Universidad de las Ciencias Informáticas

Facultad 3

Título: Desarrollo de funcionalidades de apoyo al proceso control de Neumáticos y Baterías del Sistema Control de Flotas y Mantenimiento de la Dirección de Transporte de la Universidad de las Ciencias Informáticas

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

Autor: Jorge Enrique Zamora Pérez

Tutor: Ing. Pedro Manuel Alás Verdecia

La Habana, Cuba

Junio de 2013

Declaración de autoría

Declaro Jorge Enrique Zamora Pérez ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Jorge Enrique Zamora Pérez

_____

Firma del Autor

Ing. Pedro Manuel Alás Verdecia

_____

Firma del Tutor

Datos de contacto

Autor:

Jorge Enrique Zamora Pérez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: jezamora@estudiantes.uci.cu

Tutor:

Ing. Pedro Manuel Alás Verdecia

Universidad de las Ciencias Informáticas, La Habana, Cuba

Email: pmalas@.uci.cu

Agradecimientos

A mi mamá, por el gran amor y por el apoyo ilimitado e incondicional que siempre me has dado, por tener siempre la fortaleza de salir adelante, sin importar los obstáculos, por haberme formado como un hombre de bien, y por ser la mujer que me dio la vida y me enseñó a vivirla no hay palabras en este mundo para agradecerte, mamá.

A mi padre, por las enseñanzas que me has dado, y por darme ánimos siempre, muchas gracias, papá.

A toda mi familia, por sus palabras de aliento y sus buenos deseos. A todos mis amigos y compañeros de la universidad por su apoyo en las buenas y malas experiencias.

A todos aquéllos que contribuyeron en mi formación académica y profesional: a mis profesores, que compartieron conmigo sus conocimientos a lo largo de mí educación universitaria; especialmente a mi tutor de tesis, Pedro, por su apoyo y paciencia para la elaboración de este trabajo.

Resumen

En el departamento de Soluciones Empresariales perteneciente al Centro de Informatización para la Gestión de Entidades, se está desarrollando el Sistema Control de Flota y Mantenimiento para la gestión de los procesos que se realizan en el área de Transporte de las Universidad Ciencias Informáticas.

En el presente trabajo se realiza el análisis, diseño e implementación del proceso Control de Neumáticos y Baterías de dicho sistema, con el objetivo de desarrollar funcionalidades que permitan la inserción, análisis y recuperación de toda la información referente a los neumáticos y baterías del área de Transporte de las Universidad Ciencias Informáticas y lograr una integración exitosa con el resto de los procesos del área de Transporte.

Con la implementación de estas funcionalidades se contribuye al ahorro de los recursos materiales del área de Transporte, facilitando el trabajo y la toma de decisiones con la realización de una serie de reportes que le posibilitan controlar la información referente a los datos de los neumáticos y baterías.

Se integran los procesos, Generación de Órdenes de Trabajo y Control de Vehículos con el proceso de Control de Neumáticos y Baterías, permitiendo así tener un control sobre el estado y la ubicación de los neumáticos y baterías dentro del parque vehicular.

Palabras clave:

Neumáticos, Baterías

Tabla de Contenidos

Introducción

En la actualidad existe un gran avance tecnológico que ha posibilitado que a través de sistemas informáticos sean realizadas labores que anteriormente se hacían de forma manual. Debido a este avance tecnológico está surgiendo una tendencia por parte de las organizaciones de informatizar sus procesos en aras de tener la información centralizada, controlada y de fácil acceso, contribuyendo en la toma de decisiones.

En la Universidad de las Ciencias Informáticas la Dirección de Transporte cuenta con su propio parque de vehículos y enfrenta en la actualidad serios problemas para controlar la información relacionada con los neumáticos y baterías. Información que actualmente se gestiona de forma manual lo que ha provocado que cada vez que se realiza una auditoría en la empresa se detecten violaciones de las políticas estipuladas por el Ministerio de Transporte, violaciones tales como la pérdida de información, la omisión de información o la generación tardía de información debido a los grandes volúmenes de información que se manejan en el parque, o como consecuencia de errores humanos. Además provoca que la sustitución o rotación de los neumáticos no se lleve a cabo cuando se requiere, debido a que se desconoce la ubicación exacta de cada neumático o el desgaste del mismo producido por su uso. También la información referente a las baterías de los vehículos se gestiona de forma manual, provocando que dicho proceso se alargue en el tiempo mucho más de lo que se planifica y resulte un proceso muy engorroso para el personal de esta área. Situación que ha generado grandes problemas para la toma de decisiones tanto administrativas como operacionales dentro de la empresa.

Debido a esta problemática la Dirección de Transporte ha decidido realizar la informatización de los principales procesos que se llevan a cabo dentro de sus áreas, incluyendo el proceso de Control de Neumáticos y Baterías de los vehículos. Para ello ha acudido al Centro de Informatización para la Gestión de Entidades CEIGE de la Universidad de las Ciencias Informáticas UCI, el cual tiene experiencia en el desarrollo de sistemas para la gestión de mantenimiento. Además del desarrollo alcanzado por el centro en dominios de gestión de información relacionada con el control de mantenimiento, patrimonio, inventario y vehículos. Situación que permite un aprovechamiento de los activos de software, así como las experiencias de su equipo de trabajo.

De la problemática planteada se identificó el siguiente **problema a resolver**. ¿Cómo garantizar la inserción, análisis, y recuperación de la información del proceso Control de Neumáticos y Baterías de la Dirección de Transporte de la UCI?

Se plantea como **objeto de estudio**: la gestión de los neumáticos y baterías en los sistemas de gestión de control de flota y mantenimiento vehicular.

Y el **campo de acción** se enmarca en: el proceso Control de Neumáticos y Baterías del área de Transporte de la UCI.

Se plantea como **idea a defender:** el desarrollo de funcionalidades de apoyo al proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento permitirá la inserción, análisis y recuperación de la información referente a los neumáticos y baterías del Área de Transporte de la UCI.

Para la solución del problema planteado se define como **objetivo general**: desarrollar funcionalidades que garanticen la inserción, análisis y recuperación de la información del proceso Control de Neumático y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección Transporte de la UCI.

Para dar cumplimiento al objetivo general planteado, se proponen los siguientes **objetivos específicos**:

1. Elaborar el marco teórico de la investigación.
2. Realizar el análisis del proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
3. Diseñar una solución que permita apoyar el proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
4. Implementar el diseño de la solución propuesta que permita apoyar el proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
5. Validar la propuesta de solución.

Para el desarrollo del presente trabajo se hacen uso de los siguientes **métodos científicos de investigación**:
Histórico Lógico: se centró en el estudio de la gestión del proceso de Control de Neumáticos y Baterías desde un enfoque histórico lógico, estudiando el desarrollo de este proceso en el área de Transporte de la UCI.

Analítico-Sintético: permitió separar la información para su estudio y procesamiento, facilitando el descubrimiento de las características generales y las relaciones esenciales del proceso de Control Neumáticos y Baterías.

Entrevista: se realizaron entrevistas a personal calificado y conocedor de la actividad del proceso de Control Neumáticos y Baterías, ayudando a la adquisición de información y captura de requisitos. Dichas entrevistas se encuentran recogidas en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

Observación: posibilitó revelar las relaciones esenciales y las características fundamentales de la actividad del proceso de Control Neumáticos y Baterías, accesibles a la detección de la percepción.

Este documento está compuesto de los siguientes capítulos:

Capítulo 1. Fundamentación teórica

En este capítulo se analiza la gestión de los neumáticos y baterías en varios sistemas de control de flota y mantenimiento nacionales e internacionales, determinando a partir de este análisis las razones por las cuales no fueron utilizados y las funciones comunes en ellos que constituyen soluciones deseables a parte del problema. Además se justifica el empleo de cada una de las herramientas, metodologías, lenguajes de desarrollos, marcos de trabajo y tecnologías en el desarrollo del proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento.

Capítulo 2. Modelado del negocio y requerimientos

En este capítulo se realiza la modelación del proceso de Control de Neumáticos y Baterías del área de Transporte de la UCI y se identifican y validan los requisitos.

Capítulo 3. Diseño, implementación y pruebas de la propuesta solución

En este capítulo se exponen los artefactos generados durante el diseño y la implementación de la solución, así como las métricas y pruebas utilizadas para su validación, las cuales aseguraran el correcto funcionamiento de las funcionalidades implementadas para apoyar el proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.

Capítulo 1. Fundamentación teórica

Introducción

El presente capítulo comienza abordando el tema de cómo se realiza el proceso Control de Neumáticos y Baterías del área de Transporte de la UCI, así como la importancia que posee. Se realiza un análisis de la gestión de los neumáticos y baterías en varios sistemas de control de flota y mantenimiento nacionales e internacionales. Este capítulo finaliza con la justificación de la selección de cada una de las herramientas, metodología, lenguaje de desarrollo, marcos de trabajo, gestores de base de datos y de aplicación web utilizados para el análisis, diseño e implementación del proceso Control de Neumáticos y Baterías.

1.1. Proceso de Control de Neumáticos y Baterías del área de Transporte de la UCI

En el área de Transporte de la UCI el principal objetivo de la gestión de neumáticos y baterías es registrar toda la información necesaria para elaborar el expediente de cada activo, dígase datos generales como marca, modelo, norma o medida tanto de los neumáticos como de las baterías. Además de otros datos como el vehículo en el que están asignados y la fecha de montaje. El proceso Control de Neumáticos y Baterías del área de Transporte de la UCI se relaciona específicamente con los procesos de negocio: Generación de Órdenes de Trabajo y Control de Vehículos, estos manejan la información relacionada con la cantidad de kilómetros recorridos por cada auto y las órdenes de trabajo que indican cuando un vehículo deberá sustituir o rotar un neumático o batería. Actualmente no existe una integración entre ellos, provocando falta de control sobre la cantidad de kilómetros recorridos por un neumático o el tiempo de uso de una batería. (1) El proceso Control de Neumáticos y Baterías del área de Transporte de la UCI enfrenta serios problemas para controlar la información relacionada con los neumáticos y baterías. Información que al ser gestionada de forma manual resulta muy engorroso su gestión provocando la pérdida, omisión y generación tardía de la información debido a los grandes volúmenes de información que se manejan en el parque.

1.2. Gestión de los neumáticos y baterías en los sistemas de gestión de flotas vehiculares

A continuación se realiza un análisis de sistemas con características similares que se utilizan para la gestión de flota y mantenimiento vehicular nacionales e internacionales.

1.2.1. Sistemas Internacionales

En el mundo existen varios sistemas de gestión de flotas vehiculares, los cuales brindan facilidades en el control y administración de todas las tareas que están relacionadas a los vehículos que posea una determinada empresa. Dentro de estos se encuentran:

Sistema de Gestión de Flotas Novatrans

Es un software de transporte desarrollado por la empresa malagueña Solbyte orientado al sector del transporte de mercancías por carretera. Software de uso fácil e intuitivo que pretende cubrir todas las necesidades en la gestión de una empresa de transporte incorporando novedades tales como un gestor documental en todos los módulos y una base datos SQL Server. Además incluye una personalización completa del diseño de la aplicación para la empresa y una gestión exhaustiva de permisos para cada usuario. Para ello, Novatrans, agrupa la información en 9 bloques generales. (2)

A continuación se presentan de forma breve cada uno de ellos.

Almacén: en este completo apartado encontramos toda la información referente a artículos que se encuentran en almacén, entradas y salidas de los mismos, artículos en stock, etc. (2)

Facturas: gestiona la facturación a clientes, así como la recepción de facturas. Con este sistema podremos gestionar los cobros y pagos de una manera sencilla y cómoda. En este módulo puede crear sus propias facturas, mediante el generador de facturas. (2)

Logística de almacén: a través de su sección de entrada y salida se pueden registrar todos y cada uno de los productos que entran y salen. (2)

Estadísticas e Informes: posee un amplio abanico de informes y estadísticas que incluye el control de consumo medio de los vehículos de gasoil o de otro tipo de carburante, control de los beneficios (Gastos e Ingresos) y el control de la facturación mensual. (2)

Producción: en este apartado se define qué se le va a cobrar a un cliente por origen y destino determinado. Además se indican los desplazamientos realizados en un intervalo de fechas, por un conductor y vehículo con remolque. (2)

Logística de viajes y grupaje: controla todos los pedidos que surgen se pueden dar de alta en este módulo y a posteriori. La empresa puede recoger todos y cada uno de los vehículos de los que dispone, ya sea cabezas tractoras, semirremolques, así como el conductor encargado de cada vehículo. (2)

Taller: controla toda una serie de aspectos referentes a:

• Partes de taller: recoge todas y cada una de las operaciones de taller realizadas al vehículo en cuestión. En taller se controla cuando un vehículo ha ido al taller, fecha de entrada y salida, los kilómetros que tiene en ese momento, si se le ha hecho una reparación o un mantenimiento. (2)

• Revisiones: en este apartado, Novatrans ofrece un servicio de avisos para cada una de las revisiones que deben hacerse al vehículo de la empresa. Este tipo de avisos de revisiones contiene un filtro de búsqueda que facilita observar el listado rápidamente.

• Gastos de telefonía: consiste en un listado que permite ver los gastos que se generan por cada teléfono de la empresa. (2)

• Neumáticos: para el control de neumáticos se deben de dar de alta cada uno de forma individual, indicando matrícula del mismo, dimensiones, modelo, etc. Se controlará la vida del neumático, en qué vehículos ha estado, cuántos kilómetros ha durado, el motivo de cambio de neumático y la posición del mismo. (2)

• Gastos de gasoil: en este apartado se registran todos y cada uno de los gastos generados por el repostaje de los vehículos de la empresa. (2)

Tesorería: en este apartado se incluyen diferentes opciones para la gestión de facturas de proveedores, generación de recibos, control de cobros y pagos, y control de las nóminas. (2)

Utilidades: En esta sección se maneja toda una serie de aspectos como:

• Avisos: su empresa puede generar avisos propios, que la mantengan informada sobre todos aquellas incidencias que se produzcan y necesiten revisarse, cambiarse, proveerse, etc. (2)

• Avisos predefinidos: ofrece una serie de indicaciones predefinidas a la empresa, sobre aspectos relacionados con vehículos y conductores. (2)

• Datos maestros: la empresa puede realizar una visión general de todas las informaciones que haya insertado en el software. (2)

• Exportación: permite la exportación de datos de gestión de flotas a otros programas de contabilidad. (2)

• Perfiles: a través de la opción de perfiles, Novatrans permite la adaptación del software según la persona que vaya a usarlo. (2)

• Usuarios: la opción de usuarios permite ajustar el sistema, según la persona que vaya a proceder a su uso. Dependiendo de los permisos que se le hayan concedido previamente, luego se mostraran ciertas pantallas o no. (2)

• Proveedores: la ficha de proveedores cuenta con datos semejantes a los de clientes. En ella podremos registrar todo tipo de proveedores de la empresa. (2)

• Tarjetas: en esta sección podremos registrar las tarjetas de crédito de gasoil con sus correspondientes números, fecha de caducidad, tipo, número pin, así como el vehículo en el que se encuentran. (2)
• Vehículos: en este apartado se trata de ver todas y cada una de las características que afecten a todos los vehículos de la empresa: datos generales, permisos, seguros, ficha técnica, pagos bancarios y alquileres, y todo lo referente al mantenimiento del mismo. (2)

El sistema de gestión de flotas Novatrans cuenta con un módulo, Taller, que gestiona toda la información referente a los neumáticos. Controla la vida del neumático, en qué vehículos ha estado, cuántos kilómetros ha durado, el motivo de cambio de neumático y la posición del mismo.

