dentro de las herramientas que se comercializan.

Métodos Empíricos:

- **Método Entrevista:** Se utilizará para obtener directamente impresiones y constatar las deficiencias que posee la gestión de reportes en la Dirección de Mantenimiento de la UCI.
- **Método Observación:** Mediante guía de observación para constatar las deficiencias que posee la gestión de reportes en la Dirección de Mantenimiento de la UCI.

Posibles resultados: Sistema para la gestión de reportes que permitirá a los trabajadores de la Dirección de Mantenimiento estructurar y ejecutar los procesos que se realizan durante el trabajo con los reportes que se realicen por el personal de la UCI, además de proveer a los usuarios de un sistema de registro y supervisión de sus solicitudes de mantenimiento. El sistema facilitará el manejo de toda la información referente a los reportes y estimará de manera acertada el estado real del trabajo que se realiza en esta dirección.

Introducción

El presente documento se estructura de la siguiente manera:

Capítulo 1: Fundamentos teóricos: Se realizará un estudio del estado del arte abordando como se realiza la gestión de reportes como proceso, definiéndose conceptos claves para la investigación y proponiendo el basamento teórico bajo el cual se desarrollará la misma. Además de fundamentar la elección de la metodología de desarrollo, las herramientas y tecnologías a utilizar para el desarrollo del software.

Capítulo 2: Análisis y diseño del sistema: Se analizan las características del negocio, explicando la propuesta de solución al problema de la investigación, se definen los principales requisitos a partir de la metodología seleccionada. Se describe la solución a partir de su diseño, se argumenta su arquitectura y principales componentes, patrones de diseño, además de dar una vista del entorno de despliegue del sistema.

Capítulo 3: Implementación y pruebas del sistema: Se describe todo el proceso de implementación y se detallan las pruebas realizadas al sistema para la comprobación del cumplimiento del objetivo de la investigación.

Capítulo 1. Fundamentación Teórica

Introducción

Para comprender los orígenes de la investigación, es necesario describir los aspectos teóricos relacionados con la gestión de reportes de mantenimiento y las bases técnicas a tener en cuenta para el desarrollo del sistema. En el presente capítulo se exponen los principales conceptos dentro del campo de acción y se estudian los sistemas utilizados con fines similares. Se caracteriza la metodología de desarrollo escogida y se argumentan las soluciones técnicas concebidas para la construcción del software.

1.1 Marco Conceptual

1.1.1 Gestión de la información

Las organizaciones son sistemas inteligentes, éstas deben ser generadoras, almacenadoras y transformadoras de los conocimientos que le permitan enfrentar y modificar su entorno, ya sea para adaptarse a él o, de ser posible, para adaptar el entorno a su beneficio. Es común afirmar que en ambientes tan complejos como los que deben enfrentar hoy en día las organizaciones, sólo aquellas que utilicen todos los medios a su alcance podrán lograr el objetivo de ser exitosas. Las tecnologías de información juegan un papel central en esta enloquecida carrera emprendida por las empresas contemporáneas(Fernando, 2010).

Actualmente, los sistemas de información constituyen una necesidad clave de las organizaciones, ya que este tipo de tecnologías se ven involucradas desde las líneas de producción hasta los más altos puestos encargados de las toma de decisiones. De esta forma, los sistemas de procesamiento de información inciden en la manera de trabajar, en la cultura y en la estrategia de muchas de las compañías actuales.

El facilitar la comunicación, el procesamiento de datos y la manera de almacenar la información permiten establecer redes, y mediante los procedimientos adecuados, también garantizan descubrir el conocimiento diseminado a lo largo de la organización para convertirlo en material de aprendizaje útil a todos los miembros de la misma. En el marco de competencia actual y con los retos que implican una economía global, es necesario que las empresas vean en los sistemas de información una herramienta que les permitirá adquirir una ventaja competitiva sostenible, haciendo eficientes sus procesos y optimizando sus operaciones.

Capítulo 1. Fundamentación Teórica

La gestión de la información es todo lo relacionado con la obtención de la información adecuada, en la forma correcta, para la persona indicada, al costo adecuado, en el tiempo oportuno, en el lugar apropiado, para tomar la acción correcta(Fernando, 2010). Esta definición abarca los objetivos de la gestión de información, que son:

- Maximizar el valor y los beneficios derivados del uso de la información.
- Minimizar el costo de adquisición, procesamiento y uso de la información.
- Determinar responsabilidades para el uso efectivo, eficiente y económico de la información.
- Asegurar un suministro continuo de la información.

1.1.2 Sistemas de Información

Los Sistemas de Información representan un conjunto de elementos orientados al tratamiento y administración de datos e información, organizados y listos para su posterior uso, generados para cubrir una necesidad (objetivo). Dichos elementos formarán parte de alguna de estas categorías(Blanco Espinoza, 2011):

- Personas
- Datos
- Actividades o técnicas de trabajo.
- Recursos materiales (recursos informáticos)

Todos estos elementos interactúan entre sí para procesar un conjunto de datos que se convierte en información más sustancial aportando conocimientos que influyen en la toma de decisiones.

Los elementos que conforman un Sistema de Información son cuatro:

1. **Información:** Todo aquello (número, texto, imagen, voz) que el sistema captura, procesa, almacena y distribuye.
2. **Personas:** Usuarios que introduce, procesa y/o utiliza la información del sistema.
3. **Tecnologías de la Información y las Comunicaciones:** Hardware y software empleado en las tareas del sistema.
4. **Técnicas de trabajo:** Métodos utilizados por las personas y las tecnologías para desempeñar su trabajo.

Los elementos anteriores operan coordinadamente para alcanzar un objetivo determinado en la empresa, ya sea gestionar las transacciones corrientes, facilitar la toma de decisiones estratégicas, mantener un canal comercial o cualquier otro propósito útil para el éxito de la empresa. Es de vital importancia resaltar que dentro de un Sistema de Información la capa de las TICs: hardware, redes o software constituyen la plataforma donde se desarrollan y utilizan los sistemas de información. Dicho de otra manera, los sistemas de información generan los reportes que son procesados, almacenados, analizados y distribuidos a través de aplicaciones o sistemas informáticos que se encargan de automatizar todo tipo de informaciones generadas dentro de la organización. Estas aplicaciones pueden clasificarse como:

- **Sistemas de Procesamiento de Transacciones (TPS):** Gestiona la información referente a las transacciones producidas en una empresa u organización, también se le conoce como Sistema de Información Operativa.
- **Sistemas de Información Gerencial (MIS):** Orientados a solucionar problemas empresariales en general.
- **Sistemas de Soporte a la Toma de Decisiones** (**DSS**): Herramienta para realizar el análisis de las diferentes variables de negocio con la finalidad de apoyar el proceso de toma de decisiones.
- **Sistema de Información Ejecutiva (EIS):** Herramienta orientada a usuarios de nivel gerencial, que permite monitorizar el estado de las variables de un área o unidad de la empresa a partir de información interna y externa a la misma. Es en este nivel cuando los sistemas de información manejan información estratégica para las empresas**.**
- **Sistemas de Automatización de Oficinas (OAS):** Aplicaciones destinadas a ayudar al trabajo diario del administrativo de una empresa u organización.
- **Sistemas de Planificación de Recursos (ERP):** Integran la información y los procesos de una organización en un solo sistema.
- **Sistemas Expertos (SE):** Simulan el comportamiento de un experto en un dominio concreto.

1.1.3 Sistema de información en la dirección de Mantenimiento

El mantenimiento es la realización de tareas o actividades que permitan reparar una unidad funcional para que ésta pueda continuar cumpliendo sus objetivos, e incluyen acciones de inspección, comprobaciones, clasificación, reparación, etc.(Angell, y otros, 2011)

El mantenimiento tiene varios niveles:

Capítulo 1. Fundamentación Teórica

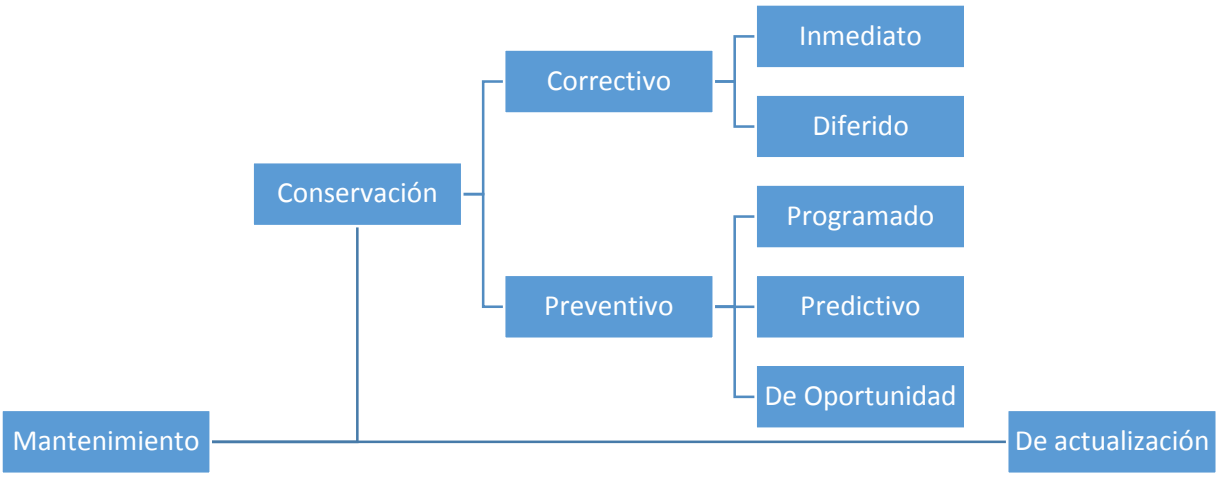


Figura 1.1 Jerarquía de mantenimiento

- De conservación: destinado a compensar el deterioro alcanzado por el uso.
- Correctivo: que corrige defectos o averías para prevenir fallas de mayor nivel.
- Correctivo inmediato: es el que se realiza inmediatamente de percibir la avería y defecto, con los medios disponibles, destinados a ese fin.
- Correctivo diferido: al producirse la avería o defecto, se produce un paro de la instalación o equipamiento de que se trate, para posteriormente afrontar la reparación, solicitándose los medios para ese fin.
- Preventivo: como el destinado a garantizar la fiabilidad de equipos en funcionamiento antes de que pueda producirse un accidente o avería por deterioro.
- Mantenimiento programado: el que se ejecuta según un programa o cronograma específico.
- Mantenimiento predictivo: que realiza las intervenciones prediciendo el momento que el equipo quedara fuera de servicio mediante un seguimiento de su funcionamiento determinando su evolución, y por tanto el momento en el que las reparaciones deben efectuarse.
- Mantenimiento de oportunidad: que es el que aprovecha las paradas o periodos de no uso de los equipos para realizar las operaciones de mantenimiento, realizando las revisiones o reparaciones necesarias para garantizar el buen funcionamiento de los equipos en el nuevo periodo de utilización.

- Mantenimiento de actualización: cuyo propósito es compensar la obsolescencia tecnológica, o las nuevas exigencias, que en el momento de construcción no existían o no fueron tenidas en cuenta pero que en la actualidad si tienen que serlo.

Tanto los mantenimientos preventivos como los correctivos son realizados por la Dirección de Mantenimiento de la Universidad. La gestión de mantenimiento se realiza a partir de reportes, emitidos por los usuarios afectados. Estos reportes constituyen el área principal entro de la Dirección de Mantenimiento ya que desencadenan todo el flujo de gestión de esta unidad.

Reporte

Un reporte es un documento que a partir de los objetivos de quien lo crea, pretende transmitir información. Concretamente, dentro de la estadística y la toma de decisiones. Al concepto de reporte se le añade las siguientes condiciones(Duggan, et al., 2012):

- La información debe ser estructurada y organizada: La información obtenida en el reporte debe tener una estructura lógica, con organización coherente que garantice que su receptor puede comprenderla fácilmente.
- Simplificación de las fuentes: Un reporte debe sintetizar la información de las fuentes, sin importar la naturaleza de éstas, añadiendo valor a la información ya sea mediante el resumen, la representación gráfica o la agrupación de elementos de interés.

Los reportes pueden ser de diversos tipos o estar soportados en diversos formatos (documentos impresos, digitales, audiovisual, etc.). En el ámbito de la informática, los reportes son informen que se organizan y muestran información a partir de fuentes de datos digitales diversas (textos, bases de datos, etc.). Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. En esta dimensión, los reportes contienen otras características (van der Graaf, 2007):

- Son generados dinámicamente: Los reportes son extraídos mediante acciones simples sin necesidad de conocer la naturaleza de las fuentes. La selección, agrupación y representación de la información es realizada por el ordenador según las exigencias del usuario. Por ejemplo, un programa que presente tendencias de ventas accede a la base de datos transaccional, obtienen las ventas diarias y construye una gráfica a partir de los valores.

- A diferencia de un formulario, los datos de los reportes no pueden ser modificados: Los reportes representan información de salida que no puede ser reajustada para ingresarlas nuevamente a las fuentes. Esta información debe ser modificada en otras partes del sistema para que sus cambios se vean reflejados cuando se genera el reporte nuevamente.
- Los reportes son específicos por áreas de interés dentro del sistema: El reporte contiene la capacidad de crear etiquetas y responder a éstas. Ejemplo, una etiqueta de compras permite reflejar en proceso de compras de una empresa a través de reportes gráficos o resúmenes en tablas.

1.2 Sistemas informáticos para la gestión de mantenimientos

1.2.1. Software para la gestión de mantenimiento

La mayoría de los sistemas de información diseñados para la gestión de mantenimiento, se originaron en empresas que a partir de sus propios objetivos necesitaran sistemas capaces de supervisar la reparación de sus principales equipos. Esta relevancia se notó inicialmente en grandes complejos industriales, industria militar y navieras. Esta área de los sistemas de información se conoce como Gestión de Mantenimiento Asistido por Computadora (Berger, 2006).

Estos son sistemas que ayudan a las empresas a gestionar sus servicios de mantenimiento. Básicamente son una base de datos operacional que contiene un registro de los servicios de mantenimiento y las acciones para realizarlos. Contribuyen a elevar la eficiencia de la empresa y brindan información para la gestión de materiales y toma de decisiones.

Las plataformas de gestión del mantenimiento asistido por computadora pueden ser utilizadas por cualquier organización que necesite gestionar el mantenimiento de sus equipos, activos y propiedades. Algunas de las soluciones existentes están enfocadas a mercados específicos (mantenimiento de flotas de vehículos, infraestructuras sanitarias, etc.) aunque también existen productos que están enfocados a un mercado general.

El software ofrece una amplia variedad de funcionalidades, dependiendo de las necesidades de cada organización, existiendo en el mercado un gran rango de precios. Puede ser tanto accesible vía web, mientras que la aplicación se encuentra alojada en los servidores de la empresa que vende el producto o de un proveedor de servicios TI o accesible vía LAN si la empresa adquisidora del producto lo aloja en su propio servidor.

Existen sistemas específicos y sistemas comerciales para este fin, algunos de los comerciales más notables del mercado son:

SAP *One Maintenance*: Es una solución del paquete empresarial *SAP Business One* que permite a las empresas que utilizan el sistema dar mantenimiento preventivo y correctivo a sus áreas a partir de una funcionalidad que se enlaza a todos los activos del negocio. El componente está al alcance de toda la organización y ofrece como principales ventajas:

- Optimización de los recursos
- Laborales: Mejora de la planificación, seguimiento y aplicación.
- Materiales: Mayor disponibilidad, disminución de existencias, fácil localización.
- Mejoras en la calidad y productividad de la organización.
- Disminución de los tiempos de paro en elementos productivos. Mayor fiabilidad y disponibilidad.
- Información actualizada, inmediata de todos los componentes del proceso.
- Mejora de los procesos de actuación establecidos.
- Posibilidad de realizar estudios y anticipar cargas de trabajo o consumo de piezas.
- Conocimiento inmediato de los gastos originados por cualquiera de los elementos controlados.
- Ajuste de los planes de mantenimiento a las características reales.
- Permitir la participación en un TPM
- Trazabilidad del equipamiento.
- Posibilidad de implementar cualquiera de las metodologías de mantenimiento existentes.
- Mejor control de actividades subcontratadas.

GMAO es un paquete de asistencia mantenimiento con módulos independientes destinados a distintos fines del sector industrial, como los equipos de ensamblaje, líneas de producción, lubricación y otros. Incluye planes programados, preventivos y correctivos además de presentar sistemas de comunicación con terceros para su integración a software de supervisión y control (SCADAs). Aunque se basa en tecnologías en desuso, muchos clientes lo utilizan debido a que es un software ligero, aunque no permite emitir reportes desde distintas áreas y su manejo logístico es pobre e ineficiente.

Máximo IBM es un sistema integral para la gestión de mantenimiento creado para los entornos laborales de la compañía IBM. El sistema presenta operaciones para órdenes de trabajo, historiales, clasificación de equipos, jerarquías organizativas, planificación y control logístico.