Sistema de Gestión de Flotas Transportex
Es la aplicación informática más completa y sencilla para la gestión de empresas de transporte de mercancía vía terrestre. Transportex está desarrollado en base a experiencias y situaciones reales en organizaciones de este tipo. Diseñada por y para las empresas de distribución. (3) Compuesta por una innumerable cantidad de poderosos instrumentos integrados, en resumen, dentro de tres módulos principales:

Gestión de Neumáticos: permite conocer detalladamente las características de cada neumático; como la distancia recorrida, espesor, modelo, proveedor, ubicación en eje del vehículo o almacén, tendencias de desgaste, entre otros. También gestiona su inventario de neumáticos de manera personalizada y asigna neumáticos de forma gráfica individualmente o en lotes. (3)
Actividades de Mantenimiento: programa y consulta tareas de mantenimiento preventivo o correctivo para vehículos y neumáticos, revisiones técnicas en toda la flota o en alguna unidad en particular, lleve registro de gastos asociados; costos por mano de obra, proveedores, materiales y repuestos utilizados, establezca un responsable, alarmas y tiempos de periodicidad mediante un sistema de avisos inteligente. (3)
Control de Viajes: establece el registro de viajes de despachos de mercancía realizados por los vehículos, controlando eficientemente los adelantos y montos a pagar a cada conductor como el salario, tickets de alimentación, días feriados, repartos, bonos extras, etc. a través de tablas de valores de costos por empresas, destinos geográficos, usos de camiones y remolques. Lleva un control detallado de los montos del flete, conceptos de gastos tales como combustible, estadías, comida, peajes, caleta, viáticos, multas, información de guías y detalles de la carga. (3)

Adicionalmente, Transportex posee módulos auxiliares que se integran eficazmente a estos módulos principales, tales como gestión de vehículos, empresas, proveedores, empleados, documentos, tablas de distancias geográficas, entre otros. (3)

El sistema de gestión de flotas Transportex cuenta con un módulo, Gestión de Neumáticos, que permite controlar toda la información referente a los neumáticos. Permite conocer detalladamente las características de cada neumático y asigna neumáticos de forma gráfica individualmente o en lotes.

Sistema de Gestión de Flotas de EUGCOM

El Sistema de Gestión de Flotas está diseñado para poder ejecutar todas las acciones que se generan en torno a una completa Administración y Gestión de flotas de vehículos (4), presentando los siguientes módulos:

Vehículos: módulo de Mantención, permite crear, modificar, eliminar e imprimir todos los datos de los vehículos del sistema. (4)

Proveedores: módulo de ingreso y mantención de todos los proveedores de insumos y servicios del sistema. (4)

Combustible: módulo de ingreso y mantención de los datos que componen el movimiento de cargas de combustible efectuados a los vehículos. (4)

Neumáticos: módulo de ingreso y mantención de los datos que componen el movimiento de cambio de neumático de los vehículos. (4)

Mantención de Servicios: módulo de mantención de los servicios que componen los movimientos relacionados a los servicios realizados al vehículo. (4)

Orden de Trabajo: módulo de ingreso y mantención de las órdenes de trabajo emitidas a los vehículos para efectuar cierta cantidad de servicios en él. (4)

Cargas de Combustible: módulo de búsqueda de datos que permite ejecutar consultas y obtener resultados acerca de las cargas de combustible registradas en el sistema. Genera un informe sumamente personalizado según los requerimientos del usuario y permite calcular los datos más importantes por cada vehículo. (4)

Viajes realizados: módulo de búsqueda de datos de los movimientos de viajes realizados por los vehículos en un lapso de tiempo y según las especificaciones establecidas por el usuario. (4)

Reprocesamiento de Cargas de Combustible: módulo que permite reprocesar los datos automáticos generados en el movimiento de cargas de combustible como distancia y rendimiento; generar un informe de errores de los usuarios en las cargas de combustible registradas en el sistema y reparar automáticamente los más usuales.

Este módulo evita que los informes de combustible contengan información no válida después de editar o eliminar un movimiento. (4)

Tablas Maestras: módulo de Mantención de los datos generales del Sistema de Gestión de Flotas. (4)

El sistema de gestión de flotas de EUGCOM cuenta con un módulo, Neumáticos, que permite el ingreso y mantención de los datos que componen el movimiento de cambio de neumático de los vehículos.

1.2.2. Sistemas Nacionales

En nuestro país existen varios sistemas de control de flotas vehiculares, los cuales brindan facilidades en el control y administración de todas las tareas que están relacionadas a los vehículos que posea una determinada empresa. Dentro de estos se encuentran:

Sistema de Gestión de Flotas Apolo

Es un sistema cliente – servidor, donde la aplicación se encuentra situada en un servidor central y los usuarios del sistema independientemente del lugar de la red donde se encuentren accederán en tiempo real, todos a la misma base informativa, es decir los datos están centralizados y todos se nutren de una misma fuente. Apolo gestiona todo tipo de flotas, con una avanzada metodología de trabajo, está concebido para ofrecer soluciones de asistencia a la gestión y la toma de decisiones. (5)

Entre sus módulos principales se encuentran:

Hoja de Ruta: esta opción permite el registro de Hojas de ruta, documento oficial para la circulación de los vehículos automotores. En la medida que se vayan especificando los datos del kilometraje en la hoja de ruta, se irá actualizando automáticamente esta información en el fichero maestro. (5)

Carta de porte automotor: documento oficial de uso general y obligatorio, que acredita a todos los efectos legales procedentes, la ejecución de la transportación que se realiza al amparo y en cumplimiento de un contrato terrestre de carga concertado entre las partes facultadas para hacerlo. (5)

Dietas: control dietas emitidas, para viajes largos. (5)

Incidencias: en el proceso productivo, puede suceder un sinfín de situaciones meritorias de ser registradas, para esto se define una base de datos de incidencias. Este módulo permite el control de las incidencias, eventualidades y aspectos de interés para el transportista acaecidos en el ciclo de viaje. (5)

Control de accidentes: en esta opción mantendrá un control de los accidentes mantenidos por la flota. Además se llevará el índice de afectaciones. (5)

Planificación de mantenimientos: brinda la posibilidad de planificar los mantenimientos de los equipos que entran al taller, ya sea de clientes naturales o clientes jurídicos. En dependencia del kilometraje concebido, se sugerirán operaciones, que podrán asumirse como operaciones válidas o no, además podrán incluirse otras. Además, se llevará un control histórico de los mantenimientos planificados, con vistas a tener la historia clínica para cada vehículo registrado. También se podrán planificar los mantenimientos, en dependencia del tipo de vehículo, el sistema y el kilometraje del mismo. (5)

Control de neumáticos: en este apartado, se trata el tema de los neumáticos, su mantenimiento y control periódico. Además permite la rotación regular de los neumáticos y su sustitución promueve el desgaste uniforme de los mismos. (5)

El sistema de gestión de flotas Apolo cuenta con un módulo, Control de neumáticos, que posibilita el control y mantenimiento periódico de cada neumático. Además permite la rotación regular de los neumáticos y su sustitución.

Sistema de Mantenimiento Vehicular Venezuela

El sistema Mantenimiento Vehicular permite gestionar los procesos de mantenimiento que se desarrollan en los Centros del Cuerpo de Policía Nacional Bolivariana. Este sistema fue desarrollado basado en los principios de independencia tecnológica utilizando software libre. Además es un sistema de gestión a través del cual se puede manejar toda la información referente a los procesos de mantenimiento preventivo planificado y correctivo de una flota de vehículos, sin importar el tamaño de esta y garantizando una correcta planificación, organización y control en la gestión del mantenimiento de sus unidades. (6)

Entre sus funcionalidades principales se encuentran:

Estructura y Composición: este proceso es el encargado de crear, actualizar y eliminar las estructuras. Además le brinda la posibilidad de definir la organización jerárquica que tienen los elementos que la componen. Puede también establecer las estructuras en cada una de las unidades a través de las áreas. (6)

Clasificadores: es el proceso donde se insertan, modifican y eliminan tipos de mantenimientos, accesorios, causas de fallas, repuestos, herramientas, tipos de unidades, documentos técnicos y unidades de medida. (6)

Configuración: es el proceso a través del cual se definen los diferentes grupos de unidades de acuerdo a su marca, modelo, régimen de mantenimiento, entre otras

características. También se define el umbral de mantenimiento por el cual se van a regir todas las unidades policiales para la realización de los mantenimientos preventivos planificados que le corresponden y los clientes. (6)

Persona: es el proceso que gestiona todo la información referente a los recursos humanos (adicionar, modificar, eliminar, imprimir). (6)

Vehículo: es el proceso desde el cual se va a gestionar toda la información de las unidades policiales (adicionar, modificar, consultar, asignar a dependencia, registrar accidentes, realizar inspecciones técnicas y recepción de unidades). (6)

Taller: es el proceso que gestiona las órdenes de trabajo (generar, emitir, modificar, cancelar, registrar recursos utilizados en una orden). También gestiona las inspecciones técnicas (adicionar, modificar, cancelar e imprimir).

Documentos: es el proceso que gestiona todos los informes generados para las unidades policiales ya sean de baja o de resultado, así como los memorándums. (6)

A pesar de que el sistema no cuenta con un módulo que gestione la información referente a los neumáticos y baterías posee otros componentes como Vehículo que son reutilizables.

1.2.3. Análisis del estado del arte

A partir de la información obtenida sobre el proceso Control de Neumáticos y Baterías del área de Transporte de la UCI, se realizó un análisis con los sistemas anteriormente mencionados, en aras de conocer cómo estos gestionan dichos recursos y determinar si era factible su utilización. El análisis arrojó como resultado que ninguno podía ser utilizado en el área de Transporte de la UCI debido a una serie de inconvenientes encontrados que se exponen a continuación.

Tabla 1 Criterios comparativos a evaluar

Sistema-Criterio	Novatrans	Transportex	EUGCOM	Apolo	MVV
Inserción información neumático	✓	✓	✓	✓	-
Análisis información neumático	✓	✓	✓	-	-
Recuperación información neumático	✓	✓	✓	✓	-

Inserción información baterías	-	-	-	-	-
Análisis información baterías	-	-	-	-	-
Recuperación información baterías.	-	-	-	-	-
Tecnologías libres	-	-	-	✓	✓
Reutilización componentes	-	-	-	-	✓

- Los sistemas **Novatrans, Transportex, Sistema de Gestión de Flotas de EUGCOM** y **Apolo** poseen un módulo a través del cual se pueda insertar, analizar y recuperar la información referente a los neumáticos pero no la referente a las baterías
- El sistema **Mantenimiento Vehicular Venezuela** no posee un módulo a través del cual se pueda insertar, analizar y recuperar la información referente a los neumáticos y baterías.
- Los sistemas **Transportex, Novatrans y Sistema de Gestión de Flotas de EUGCOM** se ejecutan sobre tecnologías privativas, lo cual impide su utilización ya que el centro CEIGE tiene como política el uso de tecnologías libres.

Sin embargo su análisis permitió identificar varias funcionalidades importantes para el control de neumáticos:

- Los sistemas **Novatrans**, **Sistema de Gestión de Flotas de EUGCOM y Transportex,** poseen un módulo que permite insertar, analizar y recuperar información de cada neumático. También permite en qué vehículos ha estado, cuántos kilómetros ha recorrido, el motivo de cambio de neumático y la posición del mismo.
- El sistema **Apolo,** posee un módulo que permite insertar y recuperar información de cada neumático. Además permite la rotación regular de los neumáticos y su sustitución.
- El sistema **Mantenimiento Vehicular Venezuela,** por ser un sistema desarrollado en nuestra propia universidad, con tecnologías no privativas,

permite reutilizar algunos de sus componentes como Taller, Vehículo, Nomencladores y otros activos de software.

Por todas estas razones existe la necesidad de implementar un sistema que permita la inserción, análisis y recuperación de neumáticos y baterías. Además de que no utilice y se ejecute sobre tecnologías privativas. Razones que impulsaron al centro CEIGE al desarrollo de dicho sistema empleando el modelo de desarrollo y tecnologías definidas por el mismo así como el marco de trabajo Sauxe sobre el sobre el cual será desarrollado el sistema.

1.3. Modelo de desarrollo orientado a componentes

El Sistema Control de Flota y Mantenimiento empleará el modelo de desarrollo orientado en componentes creado por el centro CEIGE, basado en buenas prácticas y principios de varias metodologías. Dicho modelo está orientado a las necesidades y artefactos que son generados durante el desarrollo de cualquier software. Además define todos los roles involucrados y sus responsabilidades, las diferentes actividades que realizan, junto con el flujo de estas y los artefactos que se deben generar. (7)

La utilización de este modelo se debe a que es el establecido por el centro CEIGE para el desarrollo de todos sus productos además de las características que presenta:

• **Centrado en la arquitectura**

La arquitectura determina la línea base, los elementos de software estructurales a partir de los elementos de la arquitectura de negocio. Interviene en la gestión de cambios y diseña la evolución e integración del producto. La arquitectura orienta las prioridades del desarrollo y resuelve las necesidades tecnológicas y de soporte para el desarrollo. (7)

• **Orientado a componentes**

Las iteraciones son orientadas por el nivel de significancia arquitectónicas de los componentes, los mismos son abstracciones arquitectónicas de los procesos de negocio y requisitos asociados que modelan, el componente es la unidad de medición y ordenamiento de las iteraciones. (7)

• **Iterativo e incremental**

El equipo de arquitectura, los clientes y la alta gerencia, planifican y coordinan las iteraciones, estas constituyen el desarrollo de componentes, los cuales son integrados al término de la iteración, permitiendo la evolución incremental del producto. (7)

• **Ágil y adaptable al cambio**

Los clientes y funcionales son involucrados en el proyecto, por lo que poseen parte de la responsabilidad del éxito del mismo. Semanalmente se concilian, discuten y

aprueban los cambios. El desarrollo de las partes formaliza solo las características principales de la solución, priorizándose de esta manera los talleres y las comunicaciones. (7)

• **Son utilizados solamente los artefactos necesarios para documentar el producto.** (8)

• **Se modela el negocio por procesos y no por casos de uso.** (8)

Modelo de ciclo de vida de los proyectos del CEIGE: El modelo de ciclo de vida que se presenta en la Figura 1 define las fases por las que transitarán los proyectos de desarrollo de software del CEIGE.

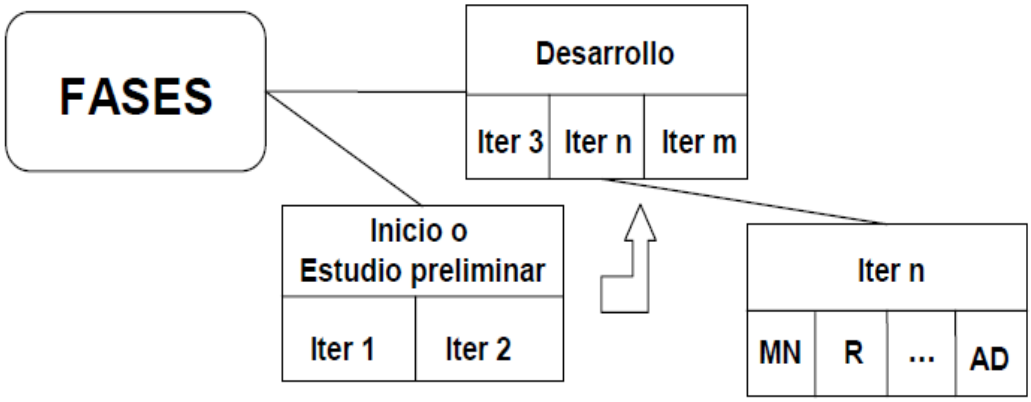


Figura 1 Fases ciclo de vida del CEIGE

Inicio o estudio preliminar: Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto a un alto nivel, la evaluación de la factibilidad del proyecto y el registro de este. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo, y decidir si se ejecuta o no el proyecto. (8)

Desarrollo: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se refinan los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. En esta fase se ejecutan las disciplinas Modelado de negocio, Requisitos, Análisis y diseño, Implementación, Pruebas internas y Pruebas de liberación. (8)

El modelo de desarrollo propuesto describe la secuencia de actividades de alto nivel para la construcción y desarrollo de soluciones. Permite orientar las iteraciones por el nivel de significancia arquitectónicas de cada componente los cuales son integrados al término de la iteración, permitiendo la evolución incremental del producto.

1.4. Técnicas para la captura de requisitos

La captura de requisitos es el proceso mediante el cual los interesados en un sistema de software descubren, revelan, articulan y entienden sus requisitos. En muchos casos, se requiere tiempo para llegar a especificar claramente lo que el interesado espera de la aplicación de software, por lo que se hace necesario por parte de los analistas el empleo de técnicas que permitan establecer una buena comunicación con los interesados del producto y así lograr la satisfacción del cliente. (9)

A continuación se enuncian las principales técnicas utilizadas para la captura de requisitos.

Entrevistas

Es la más tradicional de las técnicas de obtención y consiste en reuniones analista-interesado en las cuales se suceden preguntas y respuestas para extraer el dominio de la aplicación. (10) Roger Pressman en su libro Ingeniería del Requisito un Enfoque Práctico presenta conjuntos de preguntas que se pueden utilizar en el desarrollo de esta técnica, que tiene una alta participación del analista y se realiza en conjunto con otras técnicas.

Desarrollo Conjunto de Aplicaciones del inglés Joint Application Development

Es una técnica de definición de requisitos y de diseño de la interfaz de usuario, basada en reuniones participativas entre clientes, directiva y desarrolladores. Está orientado a proyectos de cliente (o bien sistemas a medida, como también se les conoce), y permite recolectar requisitos eficientemente. Bien utilizada, esta técnica permite ver conflictos entre requisitos y eliminar aquellos menos útiles.

Tormenta de ideas

Es una técnica de reuniones en grupo cuyo objetivo es la generación de ideas en un ambiente libre de críticas o juicios. Puede ayudar a generar una gran variedad de vistas del problema y a formularlo de diferentes formas, sobre todo al comienzo del proceso de captura, cuando los requisitos son todavía muy difusos. (11)

1.5. Técnicas para la validación de requisitos

Revisión Técnica Formal

El objetivo de la revisión formal es descubrir errores en la función, la lógica o la implementación de cualquier producto del software, verificar que satisface sus especificaciones, que se ajusta a los estándares establecidos, señalando las posibles desviaciones detectadas. Es un proceso de revisión riguroso, su objetivo es llegar a detectar lo antes posible, los posibles defectos o desviaciones en los productos que se

van generando a lo largo del desarrollo. Esta característica fuerza a que se adopte esta práctica únicamente para productos que son de especial importancia, porque de otro modo podría frenar la marcha del proyecto. (11)

Prototipos no funcionales

El uso de prototipos para recoger requisitos o comprobar si se han entendido perfectamente es una práctica cada vez más extendida, especialmente en sistemas que suponen un elevado grado de interactividad. En este caso los prototipos a evaluar no serán más que maquetas no operativas o especificaciones formales que un grupo de expertos deberán evaluar. (11)

Casos de prueba

Conjunto de condiciones o variables bajo las cuáles el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio. (11) Los casos de prueba nos ayudan a validar que el aplicativo desarrollado realice las funciones para las que ha sido creado en base a los requerimientos del usuario solicitante.

1.6. Arquitectura de software basada en componentes

Entendemos por arquitectura de software la representación de alto nivel de la estructura de un sistema o aplicación, que describe las partes que la integran, las interacciones entre ellas, los patrones que supervisan su composición, y las restricciones a la hora de aplicar esos patrones. (12)

En general, dicha representación se va a realizar en términos de una colección de componentes y de las interacciones que tienen lugar entre ellos. De esta forma aparecen las arquitecturas basadas en componentes y conectores. Este tipo de arquitecturas son completamente modulares y favorecen la reutilización de todos sus elementos, incluyendo los que definen las distintas relaciones entre ellos. (12)

Para el desarrollo del Sistema Control de Flota y Mantenimiento fue definido, por la dirección del centro CEIGE, la utilización de la arquitectura basada en componentes. Arquitectura enfocada en la división del software en componentes funcionales, lo cual permite un mayor nivel de abstracción y posibilita la reutilización de componentes pre-existentes.

1.6.1. Patrón arquitectónico Modelo-Vista-Controlador

Los patrones arquitectónicos especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Además se pueden ver como la descripción de un problema en particular y recurrente de diseño, que aparece en contextos de diseño arquitectónicos

específicos, y representa un esquema genérico demostrado con éxito para su solución. (13)

El patrón arquitectónico Modelo-Vista-Controlador es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres capas o componentes distintos. El patrón MVC se utiliza frecuentemente en aplicaciones web, donde la vista es la página HTML (*del inglés,* ***Hypertext Markup Language***) que se visualiza en el navegador y el código que proporciona de datos dinámicos a la página, el modelo es el Sistema de Gestión de Base de Datos y la lógica de negocio y el controlador es el responsable de recibir los eventos de entrada desde la vista, enviarlos al modelo y manejar también las respuestas del modelo y transmitirlas hacia la vista. (14)

La utilización de este patrón se debe a que esta implementado dentro del marco de trabajo Sauxe, sobre el cual será desarrollado el sistema, además es el patrón definido por el CEIGE dentro de cada componente que se desarrolle y se considera un patrón pues los mecanismos de diseño se derivan de su funcionamiento. Razones por lo cual será el empleado para el desarrollo del Sistema Control de Flota y Mantenimiento.

1.7. Patrones de diseño

Un patrón de diseño proporciona un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. En él se describe la estructura comúnmente recurrente de comunicar componentes que resuelve un problema de diseño general en un contexto particular. (13)

1.7.1. Patrones GRASP

Los patrones GRASP (patrones generales de software para asignación de responsabilidades) del acrónimo General Responsibility Assignment Software Patterns, describen los principios fundamentales de diseño de objetos para la asignación de responsabilidades. Actualmente se conocen nueve patrones GRASP: Experto, Creador, Alta Cohesión, Bajo Acoplamiento, Controlador, Polimorfismo, Fabricación Pura, Induración y No Hables con Extraños. De éstos se utilizaron los siguientes:

Controlador

Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación. Sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. (15)

Experto

Este patrón posibilita una adecuada asignación de responsabilidades facilitando la comprensión del sistema, su mantenimiento y adaptación a los cambios con reutilización de componentes. (16) Indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que contiene toda la información necesaria para crearlo.

Alta Cohesión

Es una medida que determina cuán relacionadas y adecuadas están las responsabilidades de una clase, de manera que no realice un trabajo colosal; una clase con baja cohesión realiza un trabajo excesivo, haciéndola difícil de comprender, reutilizar y conservar. (16)

Bajo Acoplamiento

Una clase con bajo acoplamiento no depende de muchas otras, mientras que otra con alto acoplamiento presenta varios inconvenientes: es difícil entender cuando está aislada, es ardua de reutilizar porque requiere la presencia de otras clases con las que esté conectada y es cambiante a nivel local cuando se modifican las clases afines. (16) Consiste en tener las clases lo menos ligadas entre ellas, de forma que si se produce una modificación en alguna, tenga la menor repercusión posible en las demás clases.

Creador

El patrón Creador aporta un principio general para la creación de objetos Permite identificar quién debe ser el responsable de la creación de nuevos objetos o clases. (16) Una de las consecuencias de usar este patrón es la visibilidad entre la clase creada y la clase creador. La creación de instancias es una de las actividades más comunes en un sistema orientado a objetos. En consecuencia es útil contar con un principio general para la asignación de las responsabilidades de creación.

1.7.2. Patrones GoF

Son patrones de diseño conocidos por “Gang of Four” o Pandilla de los cuatro. Existen recopilados un total de 23 patrones clasificados en tres grandes grupos: creacionales, estructurales y de comportamiento. De ellos se utilizaron los que se muestran a continuación:

Creacionales

Se encargan de la creación de los objetos ayudando a que el sistema sea independiente de la creación, composición y representación de los objetos. (17)

• **Singleton**: Este patrón asegura que una clase tendrá solo una instancia y provee un punto de acceso global a la misma. El constructor es privado y el método instance ()

es el que devuelve la única instancia de esta clase o la crea si no existe. Se utiliza para las clases que se necesita que exista un solo ejemplar de ellas, evitando de esta forma sobrecargas o problemas de seguridad. (17)

Estructurales

Son los encargados de cómo las clases y objetos están compuestos para formar estructuras más grandes. Los patrones estructurales usan la herencia para componer interfaces u objetos en tiempo de ejecución. (17)

•**Fachada**: Define una interfaz de alto nivel para un conjunto de interfaces de un subsistema que hace que éste sea más fácil de utilizar. (17)

1.8. Tecnologías

Las tecnologías que se emplean para la obtención de la solución que se propone, han sido establecidas por la dirección del proyecto Soluciones Empresariales del centro CEIGE.

1.8.1. Ajax

Ajax por sus siglas en inglés *Asynchronous JavaScript And XML* (JavaScript asíncrono y XML) es la técnica de desarrollo web que se usará para poder hacer consultas asíncronas al servidor sin necesidad de recargar la página. Esta surge de la combinación de tres tecnologías existentes: HTML (o XHTML) y Hojas de Estilo en Cascada (CSS) para presentar la información, Document Object Model (DOM) y JavaScript, para interactuar dinámicamente con los datos, además de XML y XSLT, para intercambiar y manipular datos de manera sincronizada con un servidor web. (18)

Para la implementación del Sistema Control de Flota y Mantenimiento será utilizada para la comunicación entre la capa de la vista y la controladora Ajax, por las ventajas que ofrece:

• Mejora completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, mediante la creación de un elemento intermedio entre el usuario y el servidor. (18)

• Puede ser utilizada en cualquier plataforma o navegador. (18)

• Las peticiones HTTP al servidor se sustituyen por peticiones JavaScript que se realizan al elemento encargado de AJAX. Las más simples no requieren intervención del servidor, por lo que la respuesta es inmediata y en caso contrario la interacción requiere una respuesta del servidor, la petición se realiza de forma asíncrona mediante AJAX. (18)

1.8.2. Lenguajes de programación

Los lenguajes de desarrollo que se emplean para la obtención de la solución que se propone, han sido establecidos por la dirección del proyecto Soluciones Empresariales del centro CEIGE en consecuencia con las tecnologías, herramientas y marco de trabajo definido para la confección de sus productos.

1.8.2.1. Lenguaje de programación PHP

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

Su empleo para el desarrollo del Sistema Control de Flota y Mantenimiento se debe a que el marco de trabajo Sauxe, sobre el cual se implementará el sistema, fue desarrollado con este lenguaje de programación. Además de las muchas ventajas que brinda PHP.

1.8.2.1. Lenguaje de programación Java Script

JavaScript es un lenguaje de programación de alto nivel que se adapta bien a los estilos de programación orientados a objetos y funcional utilizado principalmente para crear páginas web dinámicas, o sea, páginas web que incorporan efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario. Es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlo. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java.

Su empleo para el desarrollo del Sistema Control de Flota y Mantenimiento se debe a las muchas ventajas que brinda JavaScript.

1.8.3. Marco de trabajo

Un framework, en el desarrollo de sistemas computarizados, es una estructura de soporte definida mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. Es definido como un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular, que sirve como referencia para enfrentar y resolver nuevos problemas de índole similar. (20)

1.8.3.1. Marco de trabajo Sauxe

Sauxe es el marco de trabajo que se empleará para el desarrollo de la solución que se propone. Fue desarrollado en la UCI como fruto del paradigma de independencia tecnológica por el que aboga el país. Este marco de trabajo, fusionado bajo tecnología totalmente libre (entre ellas PHP, PostgreSQL, Apache) posee el desarrollo de tecnologías propias basadas en otros marcos de trabajo como Zend Framework para el manejo de la lógica de negocio, Doctrine para el acceso a datos y ExtJS para la capa de presentación. Cuenta con una arquitectura en capas. Contiene un conjunto de componentes reutilizables que provee la estructura genérica y el comportamiento para una familia de abstracciones, logrando una mayor estandarización, flexibilidad, integración y agilidad en el proceso de desarrollo. (21)

1.8.3.2. Marco de trabajo ExtJS 3.4

ExtJS es el marco de trabajo que se empleará para el desarrollo de la capa de presentación. Está basado completamente a la programación orientada a objeto. Cada objeto contiene lo típico: propiedades, métodos y eventos. Basa toda su funcionalidad en JavaScript a través de bibliotecas. Así, en tiempo de ejecución carga y crea todos los objetos HTML a través del uso intenso de DOM. Los datos son obtenidos con AJAX a través de XML. Una de las grandes ventajas de utilizar ExtJS es que permite crear aplicaciones complejas utilizando componentes predefinidos. La carga de procesamiento se distribuye, permitiendo que el servidor al tener menor carga, pueda manejar más clientes al mismo tiempo. (22)

En la implementación del Sistema Control de Flota y Mantenimiento será empleado para la construcción de todas las interfaces de la aplicación debido a las ventajas que ofrece.

1.8.3.3. Marco de trabajo Zend 1.12

Se utilizará el marco de trabajo Zend Framework para el manejo de la lógica de negocio. Este marco de trabajo de código abierto brinda facilidades de uso y poderosas funcionalidades. Está diseñado para la versión 5 de PHP y posee buenas capacidades de ampliación. Proporciona un sistema de caché de forma que se puedan almacenar diferentes datos, así como los componentes que forman la infraestructura del patrón Modelo-Vista-Controlador. Consta de mecanismos de filtrado y validación de entradas de datos. Permite convertir estructuras de datos PHP a JSON y viceversa, para su utilización en aplicaciones AJAX y provee capacidades de búsqueda sobre documentos y contenidos. (23)

1.8.3.4. Marco de trabajo Doctrine 1.2.2

Para la capa de acceso a datos se empleará Doctrine. Es un sistema ORM por sus siglas en inglés Object Relational Mapper (Mapeo Objeto-Relacional) para PHP 5.2 o superior que incorpora una DBL (capa de abstracción a base de datos). Uno de sus rasgos importantes es la habilidad de escribir opcionalmente las consultas de la base de datos orientada a objeto. Esto proporciona una alternativa poderosa a diseñadores de SQL, manteniendo un máximo de flexibilidad sin requerir la duplicación del código innecesario. Además, exporta una base de datos existente a sus clases correspondientes. (24)

1.8.4. Notación de modelado de procesos de negocio

BPMN (*Business Process Modeling Notation, por sus siglas en inglés*) es empleado en el desarrollo de la solución que se propone para presentar gráficamente las diferentes etapas de gestión y optimización de sus procesos de negocio. Este estándar de modelado de procesos de negocio ha sido diseñado específicamente para coordinar la secuencia de procesos y los mensajes que fluyen entre los diferentes procesos participantes. BPMN a diferencia de UML toma un perfil orientado a procesos en el modelado de sistemas, por lo tanto la combinación de ambas notaciones, al ser compatibles entre sí, ayudan a modelar con mayor precisión la situación actual y deseada en los procesos de negocio del cliente. (25)

1.8.5. Lenguaje Unificado de Modelado

UML (*Unified Modeling Language, por sus siglas en inglés*) como notación orientada a objetos cuenta con una notación estándar y semánticas esenciales para el modelado de sistemas, se empleará con el fin de modelar, especificar y documentar un sistema

de software, de un modo estándar incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema. UML implementa un lenguaje de modelado común para todos los desarrollos por lo que se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje de programación utilizado para el desarrollo. (26)

1.9. Herramientas

Las herramientas que se emplean para la obtención de la solución que se propone, han sido establecidas por la dirección del proyecto Soluciones Empresariales del centro CEIGE.