Una de las características fundamentales de estos sistemas es que deben asegurar resúmenes y reportes sobre el estado de las empresas y la gestión del mantenimiento.

1.2.2. Sistemas informáticos para la gestión de reportes

Los reportes representan un elemento sustancial en los sistemas de información y dentro de ámbito de las aplicaciones informáticas se define como un informe que organiza y exhibe la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. A través de los reportes se refleja el comportamiento de los diferentes componentes que integran cualquier sistema de información de una empresa o institución, permitiendo el reconocimiento, control y monitoreo de las diversas problemáticas existentes en la misma (Meyer, 2009). Algunos de los beneficios que se obtienen con los reportes son:

- Búsqueda fácil de información: Los reportes permiten filtrar la información y mostrar sólo los datos necesarios.
- Fácil manejo de datos: Con la ayuda de la herramientas de diseño es posible manipular los datos de tal forma que, para cada diseño se muestre información distinta ya sea con la misma y/o mayor o menor cantidad de datos.
- Mejor visualización de la información: Al tener la posibilidad de hacer su propio diseño de los reportes, podrá mejorar la visualización de los mismos y tener reportes que se ajustan a sus necesidades. A continuación se muestran algunos ejemplos de sistemas informáticos que tienen incorporados la generación de reportes como valor agregado:

Algunos de los sistemas que utilizan de forma eficiente la gestión de reportes son:

CENACAD: El Sistema de Censo Académico en Línea para la Automatización de la Evaluación a los Docentes, fue realizado por parte del personal de la Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, Ecuador. Nace de la necesidad de brindar una solución informática sobre las evaluaciones docentes en la Escuela Superior Politécnica del Litoral (ESPOL). Cuenta con un módulo de reportes el cual permite a los usuarios la obtención de diferentes tipos de reportes, a continuación se muestran algunos de ellos ((CEEC, 2012):

- Listado de mejores docentes en base a los mejores promedios obtenidos en la ESPOL por el período de un año.

- Promedio de la(s) facultad(es) o instituto(s).
- Promedios obtenidos en cada área que conforman cada una de las encuestas realizadas a cada materia.

Info@tletas: Aplicación web oficial del Instituto Nacional de Deportes, Educación Física y Recreación (INDER). Surge como colofón de varias aplicaciones de escritorio confeccionadas según la solicitud de la dirección de alto rendimiento del INDER. Se encarga principalmente de la administración de los expedientes técnico-acumulativos de atletas y entrenadores nacionales e internacionales. La aplicación cuenta con 12 módulos, entre los que se incluye el Módulo de Reportes encargado de generar reportes estadísticos de atletas y entrenadores visualizados de forma tabular y gráfica de barras(info@tltas).

Los reportes se comportan según el tipo de usuario que ha iniciado su sesión en el sistema:

- Supervisor Especial y Supervisor Nacional: visualizan los reportes a nivel nacional.
- Supervisor Provincial: visualizan los reportes a nivel provincial.
- Administrador, Metodólogo y Técnico: visualizan los reportes a nivel de centro.

CDS Report Server: Es un sistema basado en el acceso por roles, el usuario solo podrá visualizar la información y reportes de su interés. Además de esto, se generan formularios de carga de datos, accedidos desde un portal, para la carga de información relacional que permite resolver la desvinculación de los datos obtenidos de los sistemas de planta actuales. Tanto los informes como los formularios de alta, baja y modificación de datos, son visibles para roles específicos del sistema. Estos roles son asignados y administrados por un usuario administrador y se corresponden a los usuarios y grupos creados como cuentas de dominio en la empresa.

Sistemas ERP: En los sistemas ERP los reportes existen una gran cantidad de reportes dependiendo del módulo en que se encuentren. Los reportes suelen ser sencillos o completos de acuerdo el usuario decida. Por lo general los sistemas ERP ya cuentan con un reporte por defecto para cada proceso. Asimismo pueden contener desde un práctico listado con los datos más relevantes hasta uno más detallado y completo con gráficos(Pérez, 2014). Es importante destacar que el usuario puede diseñar sus propios reportes, quitar o agregar datos para mostrarlos de la manera que le sea más conveniente. Hacer el diseño de reportes es una labor muy sencilla y que cualquier usuario del sistema puede realizar de manera muy práctica ya que, por lo general, los ERP se valen de herramientas de diseño muy amigables

e intuitivas. Algunos ERP tienen un grupo de reportes para cada módulo, en otros casos, pueden contener un módulo específico de reportes por ejemplo:

- Kardex: Genera reportes de artículos, movimientos de inventario, reportes de ventas, balance general.
- OpenBravo: Es una solución económica que genera reportes a partir de la administración de todo el negocio de una empresa a través de finanzas, ventas, clientes y operaciones, todo en un solo sistema.

1.3. Metodología para el desarrollo de Software

Una metodología es un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo(Addison Wesley object technology series, 2004). Se utiliza principalmente para:

- Facilitar la tarea de planificación.
- Facilitar la tarea del control y seguimiento de un proyecto.
- Mejorar la relación coste/beneficio.
- Optimizar el uso de recursos disponibles.
- Facilitar la evaluación de resultados y cumplimiento de los objetivos.
- Facilitar la comunicación efectiva entre usuarios y desarrolladores.

Actualmente no se puede decir que existe una metodología única para garantizar el éxito de cualquier proyecto de desarrollo de software, debido a que estas deben ajustarse al tipo de proyecto que se desarrolle. Teniendo en cuenta que existen fundamentalmente dos tipos de metodologías para el desarrollo de software: Las metodologías tradicionales, también conocidas como metodologías pesadas y las metodologías ágiles o ligeras.

Dada las características del sistema a desarrollar y la cantidad de personal disponible se decide utilizar una metodología ágil, para potenciar la obtención de un producto funcional por encima de una extensa documentación. Las metodologías candidatas se definen a partir de la experiencia acumulada en investigaciones realizadas en la universidad (Extreme Programming (XP) y OpenUp que es una metodología ágil que se basa en RUP.

1.3.1. Proceso Unificado Abierto (OpenUP)

OpenUP es un proceso que mantiene las mismas características de RUP, que contiene el desarrollo iterativo, casos de uso y escenarios de conducción de desarrollo, gestión de riesgos y el enfoque centrado en la arquitectura. Open Up es un proceso extensible, dirigido a la gestión y desarrollo de proyectos de software basados en desarrollo iterativo, ágil e incremental apropiado para proyectos pequeños o de bajos recursos, aplicable a un conjunto amplio de plataformas de desarrollo. Basada en la metodología de Proceso Unificado de Rational (*Rational Unified Process*) RUP. Es el subconjunto de esta última que contiene el conjunto mínimo de prácticas que ayudan a un equipo de desarrollo de software a realizar un producto de alta calidad. Utiliza un punto de vista pragmático, una filosofía ágil que se centraliza en la naturaleza colaborativa del proceso de desarrollo del software. Una de sus principales características es su alto grado de adaptabilidad a las necesidades de un proyecto en particular.

OpenUp es un proceso de desarrollo iterativo del software que es mínimo, completo y extensible. Es mínimo en que solamente el contenido fundamental es incluido; es completo en que puede ser manifestado como todo el proceso para construir un sistema; extensible en que puede ser utilizado como fundamento sobre el cual el contenido de proceso se pueda agregar o adaptar según lo necesitado (Addison Wesley object technology series, 2004).

Como metodología de desarrollo es conducida por el principio de colaboración para alinear intereses y para compartir su comprensión. Es el proceso unificado que aplica acercamientos iterativos e incrementales dentro de un ciclo vital estructurado. En la Figura 1.2 se muestran las capas de OpenUp.

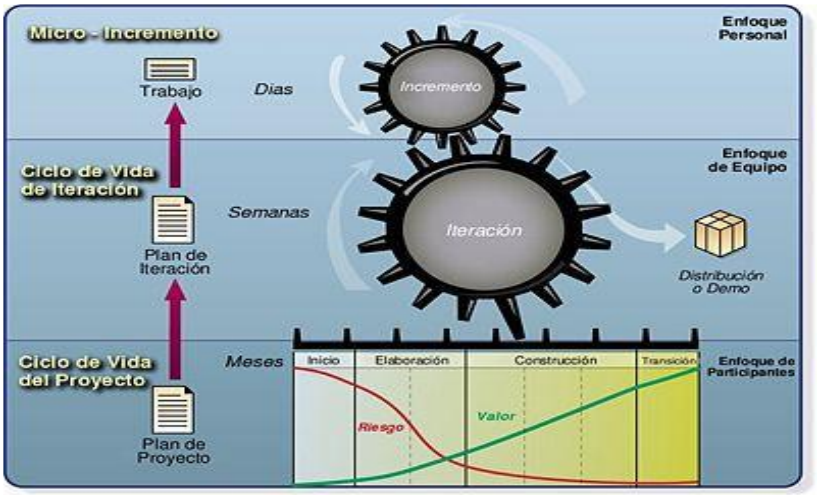


Figura 1.2: Capas de OpenUP

El ciclo de vida del proyecto provee a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos. El OpenUp estructura el ciclo de vida de un proyecto en cuatro fases: inicio, elaboración, construcción y transición.

1. Concepción: primera fase en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de ésta fase es capturar las necesidades de los *stakeholder* (clientes) en los objetivos del ciclo de vida para el proyecto.
2. Elaboración: se trata los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base la elaboración de la arquitectura del sistema.
3. Construcción: esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la arquitectura definida.
4. Transición: es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, evalúa la funcionalidad y rendimiento del último entregable de la fase de construcción.

A continuación en la Figura 1.3 se muestran las fases de OpenUp.

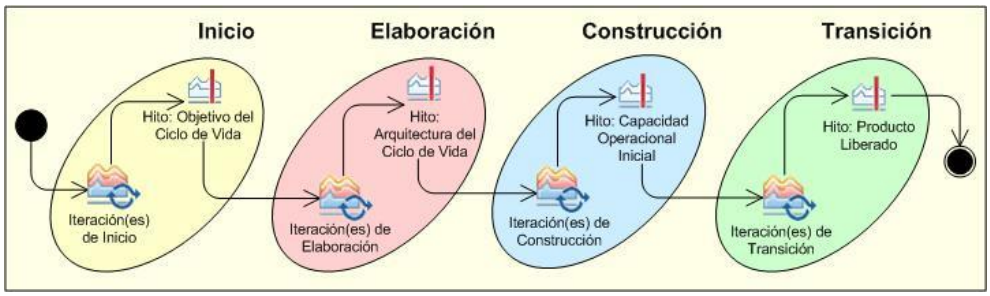


Figura 1.3: Fases de OpenUP

Después del análisis de esta metodología se decidió seleccionarla para el proceso de desarrollo, ya que OpenUp es iterativo e incremental. Es apropiado para proyectos pequeños y de bajos recursos. Permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito. Permite detectar errores tempranos a través de un ciclo iterativo. Evita la elaboración de documentación, diagramas e iteraciones innecesarias. Por ser una metodología ágil tiene un enfoque

centrado al cliente con iteraciones cortas. Es ligero y proporciona una comprensión detallada del proyecto, beneficia a clientes y desarrolladores sobre el producto a entregar y su formalidad. Es extensible ya que el proceso se puede agregar o adaptar según lo vayan requiriendo los sistemas.

1.4. Lenguaje de modelado

Un lenguaje de modelado visual se utiliza para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Con ellos es posible diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Entre los más conocidos se encuentran: UML, MOF, Ecore, entre otros (Gerner, 2011).

1.4.1. Leguaje de Modelado Unificado (UML 2.0)

El Lenguaje de Modelado Unificado (en inglés *Unified Model Language*), incluye conceptos semánticos, notación y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. UML es un lenguaje para hacer modelos y es independiente de los métodos de análisis y diseño. Existen diferencias importantes entre un método y un lenguaje de modelado. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como, generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos(Fowler, 2007). Es una consolidación de muchas de las notaciones y conceptos más usados en la modelación orientada a objetos. Es importante recalcar que UML no es una guía para realizar el análisis y diseño orientado a objetos, no es un proceso, sino un lenguaje que permite la modelación de sistemas, con tecnología orientada a objetos. Desde el año 1995, UML es un estándar aprobado por la ISO como ISO/IEC 19501:2005 *Information Technology*; está respaldado por el *Object Management Group* (OMG).
Un modelo UML está compuesto por tres clases de bloques de construcción:

- Elementos: Los elementos son abstracciones de cosas reales o ficticias (objetos, acciones, etc.)
- Relaciones: relaciona los elementos entre sí.
- Diagramas: Son colecciones de elementos con sus relaciones.

Permite la modelación del ciclo completo de desarrollo de software y contiene: Los Diagramas de casos de uso, clases, objetos, secuencia, colaboración, estado, actividades, despliegue y componentes.

Los principales beneficios de UML son:

• Mejores tiempos totales de desarrollo (de 50 % o más).

• Modelar sistemas (y no sólo de software) utilizando conceptos orientados a objetos.

• Establecer conceptos y artefactos ejecutables.

• Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.

• Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.

• Mejor soporte a la planeación y al control de proyectos.

• Alta reutilización y minimización de costos.

1.5. Herramienta Case

Las herramientas CASE, Ingeniería de Software Asistida por Computadoras (por sus siglas en inglés *Computer Aided Software Engineering*) son un conjunto de programas y ayudas que propician el desarrollo de programas informáticos a los ingenieros de software y desarrolladores. Las mismas permiten automatizar diferentes tareas que se realizan durante el ciclo de vida de desarrollo del software, tales como: la documentación, la generación de código y diseños, las pruebas de errores y la gestión del proyecto. Permite la estandarización de la documentación y la reutilización del software. Las herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costos de las mismas en términos de tiempo y de dinero. Estas herramientas son de mucha ayuda en todos los aspectos del ciclo de vida de desarrollo del software en tareas como el proceso de realizar un diseño del proyecto, cálculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores(Fowler, 2007). Dentro de las múltiples herramientas Case en el mercado, la Universidad utiliza Visual Paradigm.

1.5.1. Visual Paradigm for UML 8.0

Es una herramienta considerada muy completa, fácil de usar, con soporte multiplataforma y de probada utilidad para el analista, proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Posee un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, análisis y diseño, hasta la generación del código fuente y la documentación. Tiene licencia dual, gratuita y comercial. Esta herramienta se utilizará para visualizar y

diseñar los elementos de nuestra aplicación debido a que es multiplataforma. Se puede intercambiar diagramas UML y modelos con otras herramientas. Posee facilidades para el diseño de los diagramas necesarios y su documentación, soporta las últimas versiones de UML y la Notación y Modelado de Procesos de Negocios, a través de la utilización de un enfoque Orientado a Objetos con abundante documentación. En adición al soporte de Modelado UML, Visual Paradigm provee un generador de mapeo de objetos-relacionales para los lenguajes de programación como Java, .NET y PHP. Una de sus principales ventajas es que incorpora el soporte para trabajo en equipo, lo que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (Visual Paradimg Group, 2011). Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación (Visual Paradimg Group, 2011).

Se propone el uso de Visual Paradigm como herramienta Case por las ventajas que ofrece al entorno de desarrollo, facilitando el trabajo para el desarrollador con la generación de los artefactos requeridos por la metodología escogida, además de tener un amplio espectro en las modelaciones de clases necesarias para la definición de la arquitectura del sistema.

1.6. Lenguaje de programación

Los lenguajes de programación representan un idioma artificial diseñado para expresar procesos que pueden ser llevados a cabo por maquinas como las computadoras. Pueden utilizarse para la crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación. Están formados por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura. La selección de cada lenguaje se realiza de acuerdo a las características y al tipo de software a desarrollar (Meyer, 2009)

1.6.1. JavaScript

JavaScript es un lenguaje de programación interpretado, ligero y orientado a objetos., se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. A pesar de su nombre, no guarda ninguna relación directa con el lenguaje de programación Java, tienen semánticas y propósitos diferentes. Los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Java es un lenguaje de programación con el que se puede realizar

cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems con gran dedicación y siempre enfocado a cubrir las necesidades tecnológicas más punteras. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del *Document Object Model* (DOM) (Pérez, 2014). JavaScript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, con funciones y estructuras de datos complejas. Además, pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente. Este lenguaje por sus características y posibilidades permite un desarrollo completo de cualquier aplicación web por compleja que sea, facilita el manejo de información en el navegador sin necesidad de realizar llamadas redundantes a una base de datos.