1.9.1. Visual Paradigm 8.0

Se empleará Visual Paradigm 8.0 como herramienta CASE (Computer Aided Software Engineering*, por sus siglas en inglés*). Utiliza UML como lenguaje de modelado, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Ayudando a construir aplicaciones de calidad más rápido, mejor y en bajo costo. Visual Paradigm también permite modelar el negocio orientado a procesos usando la notación BPMN. (27)

1.9.2. Servidor de aplicaciones Apache 2.2

Se utilizará como servidor web Apache 2.2 pues es una tecnología gratuita de código abierto compatible con muchos Sistemas Operativos. Tiene todo el soporte que se necesita para tener páginas dinámicas. Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Tiene una alta capacidad de configuración en la creación y gestión de registros de actividad. Apache permite la creación de ficheros de registro a medida del administrador, de este modo se puede tener un mayor control sobre lo que sucede en el servidor. (28)

1.9.3. Sistema gestor de base de datos

Los Sistemas Gestores de Bases de Datos (SGBD) se emplean para ejecutar consultas complejas y uniones de gran tamaño. Permiten la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Además soporta

transacciones, claves ajenas con comprobaciones de integridad referencial y almacenamiento de objetos de gran tamaño. (29)

PostgreSQL 9.1

Como Sistema Gestor de Base de Datos (SGBD) se empleará la versión 9.1 de PostgreSQL. Este es un sistema de gestión de bases de datos relacional orientada a objetos. Es una herramienta de código abierto, de bajo coste y multiplataforma. Se destaca en ejecutar consultas complejas y uniones de gran tamaño. Permite la definición de tipos de datos personalizados e incluye un modelo de seguridad completo. Soporta transacciones, claves ajenas con comprobaciones de integridad referencial y almacenamiento de objetos de gran tamaño. (30)

1.9.4. Navegador

Un navegador web es un tipo de software que permite la visualización de documentos y sitios en hipertexto, comúnmente agrupados bajo la denominación de Web o Internet. (31)

Mozilla Firefox 10.6

Firefox es un navegador de Internet con interfaz gráfica de usuario desarrollado por la Corporación Mozilla y un gran número de voluntarios. Incluye entre sus funcionalidades un mecanismo para añadir funcionalidades mediante extensiones. Precisamente Firebug es el complemento que se integrará a Mozilla Firefox para ayudar a desarrollar, evaluar y depurar la aplicación, controlando el CSS y HTML en tiempo real, midiendo el tiempo de carga para optimizar la página o corrigiendo los posibles inconvenientes con JavaScript. Es un navegador de código libre y multiplataforma. Además permite la identificación de sitios web de forma instantánea, posibilita navegar de forma privada y sin registro de lo accedido en internet. (32)

1.10. Métricas para la validación del diseño

Aunque los términos medida, medición y métricas se utilizan a menudo indistintamente, existe gran confusión a la hora de referirse a ellos. Dentro del contexto de la ingeniería del software, una medida proporciona una indicación cuantitativa de extensión, cantidad, dimensiones, capacidad y tamaño de algunos atributos de un proceso o producto. La medición es el proceso por el cual los números o símbolos son asignados a atributos o entidades en el mundo real tal como son descritos de acuerdo a reglas claramente definidas, también es definida como el acto de determinar una medida. (33)

Las métricas permiten descubrir y corregir problemas potenciales antes de convertirse en defectos catastróficos. Se emplean con el objetivo de llevar un control de la calidad del producto que se está desarrollando, evaluar la efectividad del proceso y mejorar la calidad del trabajo.

1.10.1. Métricas propuestas por Lorenz y Kidd

Lorenz y Kidd dividen las métricas basadas en clases en cuatro categorías: tamaño, herencia, valores internos y valores externos. Las métricas orientadas a tamaños para una clase se centran en cálculos de atributos y de operaciones para una clase individual, para luego promediar los valores para el sistema en su totalidad. Las métricas basadas en herencia se centran en la forma en que se reutilizan las operaciones en la jerarquía de clases. Las métricas para valores internos de clase examinan la cohesión y asuntos relacionados con el código así como las métricas orientadas a valores externos examinan el acoplamiento y la reutilización. (34)

1.11. Pruebas de software

Las pruebas del software son un elemento crítico para garantizar la calidad de un software y representa una visión final de las especificaciones, del diseño y de la codificación. Una vez generado el código fuente, el software debe ser probado para descubrir y corregir, el máximo de errores posibles antes de su entrega al cliente. (35)
Glen Myers en su libro estableció varias normas que pueden servir acertadamente como objetivos de las pruebas, estas son:

•La prueba es el proceso de ejecución de un programa con la intención de descubrir un error.

•Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces.

•Una prueba tiene éxito si descubre un error no detectado hasta entonces.

•Los datos que se van recogiendo a medida que se lleva a cabo la prueba proporcionan una buena indicación de la fiabilidad del software y de alguna manera, indican la calidad del software como un todo. (35)

Para realizarle las pruebas al subsistema Control de Neumáticos y Baterías, con el objetivo de descubrir y corregir errores existentes en este, se emplean las pruebas de caja blanca y caja negra.

1.11.1. Pruebas de caja blanca

La prueba de caja blanca, denominada a veces prueba de caja de cristal es un método de diseño de casos de prueba que utiliza la estructura de control del diseño

procedimental para obtener los casos de prueba. (35) La prueba de caja blanca que se aplicará a la solución desarrollada será la Prueba del Camino Básico, que permite obtener una medida de la complejidad lógica del diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución, garantizando con estos que durante la prueba se ejecute por lo menos una vez cada sentencia del programa. (35)

1.11.2. Pruebas de caja negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. O sea, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. (35)Los métodos de caja negra que se utilizaran para asegurar la calidad de la aplicación desarrollada serán:

Partición de equivalencia

La partición de equivalencia es un método de prueba de caja negra que divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. Un caso de prueba ideal descubre de forma inmediata una clase de errores que de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico. (35)

Análisis de valores límites

Los errores tienden a darse más en los límites del campo de entrada que en el centro. Por ello, se ha desarrollado el análisis de valores límite (AVL) como técnica de prueba. El análisis de valores límite es una técnica de diseño de casos de prueba que complementa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de las clases. En lugar de centrarse solamente en las condiciones de entrada, el AVL obtiene casos de prueba también para el campo de salida. (35)

Conclusiones parciales del capitulo

Luego de realizar un análisis de las soluciones informáticas para el control de flotas y mantenimiento nacional e internacional, se evidencia la necesidad de desarrollar un nuevo sistema para la informatización de los procesos del área de transporte, específicamente el proceso de Control de Neumáticos y Baterías, que cumpla con los principios de soberanía tecnológico del país. También se caracterizaron los lenguajes de modelado, programación y las tecnologías con las que se desarrollará el proceso

de acuerdo con el Modelo de desarrollo definido por el centro CEIGE, lo que permite conocer las principales características y ventajas que estos proporcionan. Asimismo se realizó una valoración del modelo de desarrollo del centro CEIGE. Además se caracterizaron las métricas para la validación del diseño y las pruebas de software.

Capítulo 2. Modelado del negocio y requerimientos

Introducción

Este capítulo describe la solución propuesta, mostrando características específicas del sistema mediante el modelado y descripción de los procesos de negocio; se especifican los requisitos funcionales definidos para el proceso de Control de Neumáticos y Baterías así como las técnicas de captura y validación de requisitos utilizadas. Además se identifican y se registran en el modelo conceptual los conceptos fundamentales que se manejan en el área de Transporte de la UCI en cuanto a la gestión del proceso Control de Neumáticos y Baterías.

2.1. Modelado de procesos

El modelado de procesos de negocio es la representación de los procesos de negocio de una empresa u organización con objeto de que puedan ser analizados y mejorados. (36) Su elaboración brinda una serie de beneficios a las organizaciones permitiéndoles reutilizar procesos que sean más eficientes, ventajosos y productivos, detectar tareas que no puedan ser realizadas, así como mejorar de forma general los procesos que se realizan.

La confección de los modelos de procesos de negocio es realizada principalmente por los analistas, labor que les brinda una serie de beneficios como: la agilización del proceso de desarrollo y de la carga de trabajo, la identificación de errores en las fases tempranas, mayor nivel de abstracción, además de la trazabilidad del sistema por identificación de tareas y su asignación a procedimientos manuales o automatizados.

2.2. Mapa de procesos

Un mapa de procesos es un diagrama que muestra, de manera visual, los procesos que se desarrollan en una organización, así como las relaciones existentes entre ellos. Su utilización permite organizar y documentar los procesos con el fin de facilitar su descripción y entendimiento, en su elaboración es necesario tener presente que la expresión gráfica debe ser clara y precisa.

Para la realización de los mapas de procesos del Sistema Control de Flota y Mantenimiento fue empleada la plantilla conformada por el CEIGE, en la cual se representan los procesos identificados en la Dirección de Transporte de la UCI agrupados por niveles, colocando en un nivel 0 los procesos claves de esta organización y un nivel 1 sus procesos derivados (consultar documento entregable CIG-CFM-N-MTTO-i1101 recogido en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE). El contenido de este trabajo se enmarca en el proceso Control de Neumáticos y Baterías del área de Transporte de la UCI

perteneciente al nivel 1 del proceso Disponibilidad del parque automotor. Se identifican para cada uno de los procesos del nivel 1 los artefactos de entrada y salida involucrados en el proceso y el formato en que se manejan actualmente en el área de Transporte de la UCI.

La matriz de procesos contiene la relación que existe entre los procesos, sistemas o involucrados en el proceso, exponiendo los artefactos de salida de un proceso que constituyen entradas para otros y viceversa (consultar Tabla 2 y documento entregable CIG-CFM-N-MTTO-i1102 recogido en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE).

Tabla 2 Fragmento matriz de relación entre los procesos de negocio de nivel 1

Entrada/Salida	Control de Vehículos	Asignación de Vehículos	Control de Hoja de Ruta	Control de Neumáticos y Baterías
Control de Vehículos	Expediente del vehículo	Expediente del vehículo		
Asignación de Vehículos	Modelo de inspección técnica			
Control de Hoja de Ruta	Tarjeta de Combustible			
Control de Neumáticos y Baterías	Orden de trabajo			

2.3. Descripción de procesos de negocio

El proceso Control de Neumáticos y Baterías del área de Transporte de la UCI tiene como objetivo principal registrar toda la información relacionada con los neumáticos y baterías. Este proceso comienza con la generación de órdenes de trabajo. Luego se realiza solicitud de insumo al almacén y se verifica si es una batería o un neumático para en función de esto registrar los datos en los documentos de controles correspondientes, dígase Control de neumáticos o Control de baterías. Posteriormente se actualiza la Orden de trabajo y en caso que se le vaya a dar baja al insumo solicitado se debe verificar su régimen de durabilidad con el objetivo de verificar si este llega al régimen definido por el fabricante. Finalmente se le da baja al insumo.

Tabla 3 Artefactos de entrada y salida del proceso de Control de Neumáticos y Baterías 1

Artefactos de Entrada	Artefactos de Salida
Orden de trabajo	Solicitud de Insumo
	Control de baterías
	Control de neumáticos
	Orden de trabajo

Para obtener mayor información sobre la modelación de este proceso de negocio consultar documento entregable CIG-CFM-N-MTTO-i1503 recogido en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

2.4. Validación del modelado de negocio

La validación del modelado de procesos de negocio se realizó aplicando la técnica Revisión Técnica Formal, donde los clientes revisaron cada documento de descripción de procesos de negocio realizado con el objetivo de validar su correcta elaboración y que el flujo de actividades de cada proceso estuviera en correspondencia con la información brindada. Además se realizaron diversas reuniones con los especialistas del área de Transporte de la UCI y miembros del equipo de desarrollo para logar una correcta interpretación de la información transmitida con el objetivo de corregir cualquier inconformidad por parte de los clientes, los señalamientos planteados fueron recogidos y corregidos posteriormente. Como constancia de esta revisión se emite un Acta de aceptación recogida en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

2.5. Modelo conceptual

Un modelo conceptual se utiliza para describir esquemas conceptuales. Es un diagrama conformado con los conceptos que son significativos para el área que se analice, así como las relaciones existentes entre ellos. El objetivo del diagrama conceptual es describir el contenido de la información de la base de datos y no las estructuras de almacenamiento que se necesitarán para manejar esta información. Permite además identificar, organizar y realizar razonamientos sobre los componentes y comportamientos del sistema, siendo la guía para el proceso de diseño del software, pudiéndose usar además como referencia para evaluar un diseño particular y razonar sobre la solución realizada. El modelo conceptual del proceso Control de Neumáticos y Baterías fue confeccionado con todas las clases conceptuales identificadas en este proceso junto con los atributos y relaciones existentes entre las mismas (consultar Figura 2 y documento entregable CG-SM-DE-002 recogido en el repositorio del

proyecto de Soluciones Empresariales del centro CEIGE). A continuación se observa un fragmento del modelo conceptual del negocio con sus conceptos más importantes.

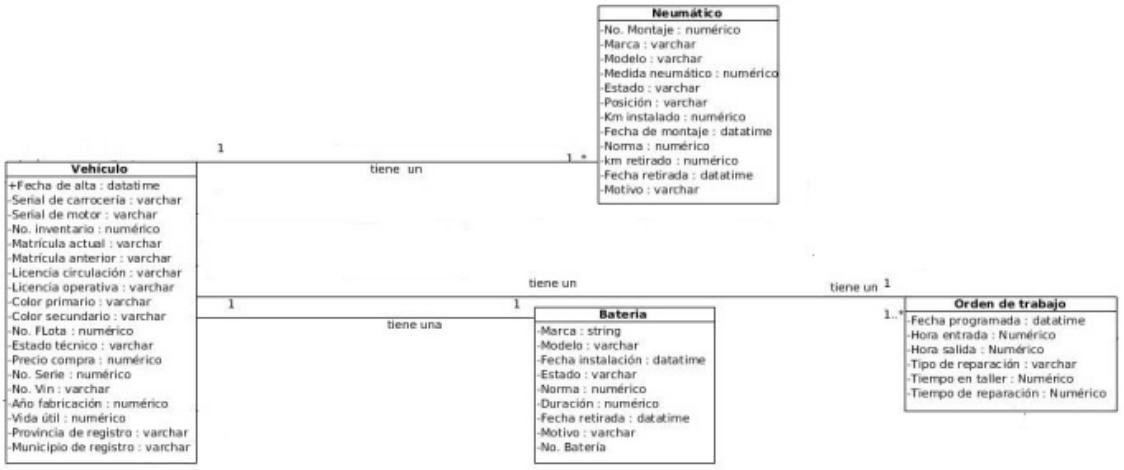


Figura 2 Modelo Conceptual

2.6. Requerimientos de software

Un requerimiento es simplemente una declaración abstracta de alto nivel de un servicio que debe proporcionar el sistema o una restricción de este. (37) Pueden dividirse en dos categorías: requerimientos funcionales y requerimientos no funcionales. Los requerimientos funcionales son los que definen las funciones que el sistema será capaz de realizar, describen las transformaciones que el sistema realiza sobre las entradas para producir salidas. Es importante que se describa el ¿Qué? y no el ¿Cómo? se deben hacer esas transformaciones. (38)

La captura de requisitos es la actividad mediante la cual el equipo de desarrollo de un sistema de software extrae, de cualquier fuente de información disponible, las necesidades que debe cubrir dicho sistema. El proceso de captura de requisitos puede resultar complejo, principalmente si el entorno de trabajo es desconocido para el equipo de analistas, y depende mucho de las personas que participen en él. Por la complejidad que todo esto puede implicar, la ingeniería de requisitos ha trabajado desde hace años en desarrollar técnicas que permitan hacer este proceso de una forma más eficiente y precisa. (39)

Antes de comenzar el proceso de captura de los requisitos del proceso Control de Neumáticos y Baterías se identificaron los proveedores válidos de requisitos teniendo en cuenta algunos criterios como: conocimiento del negocio y del mercado, disponibilidad de tiempo, habilidades de comunicación, conocimiento de informática, compromiso con el proyecto(consultar documento CIG-CFM-N-MTTO-i2401 recogido en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE)

en aras de identificar a las personas con mayor dominio del proceso en el área de Transporte de la UCI y así obtener los requisitos a implementar.

2.6.1. Técnicas para la captura de requisitos

Se aplicaron las siguientes técnicas para la captura de los requerimientos:

Entrevista: A través de encuentros planificados con los proveedores de requisitos se realizaron preguntas relacionadas con el proceso Control de Neumáticos y Baterías en función de obtener las necesidades que tenían de informatización. Esto proporcionó como resultado que fueran identificados los requerimientos por los que se guiaría el desarrollo de las funciones, a través del cual se realizaría la gestión de este proceso. Dichas entrevistas fueron registradas y recogidas en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

Desarrollo conjunto de aplicaciones del inglés Joint Application Development (JAD): El equipo de desarrollo junto con los proveedores de requisitos, fueron revisando la documentación de los requisitos a medida que iban siendo identificados, con el objetivo de chequear que las especificaciones realizadas sobre el proceso Control de Neumáticos y Baterías satisficieran las necesidades de informatización que poseían los proveedores de requisitos.

Lluvia de ideas: En cada encuentro fueron debatidos los requisitos identificados, fluyendo variadas ideas acerca de los mismos, lo cual arrojó como resultado que se obtuviera una visión más amplia de lo que se quería implementar, favoreciéndose de esta forma el avance de futuras etapas en el desarrollo del módulo, a través del cual se llevaría a cabo la gestión del proceso Control de Neumáticos y Baterías.

2.6.2. Requisitos funcionales

Los requisitos funcionales identificados del proceso Control de Neumáticos y Baterías del área de Transporte de la UCI son:

RF 1. Agrupación de requisitos: Gestionar tipos baterías

- RF 1.1 Adicionar tipo batería
- RF 1.2 Modificar tipo batería
- RF 1.3 Eliminar tipo batería
- RF 1.4 Listar tipo batería
- RF 1.5 Buscar tipo batería

RF 2. Agrupación de requisitos: Gestionar tipos neumáticos

- RF 2.1 Adicionar tipo neumático

- RF 2.2 Modificar tipo neumático
- RF 2.3 Eliminar tipo neumático
- RF 2.4 Listar tipo neumático
- RF 2.5 Buscar tipo neumático

RF 3. Agrupación de requisitos: Control de baterías.

- RF 3.1 Asociar batería
- RF 3.2 Eliminar batería
- RF 3.3 Listar batería
- RF 3.4 Buscar batería

RF 4. Agrupación de requisitos: Control de neumáticos

- RF 4.1 Asociar neumático
- RF 4.2 Eliminar neumático
- RF 4.3 Listar neumático
- RF 4.4 Buscar neumático

RF 5. Cambio de baterías

RF 6. Agrupación de requisitos: Cambio de neumáticos

- RF 6.1 Listar neumático
- RF 6.1 Cambio neumático

RF 7. Reporte cambio de batería por vehículo

RF 8. Reporte cambio de neumático por vehículo

RF 9. Reporte cambio de batería por tipo de vehículo

RF 10. Reporte cambio de neumático por tipo de vehículo

RF 11. Reporte cambio de batería por período

RF 12. Reporte cambio de neumático por período

2.6.3. Especificaciones de los requisitos funcionales

La especificación de los requisitos identificados se realizó siguiendo los pasos descritos en la plantilla conformada por el CEIGE. A continuación tenemos la especificación del requisito Asociar batería perteneciente a la Agrupación de Requisitos Control de Batería.

Precondiciones	El usuario ha sido validado. Se ha registrado al menos un batería en el sistema.
Flujo de eventos	

Flujo básico		
1	Se insertan los criterios de búsqueda: No. Batería Marca/ Modelo Matrícula	
2	El sistema muestra un listado de los baterías que cumplen con los criterios de búsqueda especificados. Se muestran los siguientes atributos: No. Batería Marca Modelo Fecha. Inst. Matrícula	
3	Concluye el requisito.	
Pos-condiciones		
1	NA	
Flujos alternativos		
Flujo alternativo 2.a No existen datos que cumplan con los criterios especificados.		
1	El sistema notifica que no existen datos que cumplan con los criterios especificados.	
Pos-condiciones		
1	NA	
Validaciones		
1	NA	
Relaciones	**Requisitos Incluidos**	NA
	Extensiones	NA
Conceptos	**Batería**	**Visibles en la interfaz:** No. Batería Marca Modelo Fecha. Inst. Matrícula **Utilizados internamente:** NA
Requisitos especiales	NA	
Asuntos pendientes	NA	

Prototipo no funcional

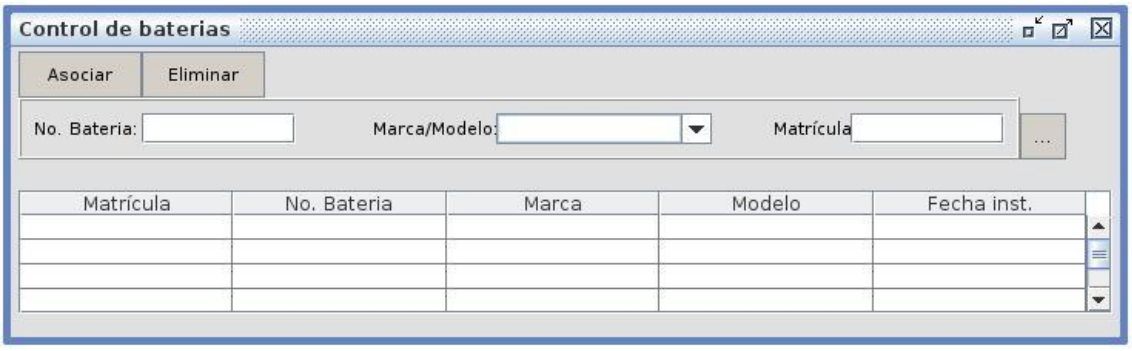


Figura 3 Control de batería

Los restantes requisitos funcionales del proceso Control de Neumáticos y Baterías se encuentran especificados en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

2.6.4. Administración de los requisitos

La administración de los requisitos es un enfoque sistemático para obtener, organizar y documentar los mismos, así como mantener un acuerdo entre el cliente y el equipo del proyecto en los posibles cambios. La clave para una efectiva administración de requisitos es: Mantener un enunciado claro, junto con los atributos para cada tipo de requisitos y su seguimiento con otros o con elementos del proyecto. (40)

Para la administración de los requisitos del proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento para la Dirección de Transporte de la UCI se utilizó la herramienta Visual Paradigm, para ello se definieron los siguientes elementos de trazabilidad:

- Requisitos
- Modelo conceptual (entidades del negocio)
- Componentes
- Diseños de casos de pruebas

Se realizaron las siguientes matrices de trazabilidad

- Requisitos Modelo Conceptual (entidades del negocio)
- Requisitos Componente
- Requisitos Diseños de casos de pruebas

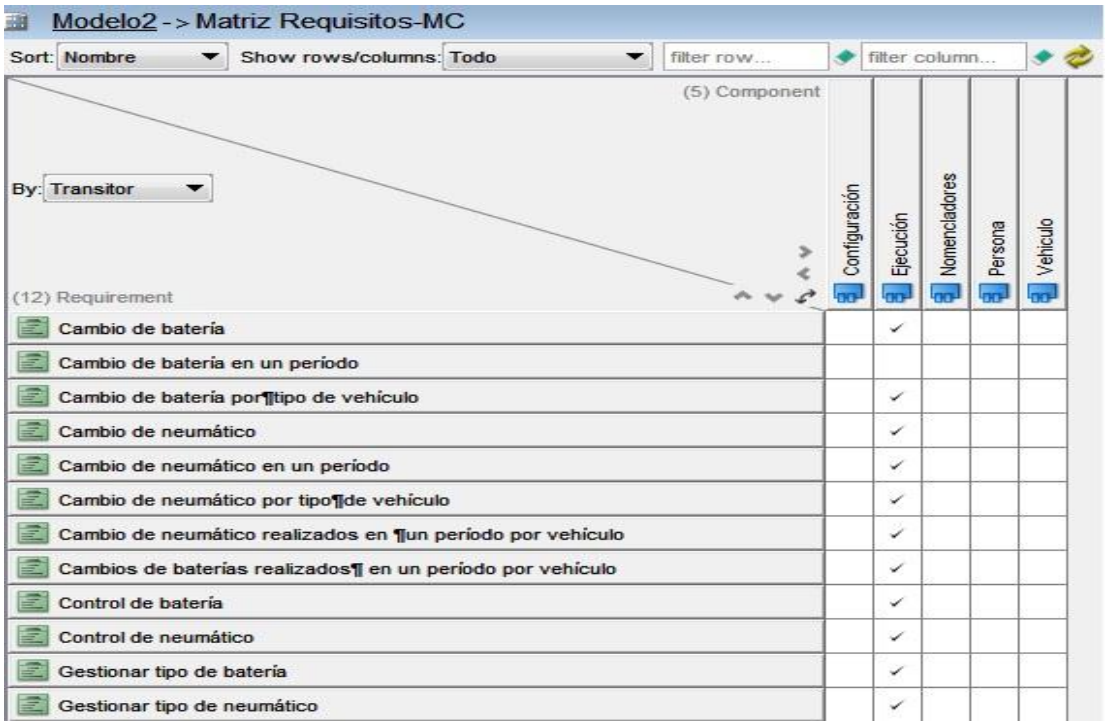
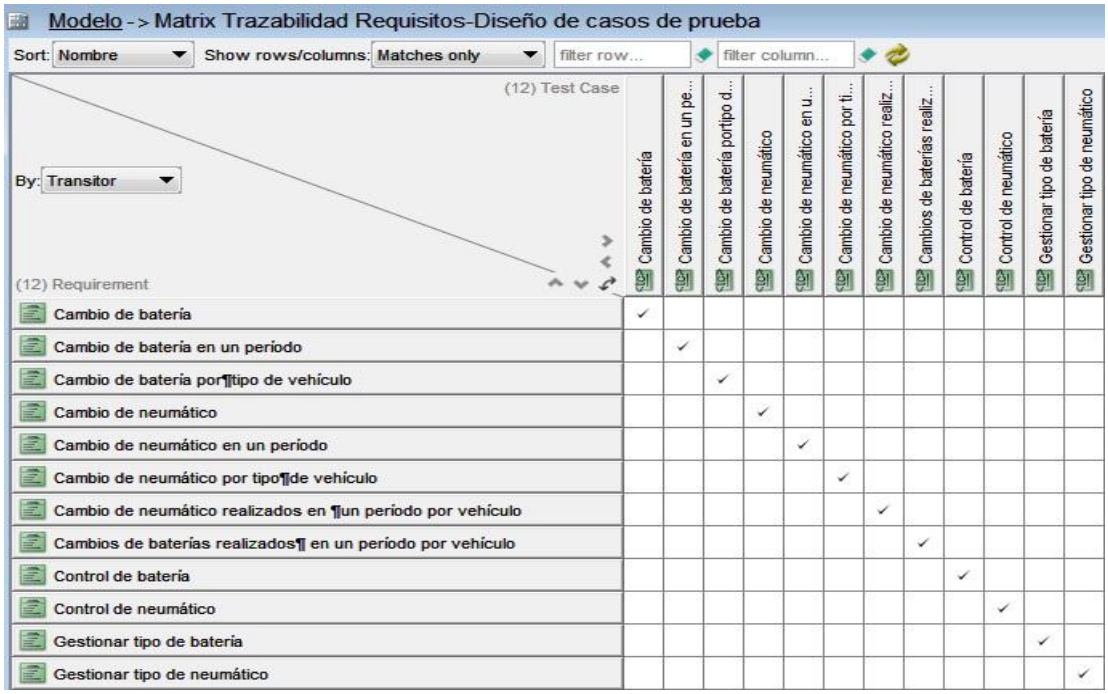


Figura 4 Matriz de Requisitos Componentes



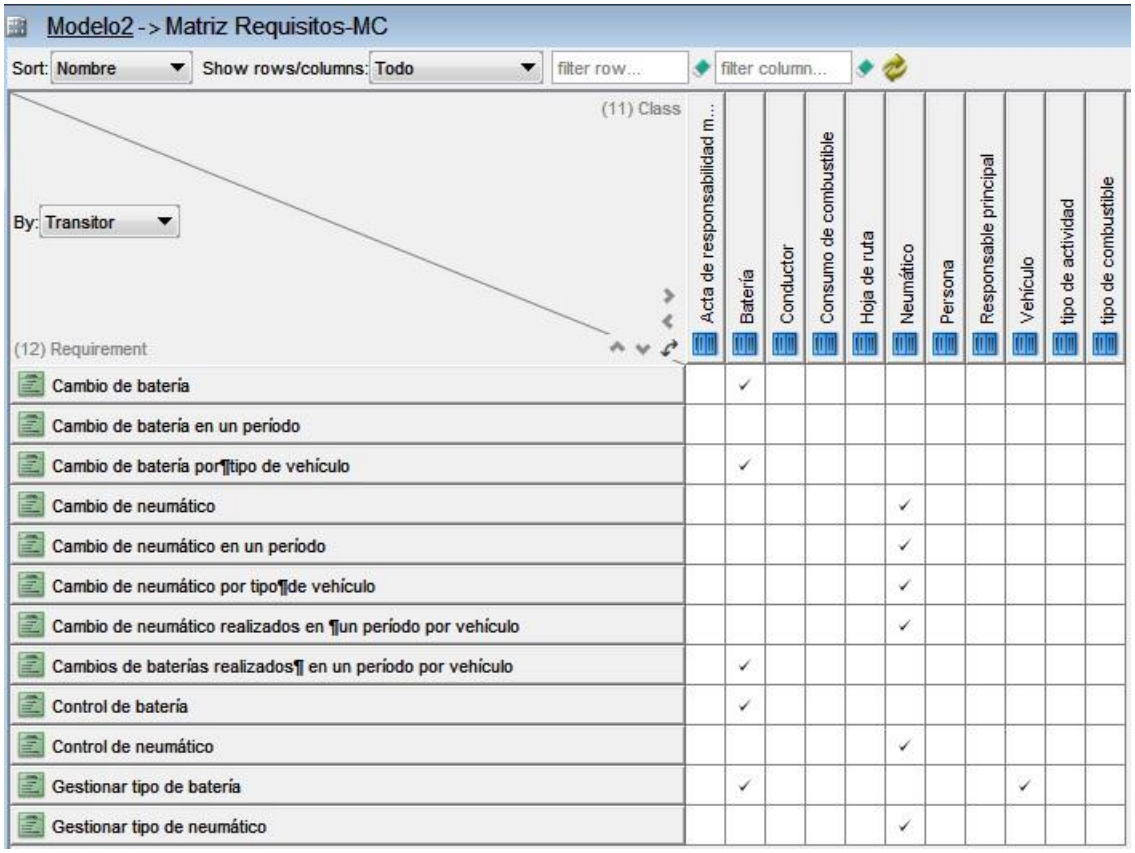
Modelo -> Matrix Trazabilidad Requisitos-Diseño de casos de prueba

Sort: Nombre | Show rows/columns: Matches only | filter row... | filter column...

By: Transitor

(12) Requirement \ (12) Test Case	Cambio de batería	Cambio de batería en un pe...	Cambio de batería por tipo d...	Cambio de neumático	Cambio de neumático en u...	Cambio de neumático por ti...	Cambio de neumático realiz...	Cambios de baterías realiz...	Control de batería	Control de neumático	Gestionar tipo de batería	Gestionar tipo de neumático
Cambio de batería	✓											
Cambio de batería en un período		✓										
Cambio de batería por¶tipo de vehículo			✓									
Cambio de neumático				✓								
Cambio de neumático en un período					✓							
Cambio de neumático por tipo¶de vehículo						✓						
Cambio de neumático realizados en ¶un período por vehículo							✓					
Cambios de baterías realizados¶ en un período por vehículo								✓				
Control de batería									✓			
Control de neumático										✓		
Gestionar tipo de batería											✓	
Gestionar tipo de neumático												✓

Figura 5 Matriz de Requisitos Diseño de Casos de Pruebas



Modelo2 -> Matriz Requisitos-MC

Sort: Nombre | Show rows/columns: Todo | filter row... | filter column...

By: Transitor

(12) Requirement \ (11) Class	Acta de responsabilidad m...	Batería	Conductor	Consumo de combustible	Hoja de ruta	Neumático	Persona	Responsable principal	Vehículo	tipo de actividad	tipo de combustible
Cambio de batería		✓									
Cambio de batería en un período											
Cambio de batería por¶tipo de vehículo		✓									
Cambio de neumático						✓					
Cambio de neumático en un período						✓					
Cambio de neumático por tipo¶de vehículo						✓					
Cambio de neumático realizados en ¶un período por vehículo						✓					
Cambios de baterías realizados¶ en un período por vehículo		✓									
Control de batería		✓									
Control de neumático						✓					
Gestionar tipo de batería		✓							✓		
Gestionar tipo de neumático						✓					

Figura 6 Matriz de Requisitos Modelo Conceptual

2.6.5. Priorización de requisitos

Para la determinación de la complejidad de los requisitos se analizan individualmente los criterios siguientes, llegando al resultado de si el requisito es de complejidad Alta, Media o Baja. La clasificación de la complejidad permite estimar el esfuerzo de implementación del requisito y contribuye a la decisión sobre la inclusión en las etapas de desarrollo del software. La priorización de los requisitos identificados para el proceso Control de Neumáticos y Baterías se realizó mediante la plantilla Evaluación de requisitos (consultar documento entregable CIG-CFM-N-MTTO-i6101 recogido en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE) realizada por el CEIGE y teniendo en cuenta los criterios definidos para determinar la complejidad de los requisitos.