1.6.2. PHP

Este lenguaje está diseñado especialmente para incrementar el dinamismo de las páginas web, aprovechando los recursos de las redes informáticas y los escasos requerimientos de hardware que solicita el lenguaje para que sus aplicaciones funcionen correctamente. La necesidad que tienen los sitios web de ser robustos, dinámicos y flexibles ha hecho que PHP se convirtiera en uno de los lenguajes de desarrollo más difundidos (Merley, 2012). PHP es el acrónimo de *Hypertext Preprocessor* (Preprocesador de Hipertexto). Se trata de un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor. Está muy orientado al desarrollo de aplicaciones web y permite insertar contenidos dinámicos en las páginas. Con PHP3 y PHP4 se había conseguido una plataforma potente y estable para la programación de páginas del lado del servidor. Haciendo posible que PHP sea el lenguaje más utilizado en la web para la realización de páginas avanzadas. Con la versión 5.3 se solucionaron algunos aspectos que se echaron en falta en versiones anteriores. El principal objetivo de PHP 5 ha sido mejorar los mecanismos de programación orientada a objetos. Un paso necesario para lograr que PHP sea un lenguaje apto para todo tipo de aplicaciones y entornos. Es el lenguaje más utilizado en la web para la realización de páginas avanzadas para todo tipo de entornos. Se caracteriza por ser Multiplataforma, se ejecuta del lado del servidor, con licencia GPL o de software libre, presenta una sintaxis cómoda orientado a objetos completamente, presenta un gran número de extensiones o bibliotecas que amplían sus funcionalidades (Merley, 2012). También ofrece la integración con varias librerías externas, que permiten que el desarrollador haga casi cualquier cosa, desde generar documentos en PDF hasta analizar código XML.

1.6.3. CSS 3

Las Hojas de Estilos en Cascada (CSS, por sus siglas en inglés) es un lenguaje creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas Web complejas. Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien formateados y con significado completo (también llamados documentos semánticos). Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes (Pérez, 2014).Al crear una página Web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, entre otros. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, además de otras funcionalidades.

1.6.4. HTML 5

De las siglas de (*Hyper Text Markup Language*) Lenguaje de Marcas de Hipertexto, es el lenguaje de marcado más utilizado para la construcción de páginas web. Es un lenguaje de composición de documentos y especificación de ligas de hipertexto que define la síntesis y coloca instrucciones especiales que no muestra el navegador, aunque si le indica como desplegar el contenido del documento, incluyendo texto imágenes y otros medios soportados. Su versión 5 brinda más facilidades que sus antecesores, lo que permite el desarrollo de la aplicación con una mayor vistosidad, calidad e integración, este es soportado por las versiones actuales de los navegadores Mozilla Firefox, Chrome, Chromiun, Safari e Internet Explorer(Fernando, 2010).

1.7. Framework

Los *framework* de desarrollo se definen como un conjunto de herramientas destinadas a la construcción de un determinado tipo de aplicación de manera generalizada. Representan una estructura conceptual que constituye la base para una aplicación. Su utilización simplifica el desarrollo de un sistema mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes; evita la re-implementación de funcionalidades que resultan ser frecuentes en este tipo de sistemas y que

tradicionalmente incluyen administración de usuarios, persistencia de datos, motores de plantillas, entre otras; facilita la programación de aplicaciones, puesto que encapsula operaciones complejas en instrucciones sencillas y además, proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener(De los Ángeles, 2012).

1.7.1. Symfony 1.4

Symfony es un completo *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Symfony se diseñó para que se ajustara a los siguientes requisitos(Potencier, et al., 2008):

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de *"convenir en vez de configurar"*, en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de *mejores prácticas* y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

1.7.2. Ext JS Framework 3.2.1

Ext JS es una librería de JavaScript ligera, de alto rendimiento para el desarrollo rápido de aplicaciones web cross-browser[1]. Presenta una interfaz de usuario personalizable "*widgets*", bien diseñada y contiene un modelo de componentes extensibles. Su API[2] es intuitiva y fácil de usar (Ext JS Inc). Posee licencias comerciales y de código abierto. Es compatible con la mayoría de los navegadores, y su familia de productos es utilizada por miles de compañías en el mundo. Además de que posee una amplia documentación u ayuda que propicia que su aprendizaje sea altamente rápido y fácil.

1.8. Modelo EBMS DSerp.

El modelo EBMS DSerp es un núcleo de sistemas de gestión creado con la tecnología Symfony y Ext por el Centro de Informática Industrial. El sistema propone una plataforma para la gestión de recursos y procesos a la que se puede adaptar cualquier negocio, a partir de un marco que define las funcionalidades de seguridad, comunicación entre módulos, conexión a terceros e interfaces externas autoconstruidas (Sistema estratégico para la gestión avanzada de recursos y procesos EBMS DSerp Agro, 2013). Este marco ha sido utilizado por diversos proyectos de la universidad y proyectos nacionales, y su elección se fundamenta en:

- Propone una estructura de desarrollo documental, robusta y con una arquitectura por capas que asegura la obtención de un producto de calidad siguiendo patrones.
- Establece un mecanismo de seguridad dinámico, basado en roles y usuarios, altamente configurable y con 4 niveles de seguridad, además de prevenirse contra los principales ataques definidos para sitios web.
- Permite la adaptación a nuevos escenarios de negocios, la integración de soluciones, es altamente escalable y estable.
- Incluye componentes para la generación de gráficos, tablas y reportes a partir de plantillas de código que aceleran el proceso de desarrollo.
- Sobre dicha plataforma se han desarrollado casos de éxito dentro y fuera de la universidad y sus creadores radican en la misma haciéndose más fácil el proceso de soporte y actualización del sistema.

[1] Cross-browser: Aplicación web que tiene la habilidad de correr en todos los navegadores web.
[2] *API*: *Application Programming Interface* en inglés o Interfaz de Programación de Aplicaciones en español.

El sistema actúa de núcleo para el desarrollo de aplicaciones modulares que pueden ser integradas entre sí. Comprende las siguientes versiones de la tecnología:

- Symfony 1.4.11
- Ext JS 3.2.1
- PostgreSQL 9.1.2

Aunque actualmente se desarrollan para versiones más actuales de las tecnologías base (Symfony y ExtJs), el núcleo con las versiones descritas es estable y responde adecuadamente a los intereses del negocio que se considera en la presente investigación. Las versiones más actuales del modelo EBMS DSerp aún están en versión Beta y su uso aún no es recomendado por sus desarrolladores.

1.9. Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado, (de sus siglas en inglés *Integrated Development Environment*) consiste básicamente en un software cuyo principal objetivo es el desarrollo de otro software. Es un entorno de programación que ha sido empaquetado como un programa de aplicación. Puede ser exclusivo para un lenguaje de programación o bien, para varios. Suele consistir de un editor de código (con facilidades como resaltado de sintaxis, completamiento de código y navegación entre clases), un compilador y herramientas de automatización de la compilación, un depurador y en algunos casos un constructor de interfaz gráfica. Actualmente los entornos de desarrollo proporcionan un marco de trabajo para la mayoría de los lenguajes de programación existentes en el mercado(Meyer, 2009).

1.9.1. IDE NeatBeans 7.4

Neatbeans es un entorno de desarrollo integrado libre, extensible para el desarrollo sobre muchos lenguajes, aunque se realizó fundamentalmente para el lenguaje Java[3]. Es un producto de código abierto desarrollado por la compañía Sun MicroSystem en el año 2000. Se basa en una filosofía modular, lo que permite el desarrollo de múltiples proyectos con el uso de varias tecnologías (Domínguez, 2004). Permite a los programadores escribir, compilar, depurar y ejecutar programas. Provee la implementación del patrón de arquitectura Modelo-Vista-Controlador, además de estar enfocado para el desarrollo del lenguaje dinámico de programación PHP. Es fácil de instalar y configurar en la mayoría de las plataformas. Entre sus principales ventajas se encuentran:

[3] Java: Lenguaje de programación orientado a objetos desarrollado por Sun MicroSystem en los años 90.

- Administración de interfaces de usuario.
- Integración a múltiples frameworks.
- Administración de almacenamiento.
- Gran cantidad de módulos y extensiones para múltiples lenguajes y tecnologías.
- Fuerte comunidad de respaldo.

Es utilizado en la aplicación debido a que brinda la posibilidad de facilitar y automatizar una gran cantidad de funcionalidades que resultan comunes en las aplicaciones web.

1.10. Servidor web

Un servidor web es un programa que implementa el protocolo HTTP y se encarga de mantenerse a la espera de peticiones llevadas a cabo por un cliente que solemos conocer como navegador. El navegador realiza una petición al servidor y éste le responde con el contenido que el cliente solicita. Este protocolo está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML: textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de sonidos (Gerner, 2011).

1.10.1. Servidor Web Apache

Apache es la plataforma de servidores web de código fuente abierto más poderosa del mundo .Su alta configuración, robustez y estabilidad hacen que más del 64% de toda web reiteren su confianza en este programa (ASF). Es un servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos. Apache es una de las plataformas de servidores Web más destacadas dentro de la gran cantidad de servidores web que existen en el planeta, de código fuente abierto para plataformas Unix (BSD, GNU/Linux, etc.), Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Es una tecnología gratuita, multiplataforma, altamente configurable y muy sencilla de utilizar e implementar, está estructurada en módulos, los cuales pueden clasificarse en tres categorías específicas para su utilización con disímiles funcionales, la elevada seguridad que le provee a sus clientes es una de sus grandes características y en la actualidad domina ampliamente el mercado de la Web con respecto a los demás servidores HTTP (Apache Software Fundation). Apache es uno de los servidores HTTP más utilizado en la mayoría de los sitios web en el mundo. Entre sus características principales se encuentran:

- Tecnología gratuita de código fuente abierta.

- Personalizable, la arquitectura modular de Apache permite construir un servidor hecho a la medida y posibilita la implementación de los últimos y nuevos protocolos.
- Servidor altamente configurable de diseño modular. Permite aumentar fácilmente su capacidad e instalar cualquier módulo para cumplir una función específica.
- Tiene la infraestructura necesaria para servir distintos protocolos.
- Rapidez y estabilidad en sistemas que no son tipo Unix, tales como BeOS, OS/2 y Windows.
- Nueva interfaz de programación (API) para los módulos, muchos de los problemas de ordenación y prioridad de módulos de la versión 1.3 desaparecieron.
- Los módulos de Apache pueden escribirse para que se comporten como filtros que actúan sobre el flujo de contenidos tal y como salen del servidor, o tal y como son recibidos por el servidor.

1.11. Sistema gestor de Base de Datos

Los sistemas gestores de bases de datos, abreviado SGBD, permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades de estos SGBD, así como su utilización y administración, se estudian dentro del ámbito de la informática. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Tiene el objetivo de suministrarle al usuario las herramientas que le permitan manipular, en términos abstractos, los datos; o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado, garantizando la seguridad e integridad de la información (Arregoces, et al., 2011)

1.11.1. PostgreSQL 9.1.2

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) que ha sido desarrollado de varias formas desde 1977 basado en el proyecto POSTGRES, de la universidad de Berkeley. PostgreSQL es un servidor de base de datos relacional libre, liberado bajo la licencia de Distribución de Software Berkeley (*Berkeley Software Distribution*) BSD. Está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle o DB2. Utiliza un modelo cliente/servidor y un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

Su distribución es libre, sin necesidad del pago de alguna licencia, lo que constituye una de las ventajas más atractivas. Algunas de sus principales características son(PostgreSQL Development Group, 2009):

- Claves ajenas también denominadas Llaves ajenas o Llaves Foráneas.
- Disparadores.
- Implementación del estándar SQL92/SQL99.
- Vistas.
- Integridad transaccional.
- Acceso concurrente multiversión (no se bloquean las tablas, ni siquiera las filas, cuando un proceso escribe).
- Capacidad de albergar programas en el servidor en varios lenguajes.
- Herencia de tablas.
- Tipos de datos y operaciones geométricas.

Por el manejo de los datos del sistema propuesto, la compatibilidad con el lenguaje de programación, la integración con el *framework* de desarrollo Symfony, las ventajas con la arquitectura cliente-servidor, es un sistema potente de código abierto que permite el constante desarrollo y perfeccionamiento de sus funcionalidades y está publicado bajo la licencia BSD que pertenece al grupo de licencias de software libre se escoge como gestor de base de datos para el desarrollo del sistema PostgreSQL en su versión 9.1.2.

1.12 Conclusiones parciales

En este capítulo se definió un marco teórico sobre el proceso de mantenimiento y gestión de reportes, a partir de sus conceptos y herramientas, lo que permitió identificar las necesidades del Sistema de gestión de reportes de la Dirección de Mantenimiento. El análisis de las herramientas para la emisión de reportes contribuyeron a definir la forma en que debía orientarse la información para dar valor a los datos que almacenará el sistema. Se realizó un estudio de las principales herramientas y tecnologías que se utilizaron en la primera versión del Sistema de gestión de reportes, así como el marco EBMS DSerp, sobre la cual se trabajará en esta nueva versión. Además se concluyó que las versiones actuales de las tecnologías seleccionadas integradas al marco anteriormente mencionado satisfacen las necesidades de la investigación.

Capítulo 2. Análisis y diseño del sistema

Introducción

En el presente capítulo se detallan las consideraciones del negocio tenidas en cuenta durante la fase de análisis de la solución, partir del estudio del proceso de reportes en la Dirección de Mantenimiento. Se analizan las condiciones actuales del Sistema de Gestión de Reportes desarrollado en años anteriores y se argumentan los requisitos identificados para esta nueva versión. Se exponen los elementos del diseño de la aplicación, la arquitectura, la interacción de sus componentes y el modelo de datos asociados. Por último, se explica cómo quedará la aplicación en su entorno de despliegue.

2.1 Análisis

2.1.1 Proceso de registro y atención de reportes

De forma general, el proceso de reportes se encarga de administrar, controlar y registrar grupos de incidencias que ocurren en un determinado proceso dentro de un ambiente empresarial, se clasifican según la finalidad que persiga:

- Reportes Informativos: Se emplean cuando se necesita proporcionar información acerca de datos específicos.
- Reportes de daños: Se utiliza para avisar, informar sobre algún problema o percance que se tenga dentro de la empresa.
- Reportes de mantenimiento: Se generan para mantener informados a los interesados del estado que puedan presentarse en equipos en la empresa.

La gestión de reportes de daños representa uno de los procesos claves en la administración y control de reportes de mantenimiento, debido a información que se genera acerca de lo que suceda a un determinado equipo que sea manipulado, entre mejor detallado se encuentre el reporte de daño mayor velocidad en las gestiones atención del departamento encargado, permitiendo que se pueda entender el problema que se reporte y como solucionarlo. De manera general los reportes de daño contienen los siguientes campos definidos:

1. Fecha
2. Descripción del incidente

3. Como se detectó:
4. Describir lo que se encontró
5. Consecuencias del incidente
6. Primeras medidas tomadas:
7. Nombre del equipo
8. Ubicación
9. Departamento

2.1.2 Gestión de reportes en la Dirección de Mantenimiento

En la Universidad de Ciencias Informáticas el proceso de gestión de reportes comienza cuando el técnico de turno recibe un reporte realizado por una persona desde un área de la universidad. El técnico debe recoger del reporte los siguientes datos:

- Nombre del usuario que emite el reporte
- Número de Solapín del usuario.
- Descripción del reporte.

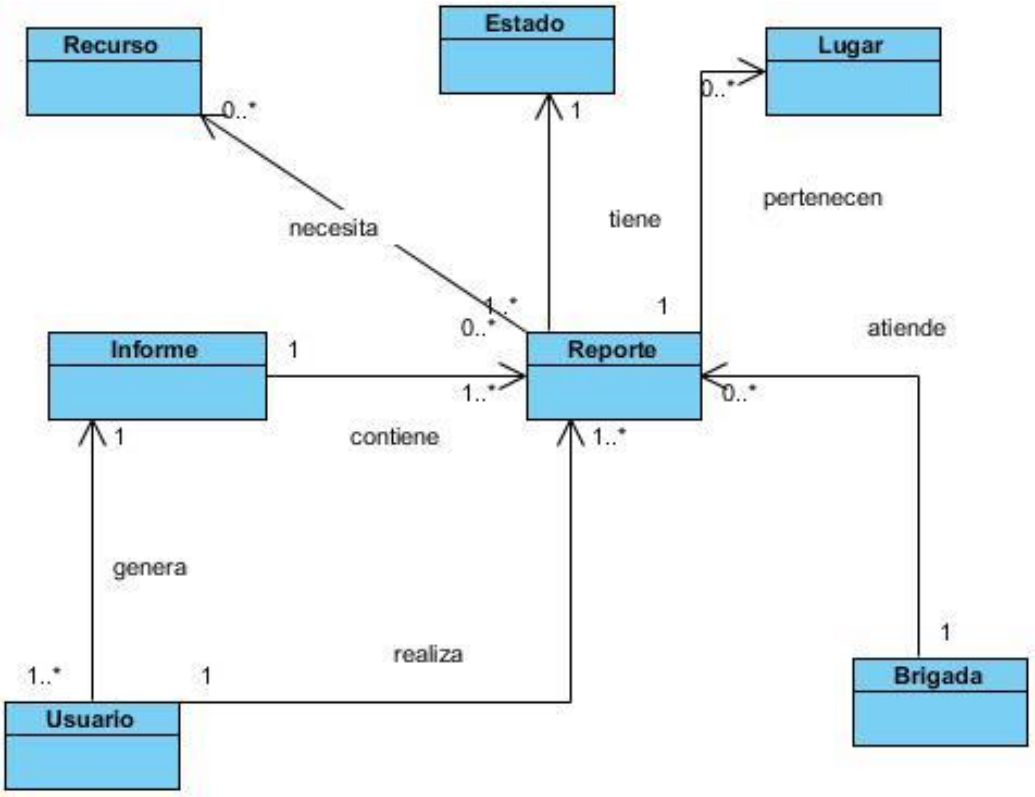
Luego debe definir para el reporte la fecha y hora de registro, asignar la brigada que debe atender el reporte y definir su prioridad. Luego de registrado el reporte y según las normas técnicas establecidas, los jefes de bridada deben establecer los recursos necesarios para atender el reporte de acuerdo a la disposición de dichos recursos. Con estos datos, se puede determinar si el reporte puede ser atendido o no, por lo que se le define un estado, que debe ser notificado al interesado en su resolución.

La dirección distribuye las brigadas por áreas específicas de la universidad, y asigna recursos a los reportes en dependencia de la prioridad de cada área. Las brigadas deben resolver un por ciento determinado de cada reporte de su área para que se considere que su trabajo ha sido satisfactorio, esto en dependencia de las causas por las cuales los reportes no han podido resolverse, que pueden estar asociadas a problemas externos o falta de material. Los jefes de brigada deben responder por la veracidad de la resolución de cada problema y de esa manera evaluar el desempeño de sus trabajadores.

Capítulo 2. Análisis y diseño del sistema

Semanalmente la dirección solicita un estado completo de los reportes, la situación de resolución por cada brigada, el resumen de consumo de recurso y las incidencias del período. Con esta información se emite un parte que refleja el quehacer de los servicios brindados por la Dirección de Mantenimiento, este informe es consultado por la dirección de la Universidad.

El proceso completo de gestión de reportes puede observarse a través del siguiente modelo de dominio. El modelo de dominio es utilizado para comprender el sector de negocios objeto de automatización. Puede utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema. Puede ser tomado como el punto de partida para el diseño del sistema, por lo que el mapa de conceptos del modelo de dominio constituye una primera versión del sistema.(Fowler, y otros, 2002).



Modelo de Dominio

Descripción de las entidades:

Usuario: Representan las personas que interactuaran con el sistema.

Reporte: Solitud de resolución de un problema de mantenimiento.

Informe: Resumen de reportes, quejas y equipos por rango de fecha. Es una ficha que se generará en el sistema con todos los reportes registrados en la fecha determinada

Estado: Define los estados que puede tener un reporte. Los estados serán configurados de acuerdo a las reglas del negocio.

Lugar: Área donde se encuentra la afectación contenida en el reporte. Se comportará en el sistema como una taxonomía en forma de árbol con las ubicaciones de las áreas de la universidad.

Brigada: Son las que ejecutan las acciones a partir del reporte registrado.

Recurso: Medio material necesario para resolver el reporte.

2.1.3 Sistema de Gestión de Reportes v.1

En el 2013 se realizó la primera versión del Sistema de Gestión de Reportes para la Dirección de Mantenimiento. Este sistema se enfocó en las siguientes funcionalidades:

- Gestionar roles y usuarios.
- Emitir reportes.
- Gestionar reportes
- Emitir informes.

En el caso de la gestión de reportes. El sistema permitía que cualquier usuario UCI emitiera reportes desde cualquier área. Esto comprometía la integridad de la información ya que los técnicos no tenían control sobre los reportes que se generaban. Otro de los problemas era durante la asignación de los reportes a las brigadas, ya que no se tenía en cuenta la prioridad de los reportes y el orden de resolución era disperso. El sistema, permitía asociar solamente un recurso a un reporte, y la disposición de este recurso según el sistema de control de activos y materiales (Assets) no era tenido en cuenta.

Por tanto, se hace necesario proponer una nueva versión de este sistema aprovechando sus funcionalidades base, para realizar el proceso de gestión de reportes de acuerdo al procedimiento que se utiliza en la Dirección de Mantenimiento.

2.1.4 Propuesta de solución.

Se propone una nueva versión de Sistema de Gestión de Reportes desarrollado para la Dirección de Mantenimiento. Este sistema se centrará en la asignación de recursos a los reportes de acuerdo a la disposición de los almacenes, la asignación, seguimiento y evaluación de brigadas y los informes de estado de resolución que se emiten a la Dirección de la Universidad.

Asignación de reportes de acuerdo a la disposición de recursos.

La asignación de reportes el sistema debe hacerla directamente a las brigadas responsables en dependencia del área para la cual se emite el reporte. Se debe tener en cuenta en primer lugar la prioridad asignada por el técnico y en segundo lugar la prioridad definida para el área. Durante la asignación de reportes el sistema permitirá al jefe de brigada seleccionar los recursos y la cantidad necesaria para atender el reporte y el sistema debe notificar si éste puede ser atendido según estos parámetros. La cantidad de recursos debe ser obtenida mediante un proceso de sincronización[4] con los estados del Assets.

Evaluación, seguimiento y evaluación de brigadas.

El sistema permitirá que los reportes le lleguen directamente a los jefes de brigada y a partir de ahí, estos podrán definir los estados de los reportes, los cuales tendrán un historial de cambios que puede ser consultado según los permisos de cada usuario. Durante las evaluaciones se podrá visualizar el por ciento de resolución según los estados definidos para los reportes.

Informes de estado.

Los informes de estado serán herramientas gráficas de barras y pastel, con el objetivo de mostrar siempre la distribución porcentual de los estados y la cantidad de reportes atendidos por cada brigada.

2.1.5 Requisitos de software

Son un punto clave en el desarrollo de las aplicaciones informáticas. Un gran número de proyectos de software fracasan debido a una mala definición, especificación o administración de requisitos. Factores

[4] Sincronización de datos: La sincronización de datos es el proceso del establecimiento de consistencia entre datos en fuentes remotas y de la continua armonización de los datos en el tiempo. Es la base para la sincronización de archivos. (ALEGSA, 2015).

tales como requisitos incompletos o mal manejo de los cambios de los requisitos llevan a proyectos completos al fracaso total (Jacobson, et al., 2000).

Requisitos funcionales

Los requisitos funcionales son las capacidades de resolución que el sistema debe tener. Los requisitos identificados para la nueva versión del Sistema de Gestión de Reportes se complementan con algunos requisitos planteados para su primera versión, como la gestión de roles y usuarios y la gestión de áreas.

RF 1. Autenticar usuario

RF 2.1 Adicionar brigada.

RF 2.2 Modificar brigada.

RF 2.3 Eliminar brigada.

RF 3. Sincronizar estado de recurso

RF 4.1 Adicionar reporte

RF 4.2 Modificar reporte.

RF 4.3 Eliminar reporte.

RF 5. Buscar reporte.

RF 6.1 Graficar cantidad de reportes por brigada.

RF 6.2 Graficar por ciento de reportes por estado.

RF 7. Actualizar estado de reporte.

Requisitos no funcionales

Requisitos de usabilidad: El sistema debe tener una navegación al lado derecho utilizando el formato de árboles. La información debe mostrarse de forma tabular en un área de trabajo. Deben mostrarse constantemente el acceso a los detalles de usuarios y durante la navegación entre las pestañas la información debe actualizarse automáticamente. Debe contener mensajes y ayudas que permitan su utilización por personas con conocimientos elementales sobre interfaces tabuladas y trabajo con árboles.

Requisitos de seguridad: El sistema establecerá una política de acceso basada en roles y usuarios, permisos de los usuarios para las operaciones y permisos funcionales sobre las operaciones. Debe garantizarse mediante este método una defensa en profundidad y asegurar la integridad, disponibilidad y confidencialidad de la siguiente forma:

- Integridad: La información generada debe ser consistente y protegida contra alteraciones de cualquier tipo. Durante la manipulación de informaciones de los procesos, éstas no pueden ser alteradas.
- Disponibilidad: El sistema deberá estar disponible durante todo el tiempo laboral, y permitir el acceso desde todas las áreas de la Universidad, para dar soporte a los procesos de decisión especializado y en grupo al personal autorizado para ello.
- Confidencialidad: La información debe responder a los permisos laborales de los usuarios que utilizan el sistema dejando constancia de las operaciones y funciones de dichos usuarios sobre los recursos y procesos que se manipulan.

Requisitos de interfaces externas: La interfaz debe tener colores serios, sobre las gamas frías. Los botones deben cumplir patrones estándar de formas e íconos. Los mensajes deben salir al centro o superior con colores que denoten su presencia. Las interfaces deben ser ajustables a los diferentes tipos de resoluciones de un ordenador estándar sin perjudicar la visualización de la información durante el reajuste de los navegadores.

Requisitos de rendimiento: La aplicación debe ejecutarse utilizando eficientemente los recursos de software y hardware tanto en el servidor como en el cliente, y además debe asegurarse que los tiempos de respuesta a las diferentes peticiones de los usuarios sean por debajo de los 1.5 segundos.

Requisitos de Software para el cliente:

- Navegador web con soporte JavaScript (Opera 10+, Mozilla Firefox20+, Internet Explorer 9+).

Requisitos de Software para el servidor

- Servidor Web Apache 2.2 o superior.
- Servidor de Base de Datos PostgreSQL 9.1 o superior.
- PHP versión 5.x

Requisitos de Hardware del servidor:

- Procesador Pentium IV/AMD, 2.4 GHz o superior.
- Memoria RAM: 1 Gb o superior.
- Disco duro: 80 Gb o superior.
- Tarjeta de Red o módem.

2.1.6 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema, documenta el comportamiento de un software desde el punto de vista del usuario. Por tanto los casos de uso determinan los requisitos funcionales del sistema, los cuales representan las funcionalidades que un sistema puede ejecutar(Addison Wesley object technology series, 2004).

A continuación se presenta el diagrama de casos de uso del sistema:

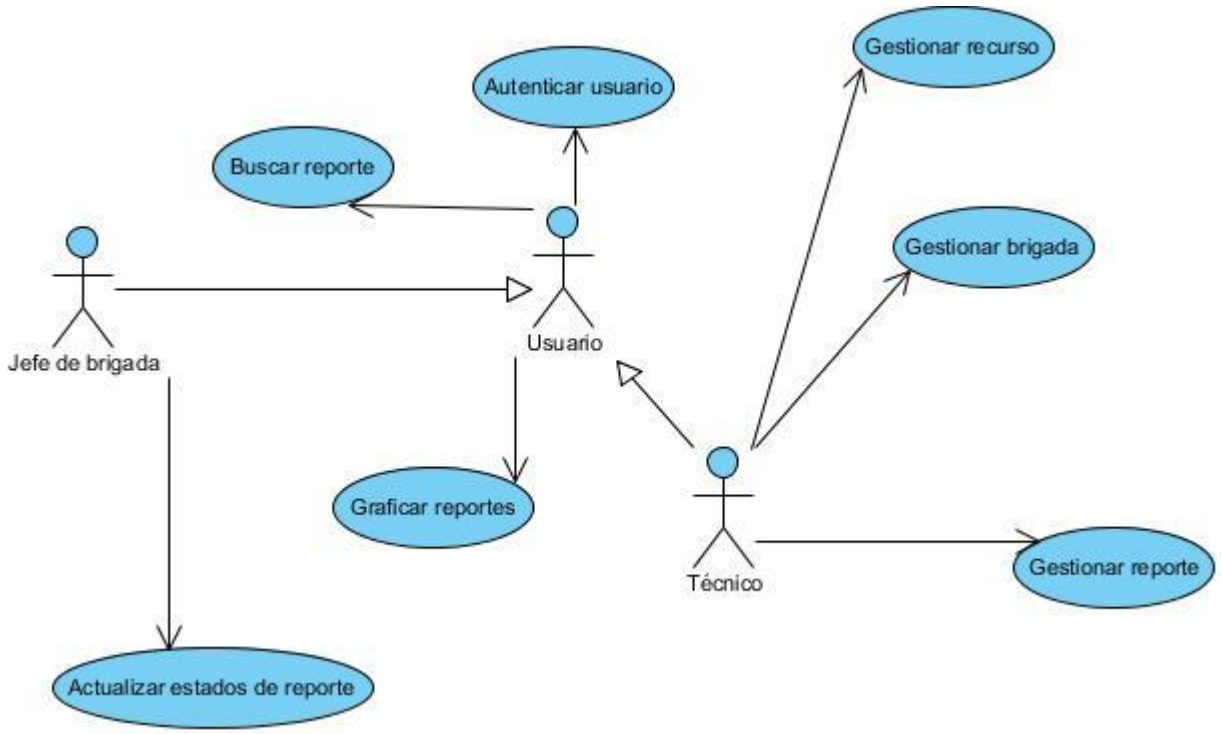


Fig. 2. Diagrama de casos de uso del sistema

2.1.7 Especificaciones de caso de uso

A continuación se especifican los casos de uso más relevantes del sistema.

Tabla 1. Especificación Caso de uso Autenticar usuario

Objetivo	Autenticar usuario
Actores	Usuario
Resumen	El caso de uso se inicia cuando un usuario accede al sistema y procede a autenticarse utilizando su usuario y contraseña. El usuario tiene la posibilidad de utilizar una contraseña local y su contraseña del dominio, siempre mediante el usuario del dominio UCI:
Complejidad	Media
Prioridad	Alta
Precondiciones	La cuenta de usuario debe estar habilitada como cuenta local por uno de los administradores del sistema
PostCondiciones	El usuario accede mediante su sección a la interfaz principal del sistema.

Actor	**Sistema**
1. El usuario accede al sistema	1.1 El sistema muestra el formulario de autenticación solicitándose usuario y contraseña.
2. El usuario introduce su usuario del dominio UCI, la contraseña y define si el acceso será local o mediante el dominio	2.1 El sistema verifica el tipo de acceso del usuario. 2.1.1 El sistema verifica para el usuario local si la contraseña coincide con la almacenada. 2.1.2 El sistema verifica para el acceso mediante dominio que la contraseña coincida con la contraseña obtenida para el usuario mediante el

	servicio LDAP. 2.2 El sistema muestra la interfaz de inicio.
Flujo Alternativo 2.1.1 y 2.1.2	
Actor	**Sistema**
	El sistema muestra un mensaje indicando "El usuario introducido, la contraseña o el tipo de acceso no es correcto"
Prototipo de interfaz	
	

Tabla 2. Especificación Caso de uso Gestionar reporte

Objetivo	Gestionar reporte
Actores	Técnico
Resumen	El caso se inicia cuando el técnico precisa gestionar un reporte generado partir de

	la solicitud de un usuario. El técnico puede adicionar un reporte, donde debe especificar la descripción del reporte, el lugar, el usuario de quien lo realiza el tipo y la prioridad. El usuario puede modificar dichos datos o eliminar el reporte siempre que su estado sea el inicial.
Complejidad	Alta
Prioridad	Alta
Precondiciones	El técnico debe estar autenticado y con permisos para gestionar reportes.
PostCondiciones	Se adiciona un nuevo reporte y queda asignado a una brigada, se modifica o elimina.

Actor	**Sistema**
1. El usuario accede mediante la funcionalidad Reportes del árbol de funcionalidades al ambiente de gestión de reportes. 2. El usuario selecciona la opción de adicionar, modificar o eliminar.	1.1 El sistema visualiza la lista de reportes.
Sección: Adicionar Reporte	
2. El usuario selecciona la opción adicionar reporte. 2.2 El usuario introduce la descripción, el lugar, el usuario que realiza el reporte, el tipo de reporte y la prioridad.	2.1 El sistema muestra un formulario solicitando los datos del reporte. 2.3 El sistema verifica si los datos están correctos. En caso de faltar algún dato se pasa al flujo alterno A, en caso de que el reporte ya exista se pasa al flujo alterno B. En caso del que usuario sea incorrecto pasa al flujo alterno C.