Indicadores definidos:

- **Diferentes comportamientos**
- **Formas de inicialización**
- **Consultas a fuentes de almacenamientos**
- **Restricciones de validación**
- **Grado de reutilización**
- **Lógica de negocio**

La siguiente tabla muestra un grupo de requisitos funcionales identificados del proceso Control de Neumáticos y Baterías con su respectivo grado de complejidad.

Tabla 4 Artefactos de entrada y salida del proceso Control de Neumáticos y Baterías

Requisitos/Complejidad	Alta	Media	Baja
Cambio de neumáticos	X		
Control de neumáticos		X	
Gestionar tipos de neumáticos			X
Gestionar tipo de baterías			X
Control de baterías	X		
Cambio de baterías	X		
Cambio de neumático por tipo de vehículo	X		
Cambio de baterías en un periodo	X		
Cambio de baterías por vehículo	X		

Cambio de baterías por tipo de vehículo	**X**
Cambio de neumáticos periodo	**X**
Cambio de neumáticos por vehículo	**X**

2.6.6. Validación de requisitos

La validación de los requisitos se realiza con la finalidad de comprobar que los requisitos identificados sean precisos, consistentes, realistas, verificables, definan lo que el usuario desea del producto final, que los errores que hayan sido detectados sean corregidos y el resultado del trabajo cumpla con los estándares establecidos para el proceso, el proyecto y el producto. Como constancia de esta revisión se emite un Acta de aceptación recogida en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

Para la validación de los requisitos fueron aplicadas las siguientes técnicas:

Revisión técnica formal: La revisión de las especificaciones de los requerimientos del proceso Control de Neumáticos y Baterías fue realizada con los proveedores de requisitos. Con la utilización de esta técnica se validó que no existieran errores en el contenido o malas interpretaciones, información incompleta, inconsistencias y que los requisitos no fueran contradictorios, imposibles o inalcanzables, dando como resultado que fueran aprobados los que estaban descritos de forma correcta, clara y consistente.

Prototipos: A partir de las especificaciones de requisitos fueron conformados prototipos de interfaz de usuario. Con esto se validó que los requerimientos estaban en concordancia con las necesidades plasmadas por los proveedores de requisitos. El empleo de esta técnica ofreció como resultados que el cliente tuviera una idea de la estructura de la interfaz de usuario y se favoreciera la comunicación con el mismo, ya que tenía una visión inicial de las funcionalidades través de las cuales se gestionaría el proceso Control de Neumáticos y Baterías.

Generación de casos de prueba (test de requisitos): Fueron definidos y diseñados casos de pruebas para cada requerimiento especificado, con el objetivo de verificar el cumplimiento de los mismos. La utilización de esta técnica ofreció los siguientes resultados: fueron identificados los posibles escenarios de los requisitos, así como los juegos de datos de los campos determinados en estos, se validaron y aprobaron los que estaban bien enunciados, descritos y consistentes.

2.7. Requisitos no funcionales

Un requisito no funcional o atributo de calidad es un requisito que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos. Se refiere a todos los requisitos que ni describen información a guardar, ni funciones a realizar.

Entre los requisitos no funcionales identificados en el Sistema Control de Flota y Mantenimiento están:

RNF 1. Usabilidad

1. El idioma de todas las interfaces de la aplicación será el español.
2. La salva de información se hará sólo cuando la información básica del concepto asociado esté completa, de no estarlo el usuario recibirá una notificación de que son necesarios dichos datos y no se continuará el flujo.
3. No se utilizarán textos para las etiquetas de la interfaz de usuario. Los íconos previstos para dichas actividades quedarán plasmados en el manual de pautas de diseño.
4. Los errores cometidos por el usuario les serán notificados y los mensajes incluirán sugerencias de las posibles soluciones.

RNF 2. Seguridad

1. El sistema manejará la seguridad de acceso y administración de usuarios mediante el otorgamiento de privilegios y roles, asignación de perfiles.
2. Se concederá acceso al sistema a partir de un nombre de usuario y una contraseña.
3. El sistema concederá acceso a cada usuario autenticado solo a las funciones que le estén permitidas, de acuerdo a la configuración del sistema.

RNF 3. Fiabilidad

1. El sistema estará disponible durante 8 horas, los 7 días de la semana, los 365 días del año.
2. El sistema tendrá un respaldo de la información diaria, permitiendo la recuperación ante la pérdida parcial o total de la información.

Conclusiones parciales del capitulo

Con la modelación del negocio del proceso Control de Neumáticos y Baterías, fueron generados una serie de artefactos que crearon las condiciones para la captura de los requisitos funcionales de este proceso, los cuales fueron identificados, especificados y validados, garantizando corregir errores que pudieran afectar futuras etapas del desarrollo del Sistema Control de Flota y Mantenimiento, dando paso así, a la realización del diseño de la propuesta de solución.

Capítulo 3 Diseño, implementación y pruebas de la propuesta de solución

Introducción

En este capítulo se aborda inicialmente sobre el diseño de la propuesta de solución, son expuestos los mecanismos de diseño utilizados, los diagramas de clases conformados a partir de lo definido en el análisis y los patrones empleados en la modelación. A través de métricas es validado el diseño propuesto, en aras de comprobar su correcta elaboración. Se muestra el modelo de datos, diagrama de componentes y el modelo de despliegue realizado para el sistema. Se tocan los puntos más importantes en la implementación de las funcionalidades a partir del cual se realizaría la gestión del proceso Control de Neumáticos y Baterías del área de Transporte de la UCI. Se describen los estándares de codificación empleados para una mayor legibilidad del código. Para finalizar son especificadas las pruebas de software a utilizar para validar que el sistema desarrollado cumpla con los requisitos aceptados por el cliente.

3.1. Diseño

El diseño es el sitio donde manda la creatividad, donde los requisitos del cliente, las necesidades de negocio y las consideraciones técnicas se unen en la formulación de un producto o sistema. Crea una representación o modelo del software, pero a diferencia del modelo de análisis, el modelo de diseño proporciona detalles acerca de las estructuras de datos, las arquitecturas, las interfaces y los componentes del software que son necesarios para implementar el sistema. (35)

El diseño de software permite la producción de modelos del sistema que se va a construir. Estos modelos pueden evaluarse en relación con su calidad y mejorarse antes de generar código, de realizar pruebas y de que los usuarios finales se vean involucrados a gran escala.

3.1.1. Mecanismos de diseño

Los mecanismos de diseño se utilizan con el objetivo de simplificar los diagramas de clases. Cada diseñador acorde a sus necesidades establece sus propios mecanismos de diseño, teniendo en cuenta los patrones y estilos seleccionados. (41) En el componente Taller en el que están incluidas las funcionalidades del proceso Control de Neumáticos y Baterías se definieron los siguientes mecanismos:

3.1.1.1. Mecanismos de diseño construcción y actualización de páginas clientes

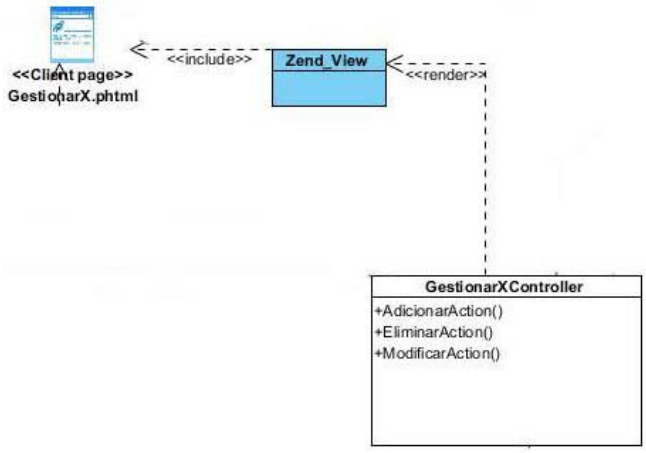


Figura 7 Mecanismo de diseño construcción y actualización de páginas clientes

Este mecanismo se utiliza siempre que se necesite construir (build) o actualizar una nueva página cliente producto de una respuesta del sistema. (42)

GestionarXControler: utiliza la Clase Zend_View para dibujar el contenido de la página clientes GestionarX.phtml. (42)

Zend_View: Incluye las páginas clientes y construye o actualiza la interfaz de las páginas clientes con la información del controlador. (42)

GestionarX.phtml: página cliente que debe ser mostrada o actualizada como respuesta del sistema. (42)

3.1.1.2. Mecanismos de diseño para las clases controladoras

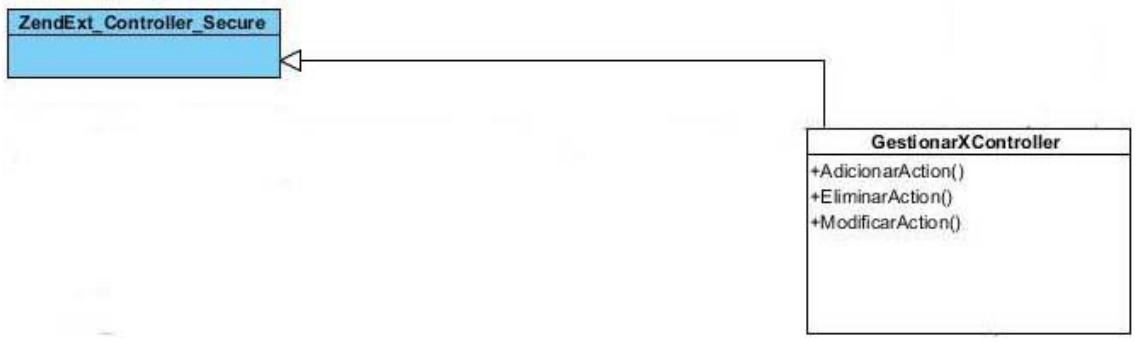


Figura 8 Mecanismo de diseño para las clases controladoras

Todas las clases controladoras definidas en el diseño propuesto heredan de la clase ZendExt_Controller_Secure, ya que en ella se incluyen numerosas funcionalidades comunes en todas las controladoras. (42)

3.1.1.3. Mecanismos de diseño inversión de control para el consumo de servicios entre componentes desde las clases controladoras y modelos

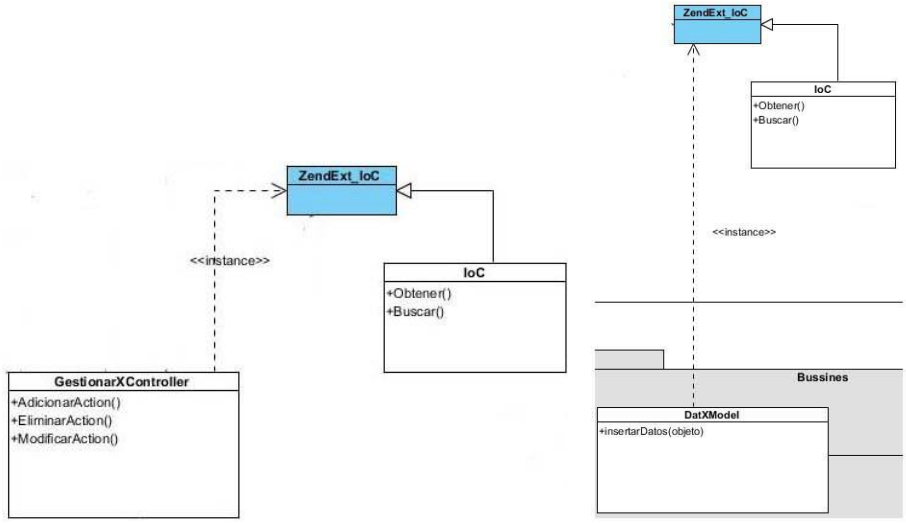


Figura 9 Mecanismo de diseño inversión de control para el consumo de servicios entre componentes desde las clases controladoras y modelos

Siempre que se necesite consumir un servicio de otro componente desde una clase controladora o de modelo se instanciará la clase IoC la cual hereda de ZendExt_IoC que permite tener acceso a los servicios de todos los componentes del sistema Control de Flota y Mantenimiento. (42)

3.1.1.4. Mecanismos de diseño para las clases modelos

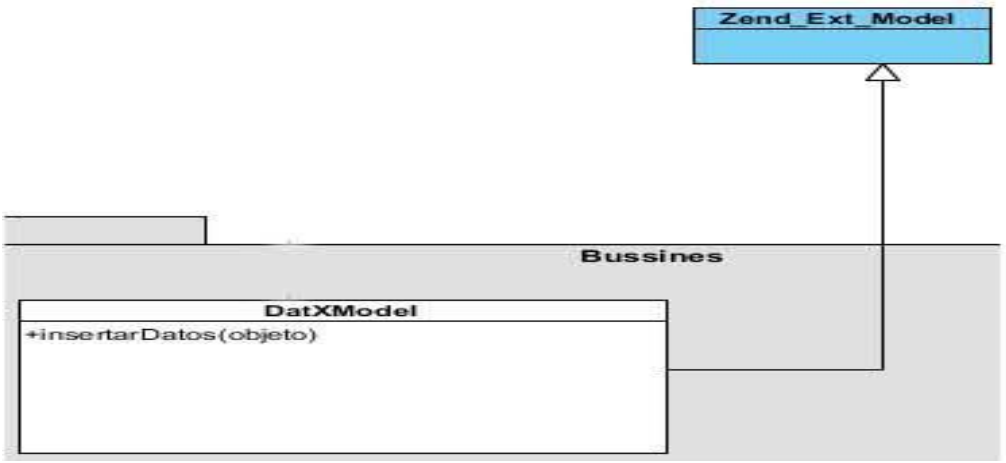


Figura 10 Mecanismo de diseño para las clases modelos

Todas las clases modelos o model definidas en el diseño heredan de la clase ZendExt_Model, ya que esta incluye las principales funciones para el manejo de los datos. (42)

3.1.1.5. Mecanismos de diseño para las clases del dominio

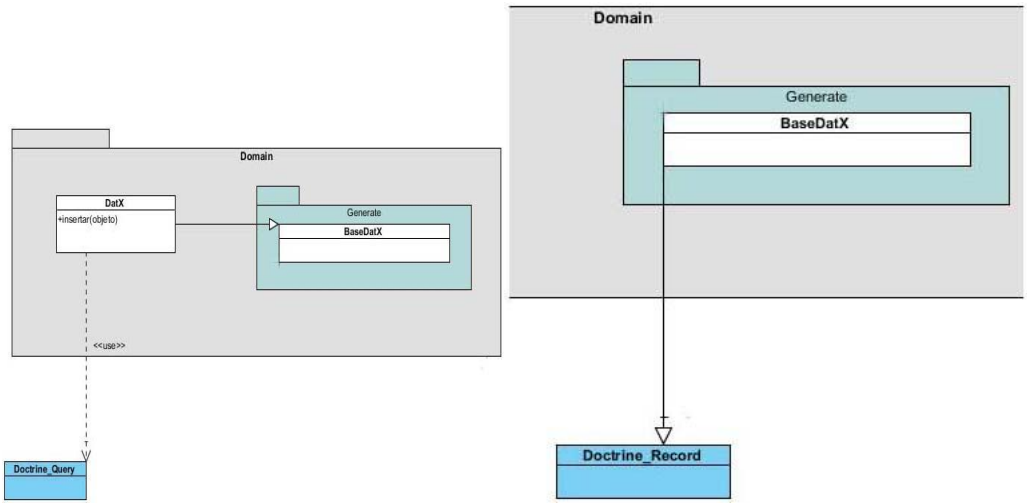


Figura 11 Mecanismo de diseño para las clases del dominio

La clase DatX representa a las clases del dominio (Domain), que usan a la Clase Doctrine_Query del marco de trabajo Doctrine, para el acceso a la base de datos a través del lenguaje DQL; estás clase además heredan de las clases cuyo prefijo es **Base** con los atributos mapeadas de las tablas de la base de datos; en este caso BaseDatX. Todas las clases con el prefijo Base como BaseDatX heredan de Doctrine_Record que permite agrupar en registros u objetos mapeados los datos de las tablas. (42)