	2.4 El sistema asigna el reporte a la brigada a partir de área introducida por el usuario. 2.5 El sistema determina para el reporte la fecha, la hora, el usuario del técnico y el estado. 2.6Registra el nuevo reporte. 2.7 Muestra un mensaje indicando el éxito de la operación.
Flujo alternativo 2.3 A	
Actor	**Sistema**
	El sistema señala los campos vacíos o con error.
Flujo alternativo 2.3 B	
	Muestra el mensaje: “El reporte definido y se encuentra registrado”.
Flujo alternativo 2.3 C	
	Muestra el mensaje: “El usuario definido no existe”.
Sección Modificar Reporte	
2. El usuario selecciona la opción modificar reporte. 2.2 El usuario puede modificar la descripción, el lugar, el usuario que realiza el reporte, el tipo de reporte y la prioridad.	2.1 El sistema verifica que se haya seleccionado un reporte. 2.3 El sistema muestra un formulario con los datos del reporte. 2.4 El sistema verifica si los datos están correctos. En caso de faltar algún dato se pasa al flujo

	alterno A, en caso del que usuario sea incorrecto pasa al flujo alterno B 2.6 El sistema determina para el reporte la fecha, la hora, el usuario del técnico y el estado modificando los datos anteriores. 2.4 Modifica el reporte seleccionado. 2.5 Muestra un mensaje indicando el éxito de la operación.
Flujo alternativo 2.1	
	El sistema muestra un mensaje “Debe seleccionar un reporte”
Flujo alternativo 2.4 A	
	El sistema señala los campos vacíos o con error
Flujo alternativo 2.4 B	
	Muestra el mensaje: “El usuario definido no existe”.
Sección eliminar reporte	
2. El usuario selecciona la opción eliminar reporte.	2.1 El sistema verifica que se haya seleccionado un reporte. 2.2El sistema verifica que el reporte esté en estado inicial. 2.3El reporte queda eliminado. 2. 4 El sistema muestra un mensaje de operación

	satisfactoria
Flujo alternativo 2.1	
	El sistema muestra un mensaje “Debe seleccionar un reporte”
Flujo alternativo 2.2	
	El sistema muestra el mensaje: “El reporte no puede ser eliminado porque ya ha sido asignado a la brigada”
Prototipo de interfaz	

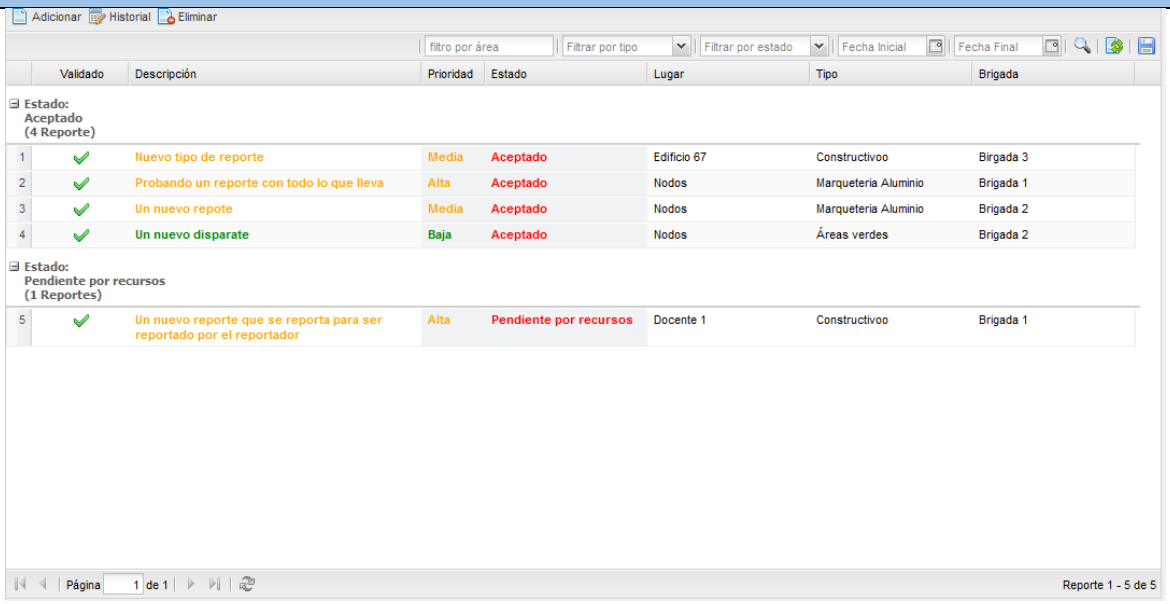


Tabla 3. Especificación del caso de uso Gestionar recurso

Objetivo	Gestionar recurso
Actores	Técnico

Resumen	Se inicia cuando el técnico accede a la interfaz de gestión de recursos y procede a la sincronización de éstos con el Assets, accediendo al archivo de recursos generados por el Assets y actualizando los nomencladores y cantidades del sistema.
Complejidad	Alta
Prioridad	Alta
Precondiciones	El usuario debe tener permisos habilitados para gestionar los recursos
PostCondiciones	Los recursos quedan sincronizados según los datos del Assets.

Actor	**Sistema**
1. El usuario accede a la interfaz de gestión de recursos.	1.1 El sistema muestra una lista con todos los recursos disponibles para su uso.
2. El usuario selecciona en la barra superior el botón “Sincronizar recursos” 2.2 El usuario selecciona “Si”	2.1 El sistema muestra un mensaje de confirmación “¿Está seguro que desea sincronizar los recursos? Todos los datos serán actualizados” En caso de seleccionar “No” concluye el caso de uso. 2.3 El sistema carga el archivo Excel generado por el Assets con el listado de recursos y su cantidad. 2.4 El sistema actualiza el registro de recursos. 2.5 El sistema muestra un mensaje de operación exitosa

Flujo alternativo 1.1	
Actor	**Sistema**
	El sistema muestra el mensaje: “No existen recursos definidos”
Flujo alternativo 2..4	
	El sistema muestra el mensaje: “Ha ocurrido un error al actualizar los datos de los recursos.”
Prototipo de interfaz	

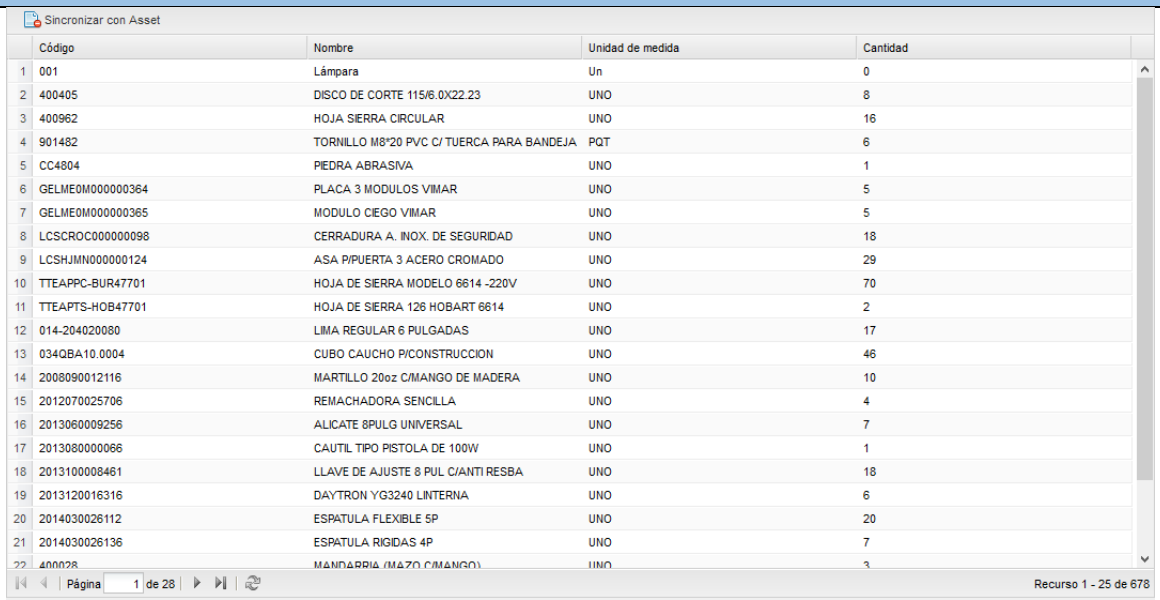


Tabla 4. Especificación caso de uso Actualizar estado de reporte

Objetivo	Actualizar estado de reporte
Actores	Jefe de brigada
Resumen	El caso de uso se inicia cuando el jefe de brigada procede actualizar el estado de

	los reportes
Complejidad	Media
Prioridad	Media
Precondiciones	El usuario debe tener permisos habilitados para actualizar los estados de los reportes.
PostCondiciones	El estado de reportes se modifica y se adiciona un registro al historial de cambios del reporte.

Actor	**Sistema**
1. El usuario acceder a la interfaz de gestión de reportes.	1.1 El sistema muestra una lista con todos los reportes pertenecientes a la brigada.
2. El usuario selecciona en la barra superior el botón “Actualizar estado de reporte” 2.3 El usuario selecciona “Si” 2.5 El usuario llena los campos.	2.1 El sistema verifica que se haya seleccionado un reporte. 2.2 El sistema muestra un mensaje de confirmación “¿Está seguro que desea actualizar el estado del reporte?” En caso de seleccionar “No” concluye el caso de uso. 2.4El sistema muestra un formulario donde permite escoger el nuevo estado de reporte y agregar una nota de no más de 250 caracteres. 2.6 El sistema actualiza el registro seleccionado. 2.5 El sistema muestra un mensaje de operación exitosa

Flujo alternativo 2.1	
Actor	**Sistema**
	El sistema muestra el mensaje: “Debe seleccionar un reporte”
Flujo alternativo 2.6	
	El sistema muestra el mensaje: “Ha ocurrido un error al actualizar los datos del reporte.”

2.2 Diseño

2.2.1. Arquitectura

La arquitectura a nivel de aplicación se basa en el patrón Modelo- Vista-Controlador. El principio más importante de la arquitectura MVC es la separación del código entrés capas, dependiendo de su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador. La programación se puede simplificar si se utilizan otros patrones de diseño. De esta forma, las capas del modelo, la vista y el controlador se pueden subdividir en más capas (Potencier, et al., 2008).

En la solución la aplicación de este patrón de forma general se observa de la siguiente forma:

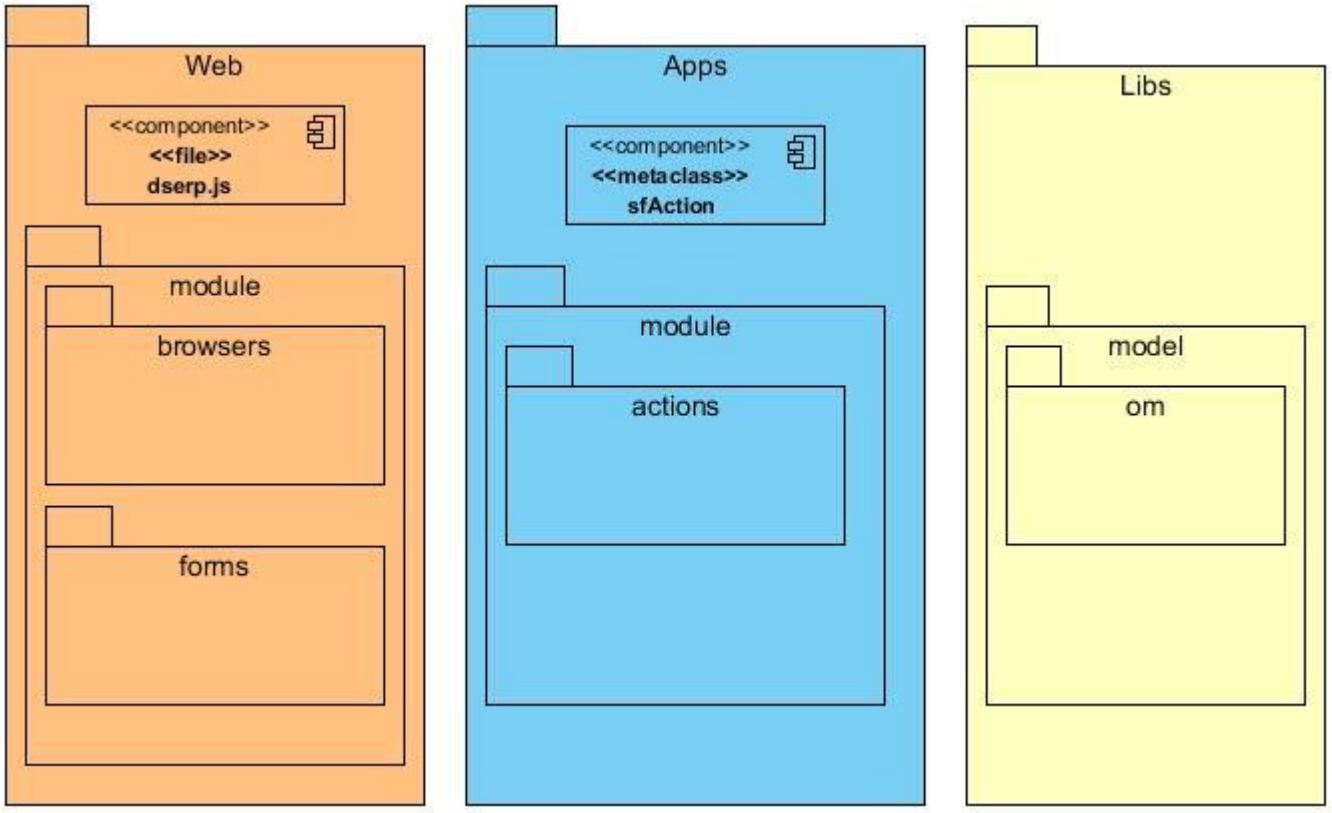


Fig. 3. Diagrama de paquetes Sistema de Gestión de Reportes

El diagrama representa la distribución de los paquetes de la solución aplicando el patrón MCV.

- Vistas: Las vistas están desarrolladas con JavaScript, utilizando el *framework* EXTJs. De esta forma la aplicación es más ligera ya que ejecuta dicho código del lado del cliente. En la vista se tiene el archivo dserp.js que contiene todas las configuraciones globales y actúa como instancia única, permitiendo la herencia de sus configuraciones hacia los archivos del paquete módulo, donde se definen las vistas propias de la aplicación. Dentro de cada módulo existe un paquete browsers, donde se guardan los archivos que permiten listar datos, y un paquete forms donde se definen los formularios.
- Controlador: Las clases controladoras se definen dentro del paquete apps de la aplicación. Todas las clases controladoras se implementan dentro del paquete actions de cada módulo. Estas clases heredan del controlador frontal de symfony sfAction, que actúa como instancia única para procesar las peticiones de los usuarios y manejar los recursos de Symfony.

- Modelo: El modelo está descrito en el paquete libs, donde se crean las interfaces de las clases persistentes, que se encuentran en el directorio model, a partir del Mapeo de Objetos Relacional que realiza Propel dentro del directorio OM.

Estilo arquitectónico

El estilo Cliente-Servidor es el que define una relación entre dos aplicaciones en las cuales una de ellas (cliente) envía peticiones a otra (servidor fuente de datos), para su procesamiento. Es un estilo para sistemas distribuidos que divide el sistema en una aplicación cliente, una aplicación servidora y una red que las conecte, en la cual la primera realiza peticiones y la segunda emite respuesta usando un amplio rango de protocolos y formatos de datos para comunicar información.

2.2.1 Patrón de diseño

Un patrón de diseño es una buena práctica que determina en gran medida la calidad de las aplicaciones de software. Es una solución documentada, que se ha aplicado con éxito en múltiples ambientes para erradicar problemas comunes de diseño de software, con una probada efectividad y con características de reutilización (Fowler, y otros, 2002). El uso del framework *Symfony* como escenario de desarrollo, por las características que tiene, brinda una gama de patrones que permiten garantizar que el software que se desarrolla tenga una línea arquitectónica establecida, robusta e integra, y que pueda establecer una correcta comunicación con los elementos que puedan interactuar internamente. En la aplicación desarrollada se evidencias los siguientes patrones.

Patrones GOF

Instancia Única: Este patrón se implementa mediante la clase sfRouting una variable global definida en el sistema. El sfRouting contiene el método getInstance () que ejecuta todas las peticiones que se realizan a la aplicación a través del controlador sfWebFrontController. Mientras que la variable global dserp, se comporta como instancia única del sistema que se encarga de construir todas las clases, las funciones y acciones dentro de las vistas generadas por ExtJS.

Comando: Este patrón se manifiesta a través de la clase sfWebController, encargada de establecer el módulo y la acción que se usará por cada petición del usuario.

Decorador: Utilizado en la clase sfView, padre de todas las vistas y que permite agregar funcionalidades dinámicamente.

Registro: Este patrón se aplica en la clase sfConfig, que es la encargada de acumular todas las variables de uso global en el sistema, es una manera fácil y rápida de compartir datos y objetos en la aplicación, permitiendo que no sea necesario conservar muchos parámetros y reduce el uso de variables globales.

Patrones GRASP

Experto: Este patrón se introduce mediante el uso de la librería Propel para mapear la Base de Datos. Esta librería, es usada en la capa del modelo, maneja toda la lógica de los datos, generando clases a partir del modelo racional con todas las funcionalidades necesarias para el desarrollo orientado a objetos.

Creador: En la clase Actions, se evidencia dicho patrón, en la misma se encuentran se las acciones que crean los objetos de las clases que representan las entidades.

Alta cohesión: El uso del framework Symfony provee mediante una alta cohesión para la asignación de responsabilidades y el empaquetamiento de archivos.

Bajo acoplamiento: Este patrón se evidencia durante la creación de objetos de las entidades con la clase Actions, que hereda únicamente del controlador sfActions, alcanzando un bajo acoplamiento de las clases. El modelo de tres capas además abstrae la vista y el controlador del modelo, proporcionado una baja dependencia.

Controlador: Todas las peticiones desde la web son manipuladas por un solo controlador: sfActions, que es el punto de entrada único para la aplicación dentro de un determinado entorno. Este patrón se evidencia en las clases sfFrontController, sfWebFrontController y sfContext.

2.2.2 Diagrama de clases de diseño

El Diagrama de Clases es el diagrama principal de diseño y análisis para un sistema. En él, la estructura de clases del sistema se especifica, con relaciones entre clases y estructuras de herencia. Durante el análisis del sistema, el diagrama se desarrolla buscando una solución ideal. Durante el diseño, se usa el mismo diagrama, y se modifica para satisfacer los detalles de las implementaciones. Describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Además de los componentes que se encargarán del funcionamiento y la relación entre uno y otro(Fowler, 2007).

A continuación se muestran los diagramas de clase del análisis para los principales casos de uso del sistema.

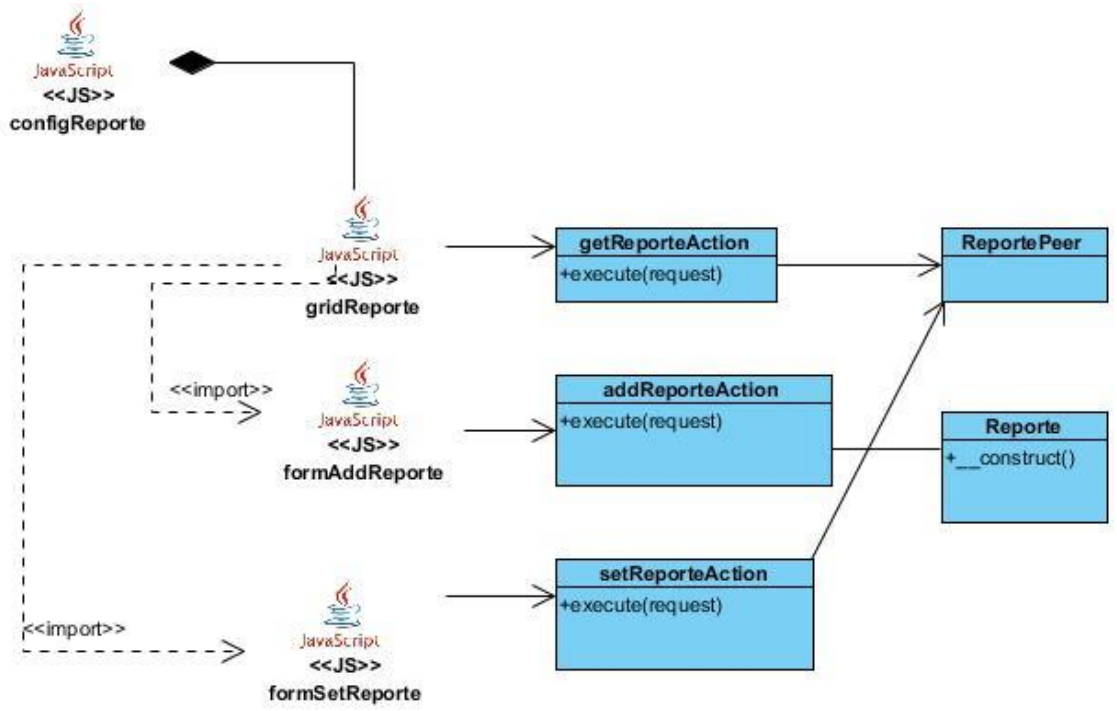


Fig. 4. Diagrama de clases del caso de uso Gestionar reportes.

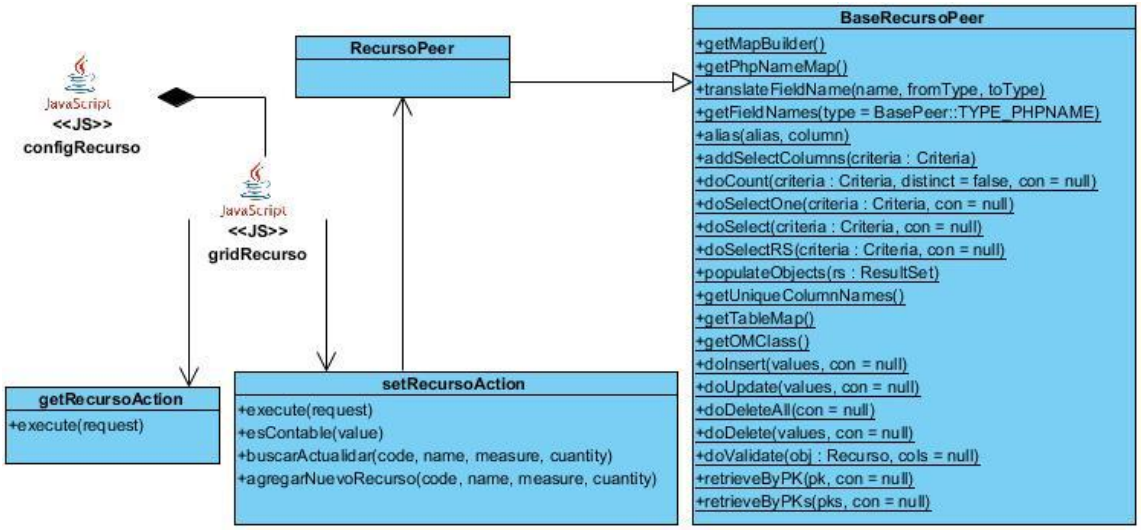


Fig. 5. Diagrama de clases Caso de uso Gestionar Recursos

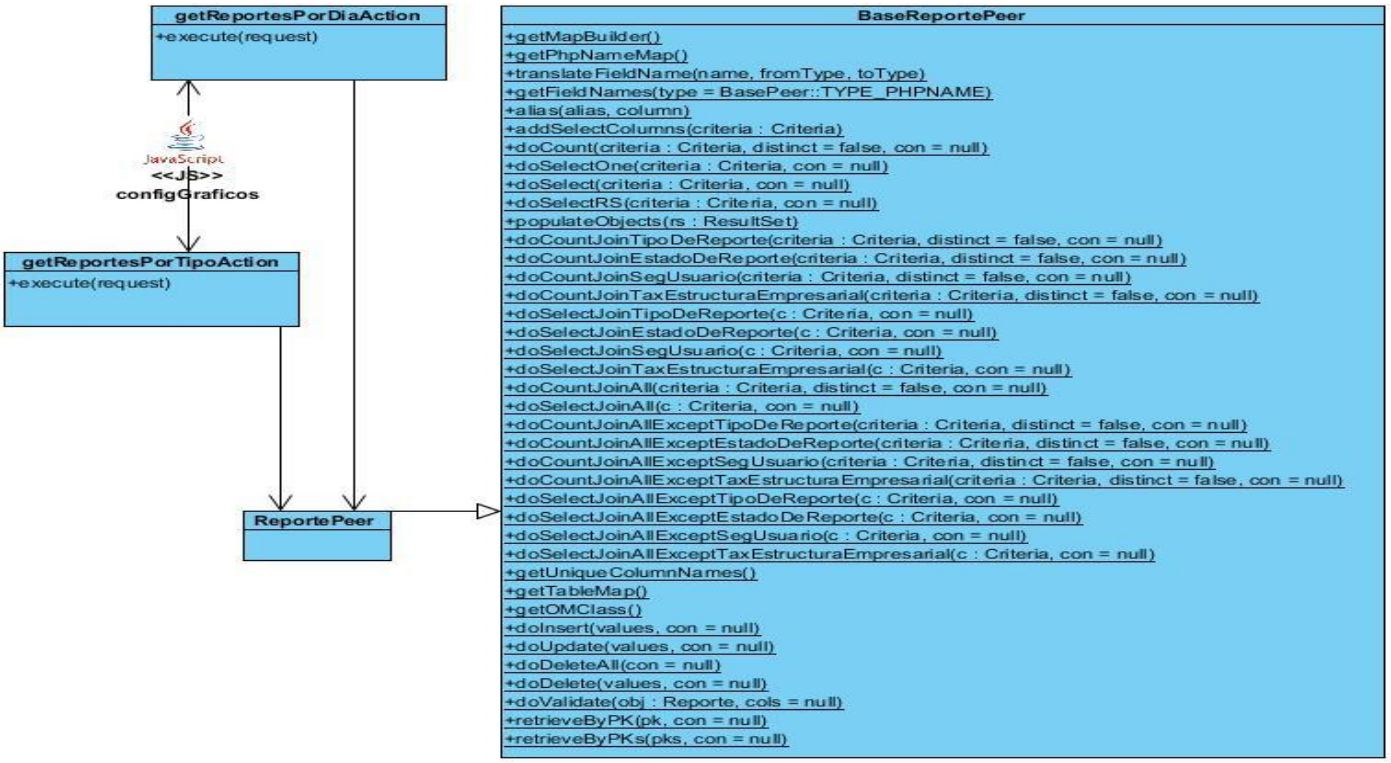


Fig. 6. Diagrama de Clases Caso de uso Graficar Reporte

2.2.3 Diagrama de interacción

Los diagramas de interacción modelan el comportamiento dinámico del sistema y el flujo central en una operación. Describe la interacción entre objetos, los cuales interactúan a través de mensajes para cumplir ciertas tareas. Existen dos tipos de diagramas de interacción: secuencia y colaboración(Fowler, 2007).

En el caso de la solución propuesta se determinaron diagramas de secuencia. Los diagramas de secuencia para los principales casos de uso pueden observarse a continuación.

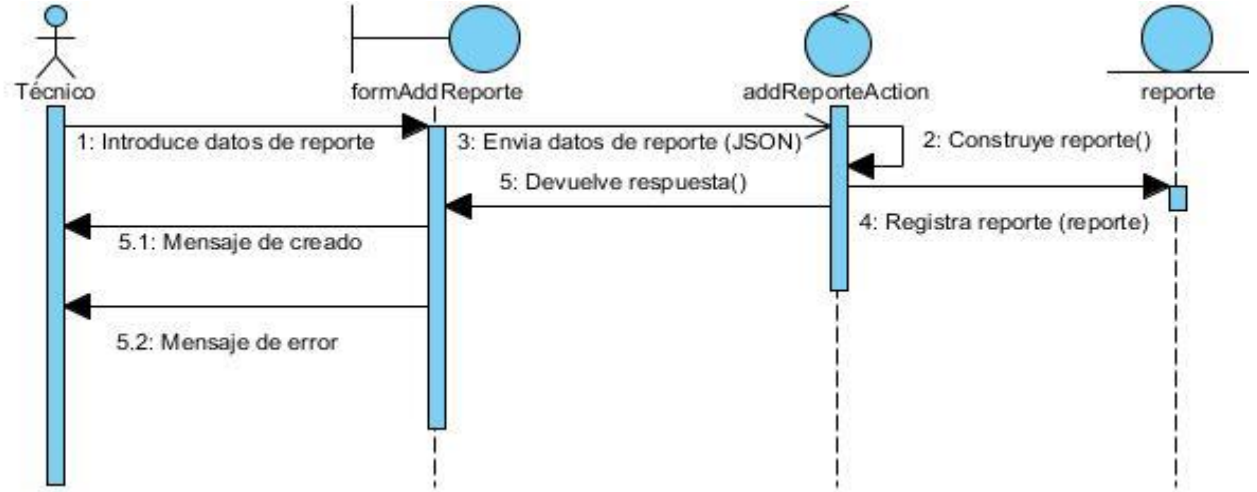


Fig. 7. Diagrama de secuencia Caso de Uso Gestionar Reporte. Escenario Adicionar Reporte

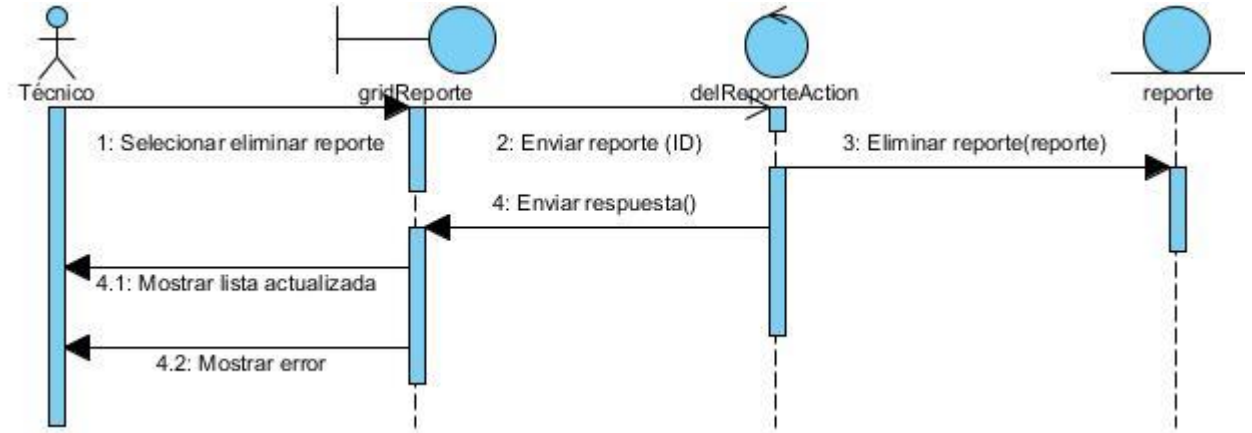


Fig. 8. Diagrama de secuencia Caso de Uso Gestionar Reporte. Escenario Eliminar Reporte

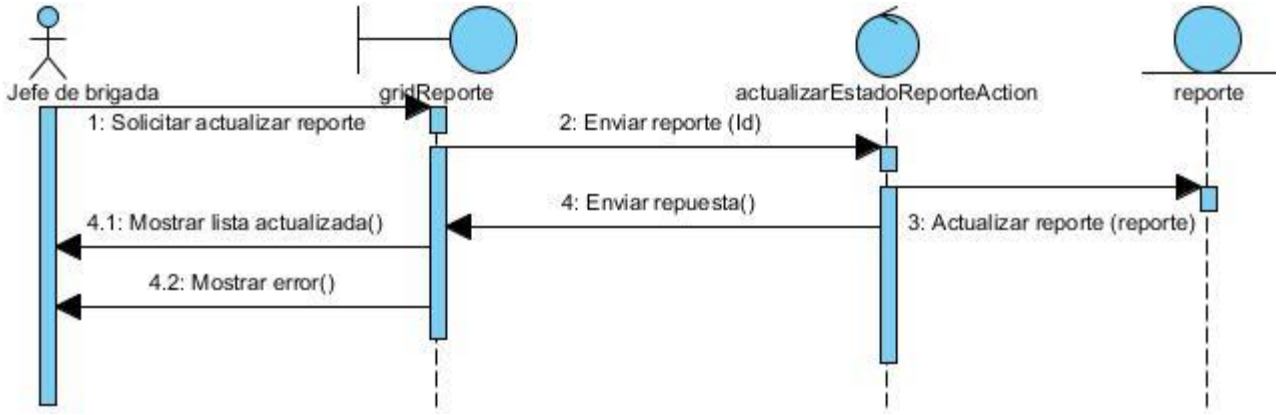


Fig. 9. Diagrama de secuencia Caso de Uso Actualizar estado de reporte.

2.2.4 Modelo de datos

El modelo de datos está descrito por el diagrama entidad-relación, que denota el modo físico en que está dispuesta la información dentro de la base de datos.