3.1.1.6. Mecanismo de diseño control, negocio y dominio

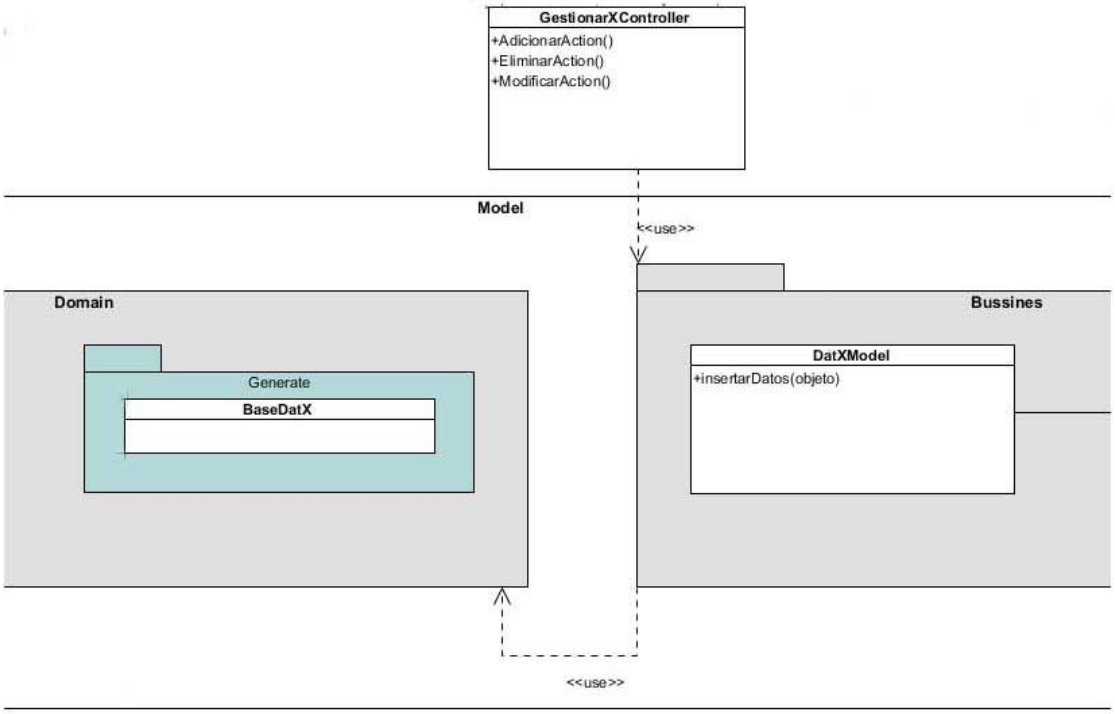


Figura 12 Mecanismo de diseño control, negocio y dominio

Este mecanismo muestra el flujo y las capas intermedias hasta acceder a los datos desde las clases controladoras. Las clases controladoras usan las clases ubicadas en el paquete Bussines (Negocio), y nunca directamente a las clases del paquete Domain (Dominio) pues son las clases Model las indicadas para usar el dominio. Esto permite aislar una capa de otra, facilitando el mantenimiento y la reutilización. Este mecanismo puede verse también en el diagrama de paquetes del modelo de diseño. (42)

3.1.1.7. Mecanismo de diseño recuperaciones con la librería TCPDF

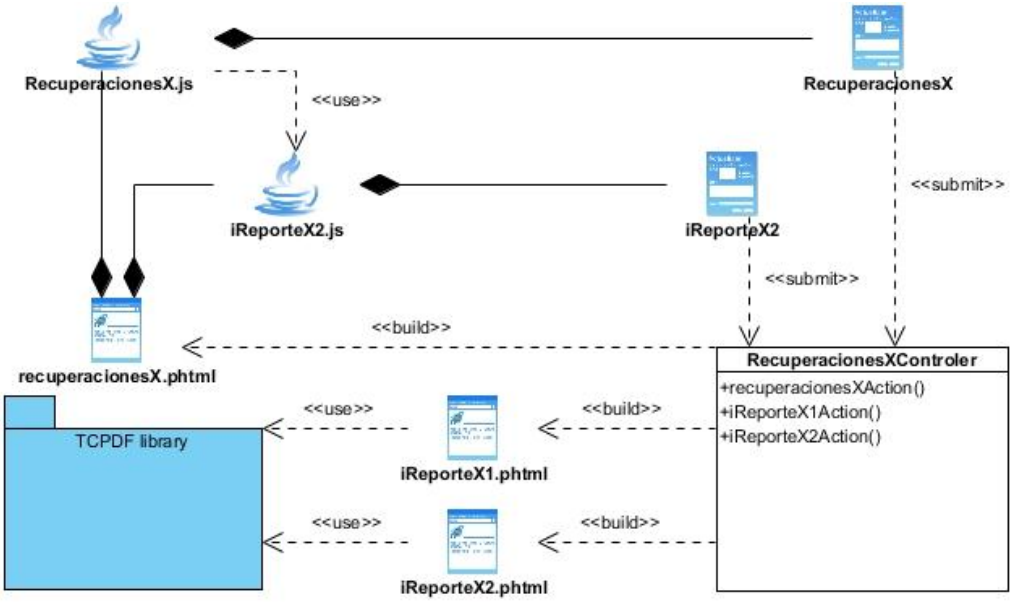


Figura 13 Mecanismo de diseño recuperaciones con la librería TCPDF

Las páginas clientes iReporteX1.phtml y iReporteX2.phtml usan la librería TCPDF para crear los reportes en formato PDF. Ambas páginas clientes fueron creadas por el controlador RecuperacionesXController.php. Para la construcción del primer reporte no se utiliza un filtro intermedio pues no lo requiere; el segundo reporte si necesita de datos de entrada de un actor del sistema para su creación que se recogen en el formulario iRepoerteX2 del fichero JavaScript iReporteX2.js estos son los únicos dos casos para imprimir un reporte utilizando la biblioteca TCPDF. Nótese además que está presente el mecanismo para las páginas y funciones de reportes, pues todas comienzan con una i (de impresión). (42)

Los demás mecanismos de diseño se encuentran recogidos en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

3.1.2. Patrones de diseño

El desarrollo del sistema empleando el marco de trabajo Sauxe proporciona ventajas significativas para los desarrolladores de software. El marco de trabajo mencionado es

capaz de fusionar buenas prácticas de trabajo por sí mismo, de forma que los desarrolladores no tengan que preocuparse por implementar varios de los patrones de diseño y arquitectónicos más utilizados en la actualidad, ya que el mismo framework los maneja. A continuación se exponen varios patrones de diseño que han sido utilizados durante el desarrollo de la solución:

3.1.2.1 Patrones GRASP

Controlador

El empleo de este se puede observar en la creación de una clase controladora para las diferentes funcionalidades que se realizan en el módulo Taller y Nomenclador o que están relacionados con estos. Para la gestión de las baterías y neumáticos fueron creadas las clases ControlBateriasController, ControlNeumaticosController, CambioBateriasController, CambioNeumaticosController y RecuperacionesController respectivamente.

Experto

Fue usado en todas las clases definidas ya que estas contaban con la información necesaria para cumplir con la responsabilidad asignada.

Alta Cohesión

Su empleo está dado en que fueron asignadas responsabilidades a las clases de forma tal que la cohesión siguiera siendo alta, o sea, cada clase se encargará de realizar solamente las funciones que estén en correspondencia con la responsabilidad que posea.

Bajo Acoplamiento

Un ejemplo donde se puede observar la utilización de este patrón es que la clase ControlBateriasController, se integra con el módulo Vehículo y no conoce la existencia de la clase DatVehiculo, que se encuentra en dicho módulo y contiene los datos que son solicitados, sino que interactúa con la clase fachada IoC, encargada de pasarle los valores pedidos.

3.1.2.2 Patrones GoF

Singleton

Este patrón fue utilizado en las clases del dominio, donde los métodos contenidos en estas se hicieron de forma estática, con lo cual se garantizaba que pudieran ser accedidos sin crear una instancia de las mismas. Por ejemplo: para acceder al método listarTodos () de la clase del domain DatBateria, desde la clase DatBateriaModel, del modelo, solo era necesario poner DatBateria::listarTodos ().

Fachada

La utilización de este patrón está reflejada en la creación y empleo de la clase IOC, ya que por la necesidad de integración entre los módulos del Sistema de Gestión de Flota y Mantenimiento, era necesaria la implementación de una clase fachada donde fueran publicados los servicios necesarios por estos, facilitando su interacción.

3.1.3. Modelo Vista Controlador

La implementación de este patrón en las funcionalidades del proceso Control de Neumáticos y Baterías se realizó de la siguiente manera: La Vista engloba la capa de presentación, en la misma es manejado todo el flujo web. Le brinda al usuario la posibilidad de interactuar con el Controlador mediante los mensajes de eventos y le permite visualizar las respuestas a sus peticiones. El Controlador por su parte representa el gestor de eventos, el cual está compuesto por las clases controladoras. Este atiende los pedidos que llegan desde la Vista accediendo al paquete models. El Modelo se encuentra dividido en los paquetes business, el cual encierra las clases donde se realiza la lógica del negocio, y domain, donde están comprendidas las clases del dominio, las que se encargan de leer y/o escribir datos en las tablas de la base de datos.

3.2. Diseño del sistema

En el desarrollo o ciclo de vida de un sistema informático, el diseño del sistema constituye un elemento fundamental del mismo. Se centra en proporcionar la funcionalidad del sistema a través de sus diferentes componentes. En esta fase se diseña una solución que convierte los requisitos del cliente en un sistema de información real.

El diseño debe proporcionar una completa idea de lo que es el software, enfocando los dominios de datos, funcional y comportamiento desde el punto de vista de la implementación.

3.2.1. Diagramas de clases del diseño con estereotipos web

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación, contiene información como asociaciones, atributos, métodos, y dependencias.

En la figura 14 se muestra el diagrama de clases del diseño para la agrupación de requisitos Control de baterías. En esta es representado el controlador “ControlBateriasController”, el cual respondiendo a una petición realizada desde la vista principal, a través de una relación <<link>>, construye la página cliente

“controlbateria.phtml” mediante la relación <<build>>. Esta página cliente contiene los script donde se encuentra implementado el código de ejecución de la página y a su vez contiene a un formulario a través de una relación <<include>>, el cual toma los datos que el usuario ingrese en la página y los envía al controlador a través de la relación <<submit>>.

Para dar respuesta a las peticiones realizadas por la página cliente, el controlador accede a las clases del modelo, que tienen la responsabilidad de realizar las operaciones sobre la información de las tablas de la base de datos y enviarle los resultados. Estas operaciones son realizadas accediendo a las clases contenidas en el paquete Domain, ver Figura 15, donde se encuentran mapeadas las tablas de la base de datos y las clases con el prefijo “Base_”, las que utilizan las funcionalidades brindadas por el marca de trabajo Doctrine para acceder a las tablas de la base de datos. En caso de que la información pedida por la página cliente se encuentre en otro componente, el controlador a través de la relación <<dependencia>> accede a la clase IOC, accede a las clases de servicio del componente y a partir de la información obtenida, realizan las operaciones necesarias y le envían los resultados al controlador. En la figura 8 se muestran las clases contenidas en los paquetes del Domain así como las relaciones entre las mismas.

En la figura 16 se muestra el diagrama de clases del diseño de los reportes de neumáticos y baterías. En este se representa el controlador RecuperacionesController, el cual respondiendo a la petición realizada desde la vista principal a través de la relación <<link>>, construye la página cliente recuperaciones.phtml mediante la relación <<build>>. Esta página cliente contiene el script recuperaciones.js, el cual incluye a su vez, el formulario recuperaciones, con una relación <<include>>, siendo el encargado de tomar los datos ingresados por el usuario en la página y enviarlos, a través de la relación <<submit>>, al controlador para procesarlos. Para dar respuesta a las peticiones realizadas por la página cliente, el controlador accede a la clase del modelo DatRecuperacionesModel, que tiene la responsabilidad de realizar las operaciones sobre la información de las tablas de la base de datos y enviarle los resultados. Para mostrar los resultados enviados desde la clase DatRecuperacionesModel al controlador RecuperacionesController, este crea una página cliente “nombredelarelación.phtml”, la cual haciendo uso de la biblioteca

TCPDF, muestra los resultados en formato pdf.