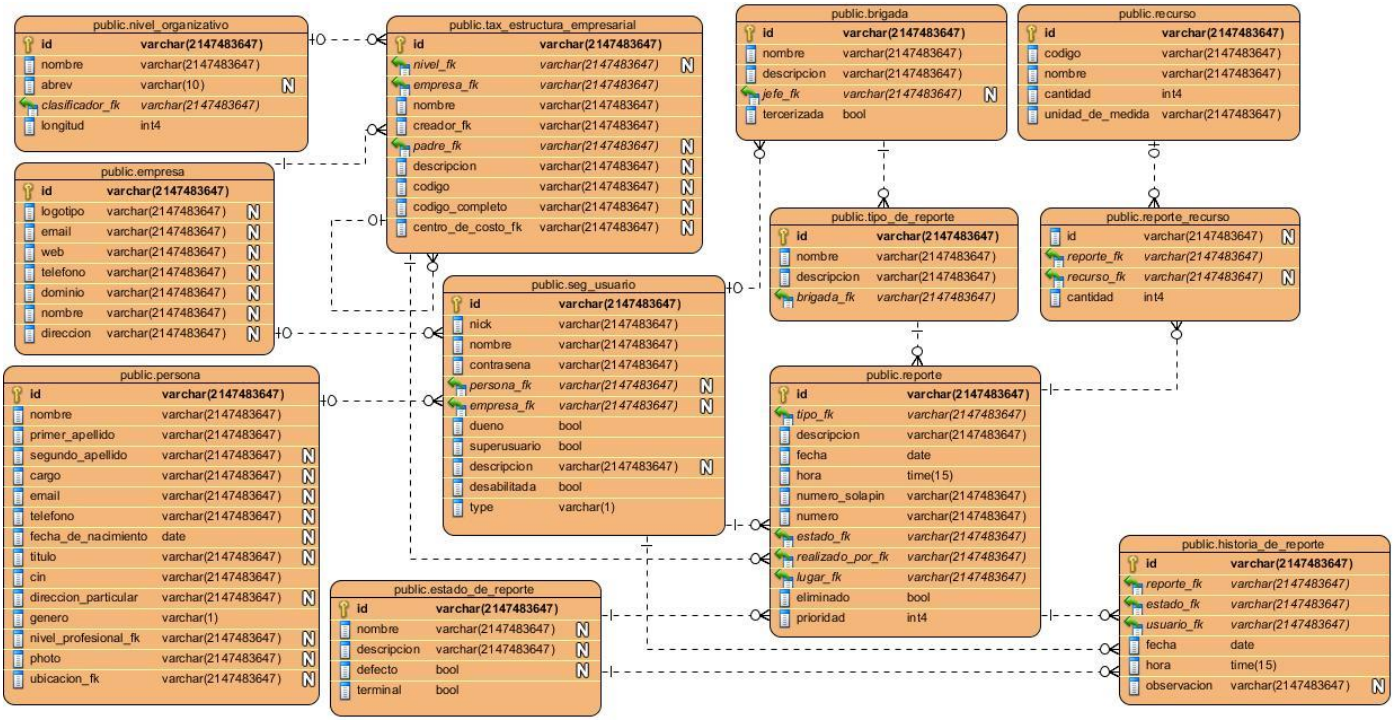


Fig. 10. Diagrama Entidad-Relación

La descripción de las entidades es la siguiente:

Tabla 5. Descripción de entidades de la base de datos

Entidad	Descripción
nivel_organizativo	Establece los distintos niveles que puede tener un área. Un nivel es un tipo de área según su propósito para la Universidad. Puede ser de nivel docente, residencia, rectorado u otro que desee definir el usuario. Su estructura y nomenclatura es propuesta por el modelo EBMS DSerp

tax_estructura_empresarial	Establece una taxonomía para el control de las áreas, lo que permite identificar la relación jerárquica entre ellas. Su estructura y nomenclatura es propuesta por el modelo EBMS DSerp
Empresa	Entidad que registra las distintas empresas a las cuales se destina el sistema, permite establecer una solución a nivel de dominio con varias direcciones utilizando el sistema. Su estructura y nomenclatura es propuesta por el modelo EBMS DSerp.
Persona	Entidad que registra los datos generales de cualquier persona, usuarios, empleados, terceros, entre otros. Su estructura y nomenclatura es propuesta por el modelo EBMS DSerp
Seg_usuario	Es la entidad principal del modelo de seguridad del EBMS DSerp. Interactúa directamente con demás entidades identificadas para el negocio y permite el registro de todos los usuarios que acceden a la aplicación.
Brigada	Registra los datos de las brigadas pertenecientes la Dirección de Mantenimiento.
Tipo_de_reporte.	Se establece para tener constancia del tipo de reporte que realizan los usuarios, lo que permite la clasificación de éstos y el análisis estadístico.
Reporte	Entidad principal que recoge los datos de los reportes.
Estado_de_reporte	Permite definir los distintos tipos de estado en que se puede encontrar un reporte. Los estados también sirven para determinar los estados por defecto y los estados terminales.
Recurso	Entidad que registra los recursos disponibles y su cantidad a partir de la sincronización con el Assets.
Reporte_recurso	Dada la relación múltiple entre recursos y reporte, esta entidad

	permite relacionar ambos conceptos siguiendo los principios de normalización de bases de datos.
Historial_reporte.	Permite definir los diferentes cambios que tiene un reporte, lo que puede ser analizado durante las evaluaciones y representaciones gráficas de los estados dentro de la Dirección de Mantenimiento.

Para una mejor compresión del ambiente donde se desplegará la solución se propone el siguiente diagrama de despliegue.

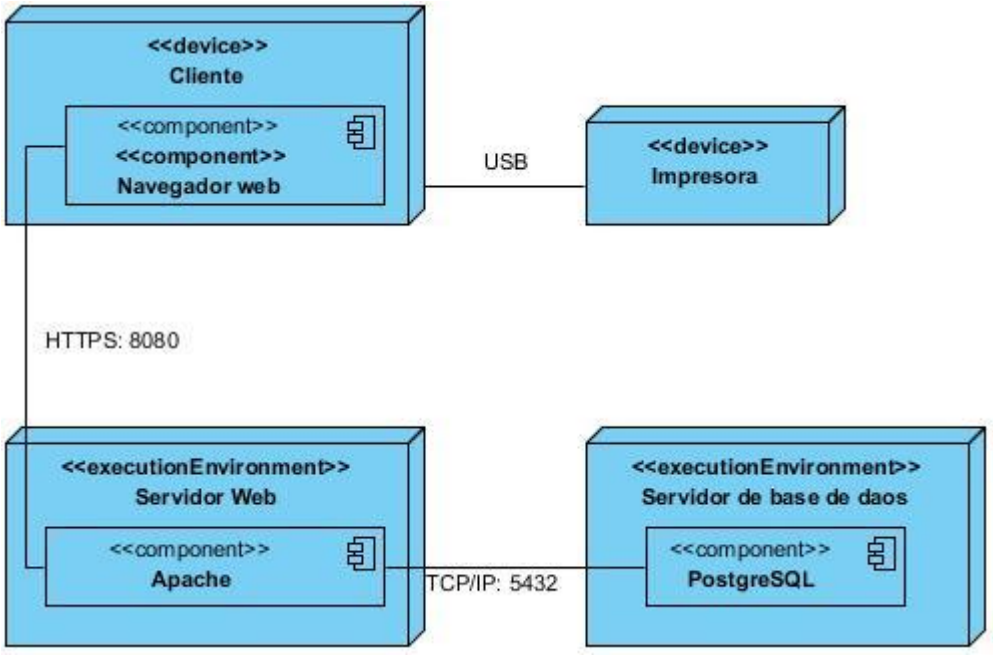


Fig. 11. Diagrama de despliegue

2.2.5 Conclusiones parciales

En el capítulo se describió de forma detallada los aspectos fundamentales del negocio a partir de un modelo de dominio. Se presentó la propuesta de solución indicando los elementos que la conformarán, describiéndose éstos mediante los artefactos del análisis y diseño. Se evidenciaron aspectos del diseño

de la solución propuesta centrándose en el diseño lógico, quedando definida la arquitectura a utilizar Modelo-Vista-Controlador por las ventajas que ofrece y la relación con las tecnologías empleadas. Se mostraron y describieron los principales patrones de diseño y arquitectónicos que rigen el desarrollo. Se presentó una vista de cómo quedaría conformado el sistema tanto a nivel de la aplicación a través del diagrama de clases, y el modelo de la base de datos que respaldará al sistema en ambos casos se realizó una descripción de los elementos más importantes. Los artefactos mostrados contribuyeron a representar lógicamente el sistema propuesto y conocer sus principales características según la arquitectura definida.

Capítulo 3. Implementación y pruebas

Introducción

En el siguiente capítulo se abordará la etapa de implementación a partir de la definición del estándar utilizado y detallando el sistema mediante en modelo de implementación. Además, se realizará la validación del sistema a partir de las pruebas de software y se argumentarán los resultados del despliegue experimental de la solución propuesta.

3.1 Modelo de implementación

Un modelo de implementación es un conjunto de artefactos que permiten documentar la construcción de un software en su etapa de codificación, a partir de la colección de componentes y subsistemas de implementación que lo conforman. Describe tanto los elementos del modelo de diseño, como las clases y otros artefactos que se comportan como compontes dentro del sistema, estos pueden incluir ejecutables, ficheros de código fuente, script y otros elementos necesarios para la implementación y el despliegue.

3.1.1 Estándar de codificación

Para la codificación del sistema se utilizó el estándar propuesto por el equipo de desarrollo del modelo DSerp. A continuación se detallan las principales características del estándar.

Estándar de base de datos

1. Todas las entidades tienen un prefijo que define el esquema al que pertenece, menos las entidades públicas. Ejemplo, la entidad que define los usuarios se llamará seg_usuario ya que pertenece al esquema de seguridad.
2. Todas las llaves foráneas tendrán el sufijo _fk.
3. Todas las relaciones tendrán sus identificadores generados a partir de la función replica_id.
4. Los nombres de las entidades y sus atributos se escriben en minúscula.
5. Todas las entidades y los atributos están comentadas.

Estándar de desarrollo

1. El estilo de codificación será lowCamelCase, se utilizará para la definición de archivos, Clases, métodos, atributos y servicios.
2. Las clases tendrán la nomenclatura nombreClaseAction.class.php

3. El estilo será el definido por defecto por el IDE Netbeans.
4. Todos los formularios de las vistas tendrán el sufijo form seguido del propósito del formulario. Ejemplo. formAddRecurso, formSetTipoDeRecurso.
5. Todas las tablas tendrán el sufijo grid. Ejemplo: gridRecurso.
6. Todas las clases controladoras tendrán como sufijo la acción que ejecuta. Ejemplo: addRecursoAction.class.php.
7. Cada clase tendrá en su parte superior un texto de licencia y los datos del autor, fecha de programación y propósito.

3.2 Pruebas de software

Las pruebas son actividades que se realizan para garantizar la calidad de las aplicaciones de software a partir del testeo de la misma o sus componentes. Las pruebas representan la revisión final de las especificaciones del diseño y la codificación. El resultado de las pruebas determina si el software cumplió no los requisitos establecidos durante su definición (Addison Wesley object technology series, 2004).

Una prueba debe comprobar la integración de los elementos del sistema, verificar el funcionamiento completo de las interfaces así como la comunicación entre los subsistemas. Por tanto, para garantizar que las pruebas sean exitosas es necesario dividirlas por niveles.

Niveles de prueba

Los niveles de prueba permiten definir pruebas de software desde los elementos de cada módulo hasta las cualidades del sistema como un todo. Para el sistema se realizaron las pruebas en diferentes niveles como se detalla a continuación.

Pruebas unitarias: Las pruebas unitarias prueban el funcionamiento correcto de cada módulo de código, su objetivo es asegurar que dicho módulo funciona por separado, aislando cada parte del programa y demostrando que sus individualmente se cumple con los propósitos para los cuales se programó el módulo. Para cada módulo de código se realizaron pruebas unitarias utilizando las funcionalidades de depuración del *framework* Symfony, que permitió a partir del controlador frontal testear el funcionamiento de cada clase controladora y su interacción con las clases del modelo. Una vez concluida las pruebas unitarias a cada clase controladora se le añadió una verificación del método de entrada para evitar el acceso indebido a estás utilizando el controlador frontal.

Pruebas del sistema: Son pruebas que verifican el funcionamiento del sistema como un todo. Una vez integrados los componentes debe comprobarse el cumplimiento de los requisitos planteados. La comprobación a este nivel permite verificar los atributos de calidad que determinan si el sistema cumple o no los objetivos planteados. Para realizar pruebas al sistema se decidió utilizar el método de la caja negra. Este método permite comprobar el funcionamiento de un sistema a partir del resultado de cada una de las operaciones que debe cumplir, documentando los resultados sin interesarse en la complejidad de desarrollo de cada operación; a diferencia de los métodos de caja blanca, que si valida que la complejidad y el funcionamiento interno de cada componente cumpla con los requisitos de calidad establecidos para el desarrollo del software.

3.2.1 Casos de prueba

Un caso de prueba es una forma específica de probar un sistema, testeando las entradas y resultados con las que debe operar el sistema en las condiciones donde debe probarse. Los casos de prueba son artefactos útiles para realizar pruebas funcionales(Jacobson, y otros, 2000).

Los casos de prueba son la documentación de una prueba en la mayoría de los casos, de Caja Negra, donde se prueba un componente, módulo, acción o propiedad según el resultado que se espera, sin detenerse en su funcionamiento interno. Para probar la solución se definieron pruebas de funcionalidad, por tanto, los casos de prueba representan pruebas de Caja Negra de Funcionalidad.

- **Caso de prueba para el Caso de uso Autenticar usuario.**

Tabla 6. Variables del caso de prueba Autenticar usuario

No.	Nombre	Descripción	Tipo	Nulo
1	Usuario	Nombre del usuario que se autentica	Texto	No
2	Contraseña:	Contraseña del usuario	Texto	No
3	Dominio	Define si el usuario accede mediante una cuenta local o una cuenta UCI	Lista desplegable	No

Tabla 7. Matriz del caso de prueba Autenticar usuario

Escenario	Combinación	Respuesta esperada	Resultado
Introducir correctamente usuario y contraseña	1-1-1	El sistema debe validar los datos y dar acceso al usuario según los permisos de este.	Satisfactorio
Introducir datos de usuario y contraseña incorrectos	1-1-1	El sistema valida los datos y notifica al usuario que la combinación usuario contraseña es incorrecta	Satisfactorio
Dejar en blanco campo o usuario o contraseña	0-1-1 o 1-0-1	El sistema muestra el campo vacío en rojo con un mensaje al usuario de: “el campo no puede estar vacío”	Satisfactorio

- **Caso de prueba del caso de uso: Gestionar reporte.**

Tabla 8. Variables del caso de prueba Gestionar Reporte

No.	Nombre	Descripción	Tipo	Nulo
1	Descripción	Describe el hecho que origina el reporte.	Texto	No
2	Lugar	Lugar o área de la universidad desde donde se realiza el reporte	Auto completable	No
3	Usuario	Define el nombre de usuario uci que realiza el reporte	Texto	No
4	Tipo	Define el tipo de reporte según la clasificación especificada por la dirección de mantenimiento.	Lista desplegable	No
5	Prioridad	Define la prioridad del reporte según la matriz de prioridades de la dirección de mantenimiento.	Lista desplegable	No

Tabla 9. Matriz del caso de prueba Gestionar reporte

Escenario	Combinación	Respuesta esperada	Resultado
Introducir correctamente datos del reporte.	1-1-1-1-1	El sistema asigna al reporte la fecha y la hora, le asigna la brigada y registra el técnico que define el reporte y muestra una notificación de “Reporte guardado satisfactoriamente”	Satisfactorio
La descripción del reporte está vacía	0-1-1-1-1	El sistema señala el campo de descripción con un color rojo mostrando un mensaje: “La descripción del reporte no puede estar vacía”	Satisfactorio
En el campo de lugar se introduce un valor que no corresponde a ningún lugar de la universidad.	1-0-1-1-1	El sistema muestra un mensaje de : “Le lugar definido no aparece definido como área de la Universidad”	Satisfactorio
No se selecciona la prioridad o el tipo de reporte	1-1-1-0-1 o 1-1-1-1-0	El sistema señala con color rojo en campo vacío y muestra el mensaje: “El campo no puede estar vacío”	Satisfactorio
El tipo de reporte no está asignado a ninguna brigada	1-1-1-1-1	El sistema muestra un mensaje de: “el reporte no puede ser registrado porque no existen brigadas definidas según el tipo o el área seleccionada”	Satisfactorio
Se accede a la opción modificar reporte sin haber marcado un reporte de la lista	-	El sistema muestra un mensaje de: “Debe seleccionar un reporte para modificarlo”	Satisfactorio
Se accede a la opción	-	El sistema muestra un mensaje de:	Satisfactorio

eliminar reporte sin haber marcado un reporte de la lista		"Debe seleccionar un reporte para eliminarlo"	
Se intenta modificar un reporte o estado cuyo estado no es el inicial	-	El sistema muestra un mensaje de: "El reporte seleccionado no puede ser modificado ni eliminado porque ya ha sido aceptado por la brigada"	Satisfactorio
Se define un reporte con un usuario que no existe en el directorio LDAP UCI	1-1-0-1-1	El sistema muestra un mensaje: "El usuario definido no existe"	Satisfactorio

- **Caso de prueba para el caso de uso Gestionar recurso**

Tabla 10. Matriz del caso de prueba Gestionar recurso

Escenario	Combinación	Respuesta esperada	Resultado
Se sincroniza la aplicación sin que exista un archivo de datos del Assets	-	El sistema muestra un mensaje de: "No existe una fuente de datos para sincronizar los recursos."	Satisfactorio
Se sincroniza correctamente el sistema a partir de una fuente de datos.	-	El sistema actualiza el estado de los recursos, la cantidad y las unidades además de registrar la fecha de la sincronización	Satisfactorio
El archivo Assets contiene errores del formato	-	El sistema muestra el mensaje de: "ha ocurrido un error durante la sincronización con el archivo de datos"	Satisfactorio

Tabla 11. Matriz del caso de prueba: Gestionar recurso.

- **Caso de prueba caso de uso actualizar estado de reporte**

Tabla 12. Variables del caso de uso: Actualizar estado de reporte.

No.	Nombre	Descripción	Tipo	Nulo
1	Estado	Define los posible estados a los que puede ser actualizado un reporte	Lista desplegable	No
2	Causa	Define una posible causa o nota que permita comprender el porqué del nuevo estado.	Texto	Si

Tabla 13. Matriz del caso de prueba Actualizar estado de reporte.

Escenario	Combinación	Respuesta esperada	Resultado
Se define el estado y la causa correctamente	1-1	El sistema modifica el estado de reporte, actualiza el historial de reporte y muestra el mensaje: “Reporte actualizado satisfactoriamente.”	Satisfactorio
Se introduce el nuevo estado del reporte pero no se define una causa	1-1	El sistema modifica el estado de reporte, actualiza el historial de reporte y muestra el mensaje: “Reporte actualizado satisfactoriamente.”	Satisfactorio
Se deja en blanco el campo de estado del reporte	0-1	El sistema muestra el mensaje de: “Debe definir el nuevo estado del reporte”	Satisfactorio
Se deja intacto el estado del reporte	1-0	El sistema muestra un mensaje: “El estado ha sido modificado satisfactoriamente”	Satisfactorio.

Para el desarrollo de las pruebas se utilizó el método de particiones equivalentes, que permite dividir la etapa de pruebas en iteraciones, lo que garantiza que en cada iteración se pueda solucionar las No Conformidades de la iteración anterior. Para la solución que se presenta se aplicaron dos iteraciones de

prueba. Las No Conformidades detectadas durante ambas iteraciones por cada caso de uso pueden observarse a continuación.

Tabla 14. No Conformidades por Caso de uso

Caso de prueba	Iteración	No Conformidades	Cerradas
Autenticar usuario	1ra	2	2
	2da	1	1
Buscar reporte	1ra	1	1
	2da	0	0
Gestionar recurso	1ra	6	6
	2da	3	3
Gestionar brigada	1ra	1	1
	2da	0	0
Gestionar reporte	1ra	4	4
	2da	2	2
Actualizar estado de reporte.	1ra	2	2
	2da	0	0
Graficar reporte.	1ra	1	1
	2da	0	0

A partir del cierre de las No Conformidades detectadas en las dos iteraciones de la etapa de prueba se demuestra el cumplimiento de los objetivos planteados para la investigación. Como contrapartida, el

proceso de prueba fue validado mediante pruebas funcionales realizadas por los clientes finales, que aprobaron la herramienta y certificaron su realización satisfactoria.

3.2.2 Pruebas para requisitos no funcionales

Para realizar las pruebas sobre los requisitos no funcionales de la aplicación, se utilizó la herramienta JMeter. Esta herramienta es un componente que permite realizar pruebas de esfuerzo y carga sobre los navegadores clientes, de forma que se puedan validar los atributos de calidad de una aplicación web de acuerdo a las propiedades de los nodos involucrados, los tiempos de respuesta, la veracidad del conjunto de acciones que se ejecutan mediante la red según tasas de conexiones simuladas desde un solo ordenador cliente.

Para realizar dichas pruebas se preparó un servidor con las siguientes propiedades:

- Procesador Pentium IV/AMD, 2.4 GHz o superior.
- Memoria RAM: 1 GB.
- Tarjeta de Red Ethernet 100 Mb/s
- Capacidad: 80 GB.

El ordenador cliente se probó con las siguientes propiedades.

- Procesador Intel Celeron 1.6 GHz.
- Memoria RAM: 512 Mb
- Tarjeta de Red Ethernet: 100 Mb/s
- Capacidad: 60 Gb.

Durante las pruebas de carga el servidor soportó 243 conexiones simultáneas a una tasa estable de respuesta de 61 ms, con una carga para el cliente de 3.4 Mb durante el primer almacenamiento del contenido en la caché del navegador cliente.

Para las pruebas con la información almacenada previamente en la caché, las tasas estables se comportaron para 58 peticiones de carga, 61 kb de datos, con una cantidad máxima de 315 conexiones simultáneas.

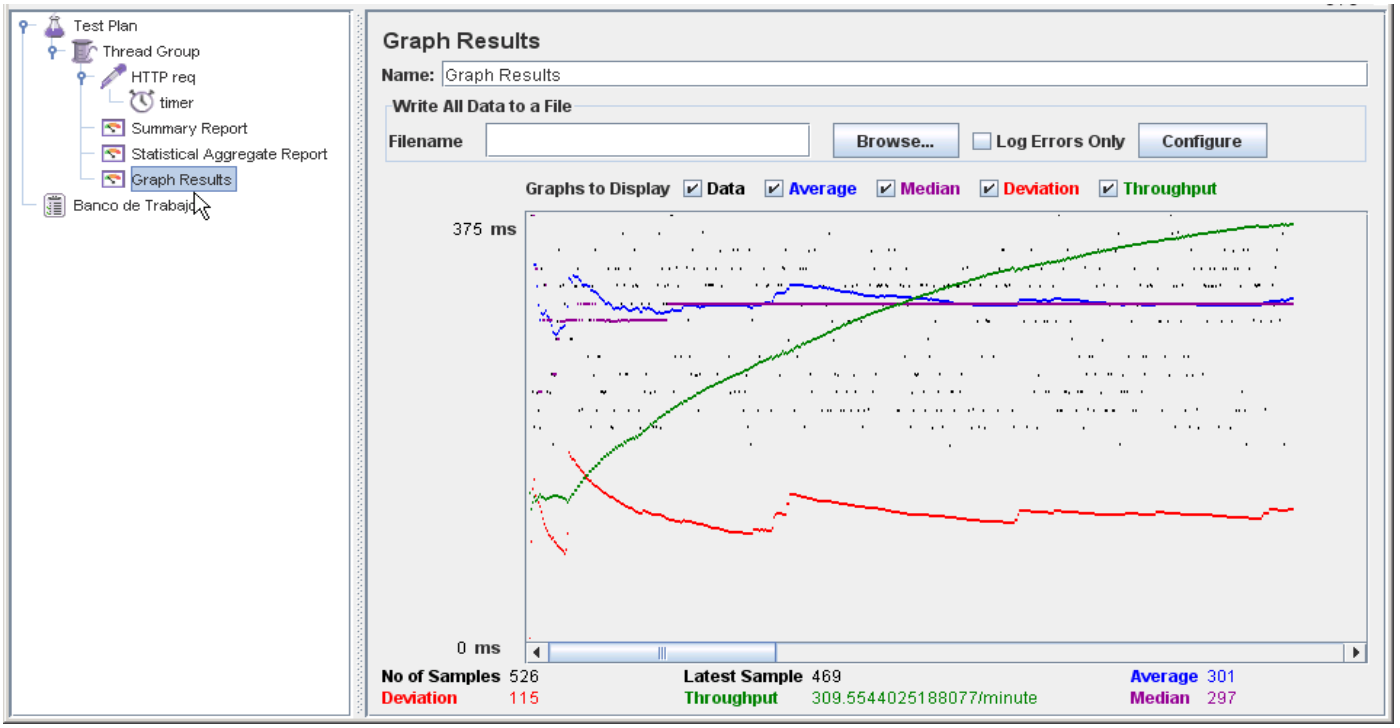


Fig. 12. Prueba de rendimiento

3.3 Prueba de rendimiento

Para dichas cantidades de conexiones, se aseguran todos los requisitos no funcionales, por lo que, dada las condiciones del ambiente de despliegue del sistema, las características de los ordenadores clientes y el servidor, además de las posibles conexiones simultáneas, se considera que las pruebas son satisfactorias.

3.4 Conclusiones parciales.

En la etapa de implementación se logró completar el desarrollo del sistema siguiendo los principios, modelos y definiciones de la etapa del análisis y diseño, obteniéndose una herramienta funcional que satisface los requisitos establecidos. La representación mediante el modelo de implementación permitió reflejar la comunicación entre los principales componentes del sistema, lo que servirá de base para el futuro de desarrollo de este. Las pruebas reflejaron que los requisitos establecidos fueron cumplidos según los parámetros de calidad de software que se tuvieron en cuenta y garantizaron determinar las No Conformidades así como la evolución del sistema hasta la herramienta que satisface los problemas de automatización del negocio.

Conclusiones

Una vez obtenida una versión más completa del Sistema de Gestión de Reportes de la Dirección de Mantenimiento, se puede concluir que:

1. El establecimiento de un mecanismo de sincronización con el estado del Sistema de Control de Activos (ASSET) permitió que el sistema operara con datos reales, lo que maximiza su capacidad de respuesta ante la disponibilidad de recursos, eliminando las inconsistencias presentadas en la versión anterior y apoyando las tareas ejecutadas por los técnicos de mantenimiento.
2. La utilización de un sistema de prioridades así como la asignación directa a brigadas gestionadas dentro de la aplicación, garantizaron que el sistema se comporte de manera fiel y responda a las necesidades reales de la Dirección de Mantenimiento, mejorando los procesos de distribución de reportes según el área, asegurando la trazabilidad y brindando mejor clasificación.
3. El sistema de Gestión de Reportes, con la incorporación de un nuevo modelo de seguridad y la actualización de los reportes gráficos que se envían a la Dirección, aseguran que se mejore la capacidad de gestión de la Dirección de Mantenimiento, mejora las búsquedas para las tareas de análisis, garantiza mejores estimaciones e involucra a los usuarios en un proceso más cómodo y seguro, lo que influye positivamente en la toma de decisiones.

Recomendaciones

Se recomienda lo siguiente:

1. Integrar la aplicación a otros sistemas de control de activos de la universidad, de forma que se garantiza una mejor gestión económica para los recursos y reportes.
2. Desarrollar un sistema de notificación que permita a los usuarios que emiten reportes obtener el estado actual de estos a partir de las propias tareas de la Dirección de Mantenimiento.

Bibliografía

POTENCIER, F. ZANINOTTO, F. *Symfony, la guia definitiva.* libroweb,EE.UU, 2008.

ADDISON W. *Softwre development for small team: A RUP - centric approach.* s.l., 2004. ISBN: 978-0321-199-50-8.

ANGELL, I. SMITHSON, S.*Information System Management, Opportunities and Risks.* 1991.

ARREGOCES, M. PORTOLANI, M.*Data Center Fundamentals.* Indiana : Cisco Press, 2011. ISBN: 1-58705-023-4.

ASF. *Apache Software Fundation*. [En línea] Apache Software Fundations. [Citado el: 26 de Julio de 2014.] Disponible en http://www.apache.org/.

BENVENUTO, A.*Implantacion de Sistemas ERP, su impacto en la gestión de empresa e integración con las TIC.* Universidad de Concepción,Santiago de Chile, 2006. ISSN: 0718-4662.

BERGER, D P.*Six steps to condition-based maintenance.* s.l. : Industry Publications, 2006.

BLANCO, L.*La Informática en la direccion de empresas.* Félix Varela, La Habana , 2011.

BROOKS, F P. *Prácticas de Software*. [En línea] 2011. [Citado el: 11 de Agosto de 2014.] Disponible en http://www.practicasdesoftware.com.ar.

CEEC.*Informe Central.* CEEC, La Habana , 2012.

DE LOS ÁNGELES, O.*Marcos de Trabajo para el desarrollo de Aplicaciones Web.* 2012.

DOMÍNGUEZ, D. *NetBeans IDE 4.1, la alternativa a Eclipse.* Iberprensa Madrid , 2004.

DUGGAN, E. REICHGELT.*Measuring Information Systems Delivery Quality. Hershey.* s.l. Idea Group Inc., 2012. ISBN 1-59140-859-8.

PATSI,GDR.*Manual de usuario.* Universidad de las Ciencias Infomáticas, La Habana , 2010.

ESPINOSA, F.*Sistema de informacion para la gestion de empresa.* s.l. : Universidad de Talca, 2013.

EXTJS . *Ext JS*. [En línea] Sencha Inc. [Citado el: 11 de Septiembre de 2014.] http://www.extjs.com.

FAST,Report. *Fast Report*. [En línea] [Citado el: 21 de Septiembre de 2014.] Disponible en http://www.fast-report.com/en/blog/42/show/.

FERNANDO, A. *Monografías*. [En línea] Monografías, 2010. [Citado el: 28 de Julio de 2014.] Disponible en http://www.monografias.com/trabajos89/introduccion-al-html-5/introduccion-al-html-5.shtml..

Referencias Bibliográficas

FOWLER. *UML destilled, A brief guide to standard object modeling languaje.* Tercera. s.l. : Pearson Education, 2007. ISBN: 978-8131-715-65-9.

FOWLER, M, RICE, D y FOEMMEL, M. *Patterns of Enterprise Application Arquitecture.* s.l. : Addison Wesley, 2002. ISBN : 0-321-12742-0 .

GERNER, J. *Professional LAMP: Linux, Apache, MySQL and PHP 5 Web Development.* : Wiley Publishing, Indianapolis, 2011. SBN: 978-0-7645-9723-7.

INFANTE, J. HERNANDEZ , Y.*Sistema de Gestión de Reportes Dinámicos.* Tesis ,Universidad de las Ciencias Informáticas, La Habana, 2009.

MORALES.Y *info@tltas*. [En línea] info@tltas. [Citado el: 16 de Septiembre de 2014.] Disponible en http://afide.inder.cu/PDF/AREA%202/AR/AR019.pdf

JACOBSON, I. BOOCH, G. RUMBAUGH, J. *El proceso unificado de desarrollo de software.* Pearson Educación, Madrid, 2000. ISBN: 84-7829-036-2.

KRCUTEN, P. *The Rational Unified Process, An Intruduction.* s.l. : Addison Wesly, 2004. ISBN: 978-032-1197-70-2.

LAUDON, L. LAUDON, K.*Sistemas de información gerencial- Administración de la empresa digital.* s.l. : Prentice Hall : Pearson Education, 2006.

MCCONELL, S. *Professional Software Development.* s.l. : Addison Wesley, 2003. ISBN : 0-321-19367-9.

MERLEY, J. *¿Por qué elegir PHP? ,*2012.

MEYER, B. *Construcción de Sofware Orientado a Objetos.* Féliz Varela, La Habana, 2009.

OMG. *Unified Modeling Language Especification.* s.l. : Pearson Education, 2001.

PÉREZ , C. *Zafiro Business Software*. [En línea] www.ZafiroERP.com. [Citado el: 16 de Septiembre de 2014.] Disponible en http://zafirosoft.com/noticias/la-funcion-de-los-reportes-en-un-sistema-erp/.

PÉREZ, J. 2014.*¿Por qué JavaScript?* s.l. : La web del programador, Blog, 2014.

POSTGRESQL D.*PostgreSQL 8.4.0 Documentation.* s.l. Manual: PostgreSQL Global Development Group, 2009..

PRESSMAN, R.*Ingeniería de Software: un enfoque práctico.* s.l. : Mc Graw Hill, 2008. ISBN: 970-10-5473-3.

PROBASCO, L. *The ten essentials of RUP. The essence of a effective development proceess.* Lexington : Rational Software, 2005.

Referencias Bibliográficas

QUESADA, A. *Pruebas de Software.* 2009.

ROBERTSON. *Mastering in Requirements Process.* s.l. : Pearson Education, 2006. ISBN: 978-8131-711-94-1.

KERTON, L.*Sistema estratégico para la gestión avanzada de recursos y procesos EBMS DSerp Agro.* : VI Taller de Inteligenca Organizacional y Gestión Empresarial, La Habana, 2013. ISBN: 978-959-286-019-3.

STROUPTUS, VB.*The C++ Programming Language (6th Edition ed.).* : Addison Wesley, New Jersey, 2011.

VAN DER GRAAF, S.*Information Communication Technologies and Emergin Business Strategies.* : Idea Group Inc, London UK, 2007. ISBN 1-59904-236-3.

VISUAL P.*Reasons to Choose Visual Paradigm.* s.l. : Visual Paradimg Group, 2011.

WALLACE, D y RAGGET, I.*Extreme Programming for Web Projects.* s.l. : Addison Wesley, 2002. ISBN : 0-201-79427-6.

Anexos

Modelo de entrevista para especialistas de la Dirección de Mantenimiento

Buenos días:

Como parte del proceso de preparación de mi trabajo de diploma, estoy recopilando información acerca del estado actual del proceso de gestión de reportes de la Dirección de Mantenimiento. Sería de gran ayuda para mi investigación que usted me diera su opinión sobre este tema.

Pregunta # 1: ¿Cuál es su nombre?

Pregunta # 2: ¿Qué cargo ocupa en esta dirección?

Pregunta # 3: ¿Cómo se realiza la captura de reportes por parte de los usuarios?

Pregunta # 4: ¿Qué herramientas informáticas intervienen en este proceso?

Pregunta # 5: ¿Cuáles son las limitaciones o deficiencias de estas herramientas?

Pregunta # 6: ¿Cómo se realiza la verificación de cantidad de recursos?

Pregunta # 7 ¿Esta verificación se realiza manual o automática?

Pregunta # 8: ¿Las herramientas actuales permiten el análisis estadístico y la toma de decisiones?

Pregunta # 9. ¿Considera usted necesario de una nueva herramienta informática?

Muchas Gracias, su colaboración será útil para mi investigación.