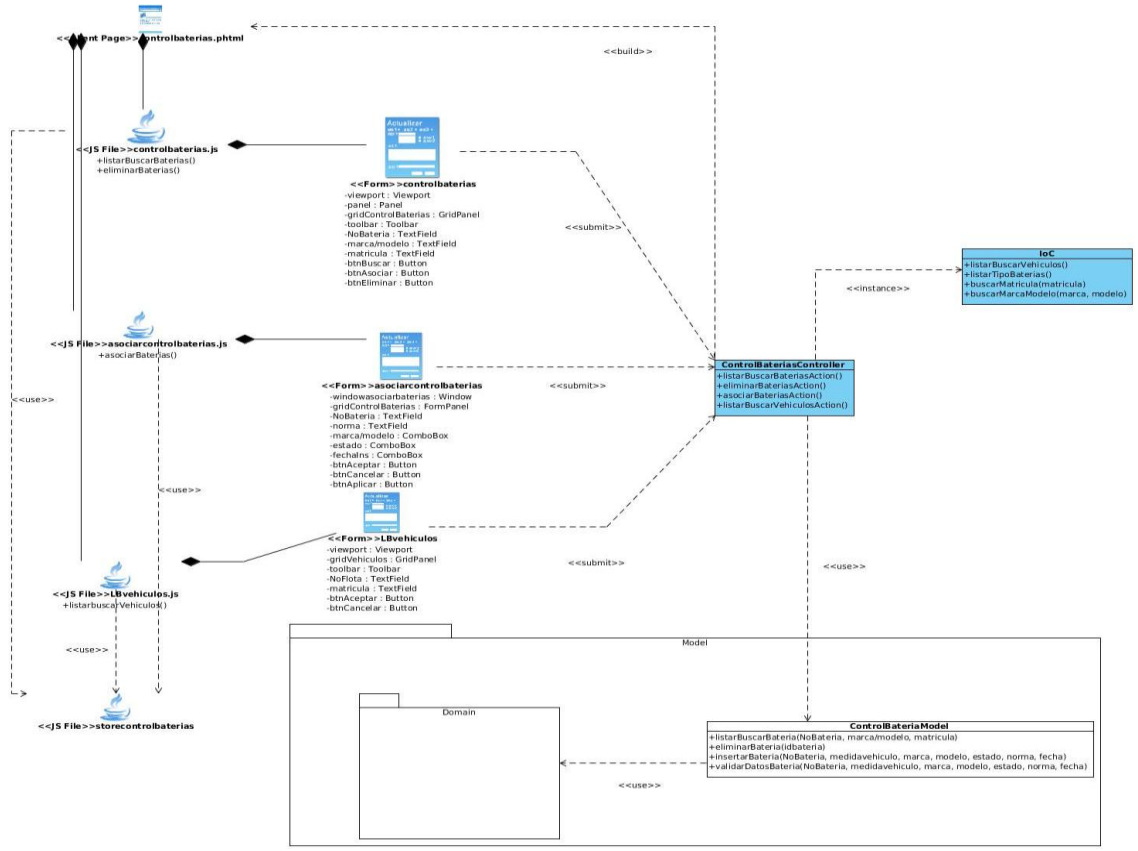


Figura 14 Diagrama de clases Control de baterías

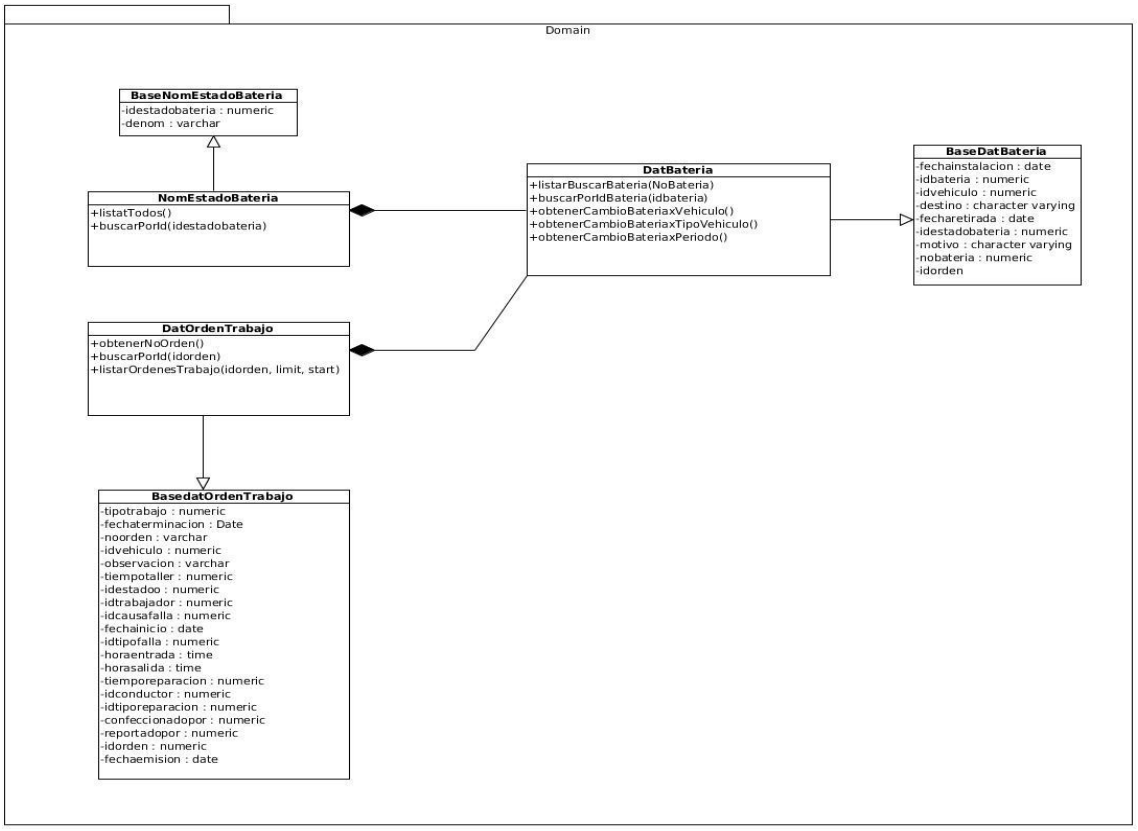


Figura 15 Diagrama de clases Control de baterías del Domain

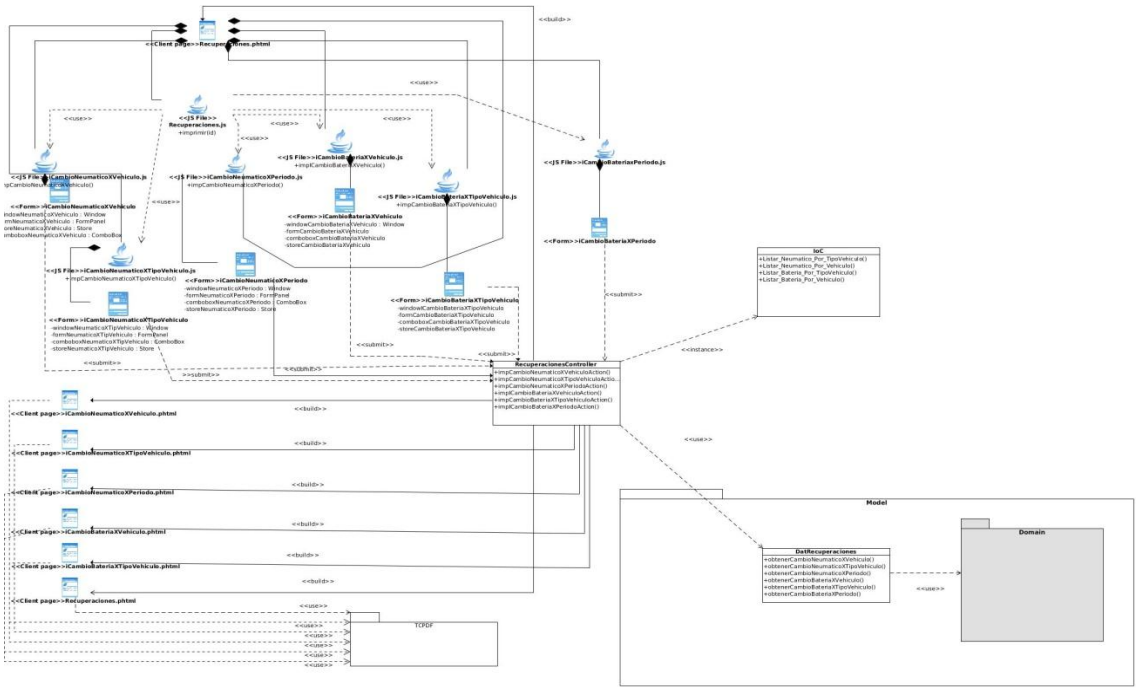


Figura 16 Diagrama de clases Reportes Control de neumáticos y baterías

Para consultar los diagramas de clases correspondientes al resto de los requisitos consultar entregables recogidos en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

3.2.2 Diagrama de secuencia

El diagrama de secuencias, que se define en UML (Unified Modeling Language), es uno de los más utilizados para identificar el comportamiento de un sistema (35), por representar los objetos que se encuentran en el escenario y la secuencia de mensajes intercambiados entre los objetos para llevar a cabo la funcionalidad descrita por una transacción del sistema.

La figura 17 muestra un ejemplo de diagrama de secuencia, perteneciente al RF Listar Batería. Para consultar los diagramas de secuencia correspondientes al resto de los requisitos consultar entregables recogidos en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

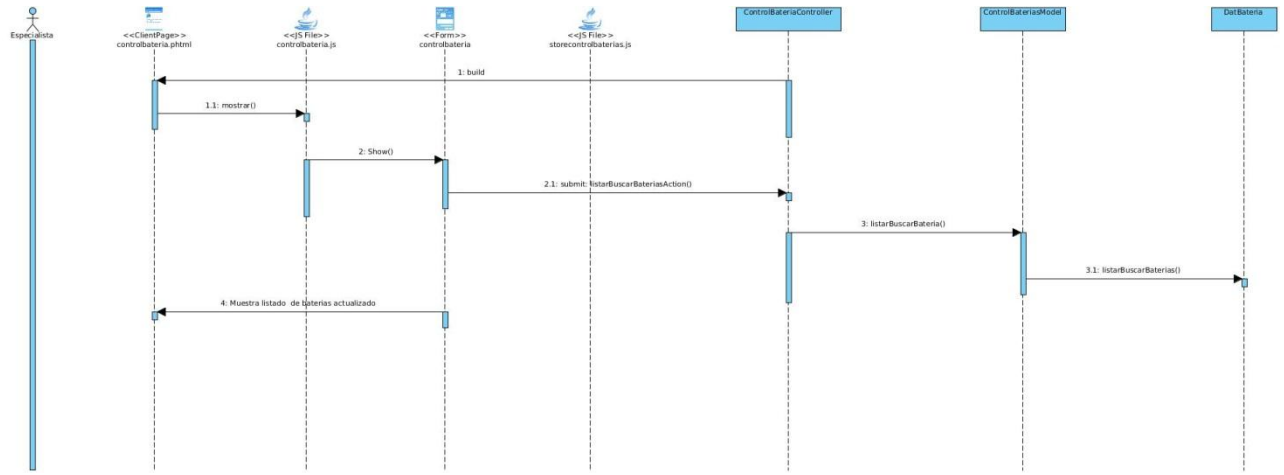


Figura 17 Diagrama de secuencia Asociar baterías

3.2.3. Descripción de las clases del diseño

Tabla 5 Descripción de clase del diseño

Nombre: DatNeumaticoModel	
Tipos de clase: Modelo	
Para cada responsabilidad:	
Nombre	Guardar($Neumático)
Descripción	A partir de un arreglo pasado con los datos obtenidos de un neumático, realiza su adición al sistema, enviando estos a la base de datos utilizando la clase DatNeumatico.
Nombre	listarNeumaticos($limit,$start)
Descripción	Permite listar los datos de los neumáticos que se encuentran en el sistema.
Nombre	verificarExistenciaNeumatico($nobateria,$idvehiculo,$fechainstalacion,$idtiponeumatico)
Descripción	Permite verificar a partir del nobateria, el idvehiculo, la fechainstalacion y el idtiponeumatico si existe un neumático con esos mismos parámetros.
Nombre	buscarNeumaticos ($nobateria,$marca,$modelo,$matricula)
Descripción	Permite listar los datos de los neumáticos que se encuentran en el sistema que coinciden con estos parámetros.
Nombre	Eliminar($Neumático)

Descripción	A partir del identificador de un neumático, se realiza su eliminación del sistema, utilizando la clase DatNeumatico.

3.2.4. Métricas para la evaluación del diseño

Del conjunto de métricas planteadas por Lorenz y Kidd se aplicó al diseño propuesto:

- Tamaño operacional de clase (TOC)

El TOC está dado por el número de métodos asignados a una clase. Es propuesta por Lorenz y Kidd, en su libro sobre métricas Orientada a Objetos (OO) y se centran en el recuento de atributos y operaciones para cada clase individual, y los valores promedio para el sistema como un todo.

3.2.4.1. Métrica de Tamaño Operacional de Clase

La métrica Tamaño Operacional de Clase (TOC) se encarga de medir la calidad de acuerdo a los atributos Responsabilidad, Complejidad de Implementación y Reutilización de las clases.

La aplicación de la métrica TOC define los siguientes atributos de calidad:

- Responsabilidad: Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- Complejidad de implementación: Consiste en el grado de dificultad que tiene implementar un diseño de clases determinado.
- Reutilización: Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.

Para evaluar las métricas son necesarios los valores de los umbrales para los parámetros de calidad. Algunos especialistas plantean umbrales para esta métrica basándose en el promedio de operaciones por clases obtenidos, estos valores fueron los aplicados en el diseño del sistema y los mismos son reflejados en la siguiente tabla.

Tabla 6 Umbrales para la TC

Atributo	Clasificación	Valores de los umbrales
Responsabilidad	**Baja**	<=Promedio de operaciones(PO)
	Media	>PO y <=2*PO
	Alta	>2*PO
Reutilización	**Baja**	>2*PO
	Media	>PO y <=2*PO

	Alta	<=Promedio de operaciones(PO)
Complejidad	**Baja**	<=Promedio de operaciones(PO)
	Media	>PO y <=2*PO
	Alta	>2*PO

Tabla 7 Umbrales para la TC

No	Nombre	Cantidad de atributos	Cantidad de operaciones	Tamaño
1	GestneumaticoController	0	4	Baja
2	GesttipobateriaController	0	5	Baja
3	NomTiponeumaticoModel	0	7	Baja
4	NomTipobateriaModel	0	10	Baja
5	ControlbateriasController	0	6	Baja
6	ControlneumaticosController	0	6	Baja
7	DatBateriaModel	0	10	Baja
8	DatneumaticoModel	0	9	Baja
9	GestOrdenTrabajoController	0	25	Alta
10	NomEstadobateriaModel	0	4	Baja
11	NomEstadoneumaticoModel	0	4	Baja
12	NomPosicionneumaticoModel	0	5	Baja
13	DatUnidadmedidaModel	0	7	Baja

A continuación se muestran los resultados de la evaluación de la métrica TOC:

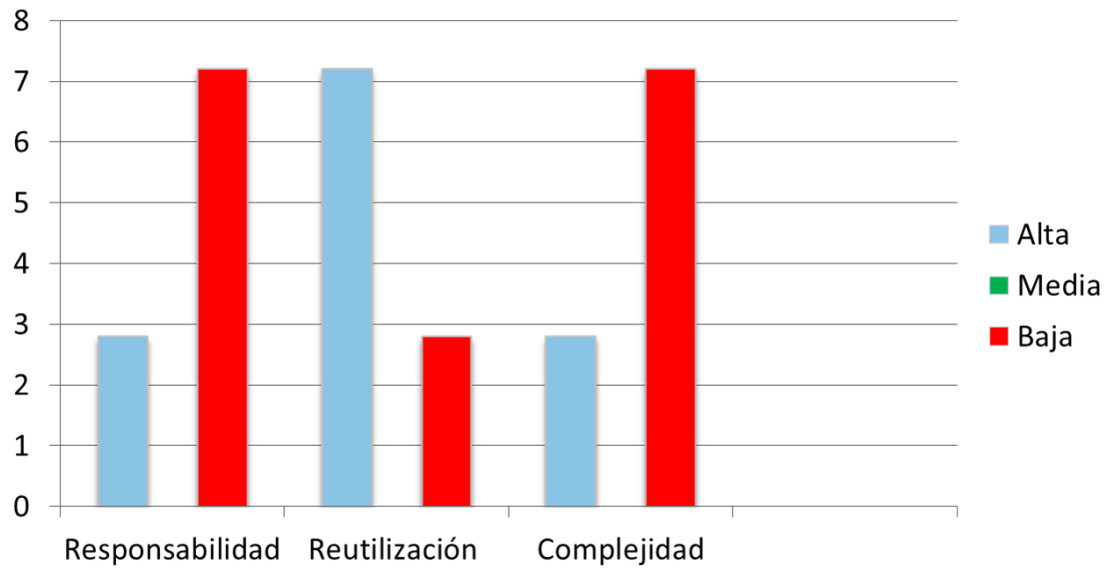


Figura 18 Resultados de la evaluación del diseño con TOC

Haciendo un análisis de los resultados obtenidos en la evaluación de la métrica TOC, se demuestra que el diseño presenta una calidad aceptable pues más del 70% de las clases poseen evaluaciones positivas en los atributos de calidad Responsabilidad, Complejidad de Implementación y Reutilización.

3.2.4.2. Métrica de Relaciones entre Clases

Las relaciones entre clases están dadas por el número de relaciones de uso de una clase con otras.

Tabla 8 Rango de valores para los criterios de evaluación de la métrica Relaciones entre clases (RC)

Atributo	Categoría	Criterio
Acoplamiento	Ninguno	0
	Bajo	1
	Medio	2
	Alto	>2
Complejidad de mantenimiento	Baja	<=Promedio
	Media	>Promedio y <= 2*Promedio
	Alta	>2*Promedio
Reutilización	Alta	<=Promedio
	Media	>Promedio y <= 2*Promedio
	Baja	>2*Promedio
Cantidad de pruebas	Baja	<=Promedio
	Media	>Promedio y <= 2*Promedio
	Alta	>2*Promedio

A continuación se muestran los resultados de la evaluación de la métrica RC:

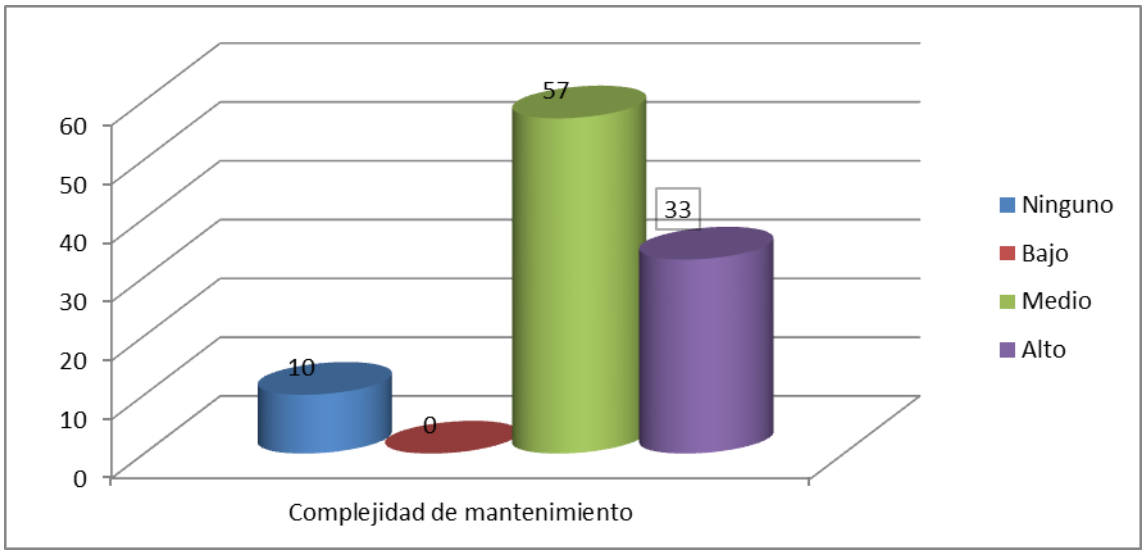


Figura 19 Gráfica que representa el % de clases por categorías del atributo Complejidad de Implementación obtenidos en la aplicación de la métrica RC

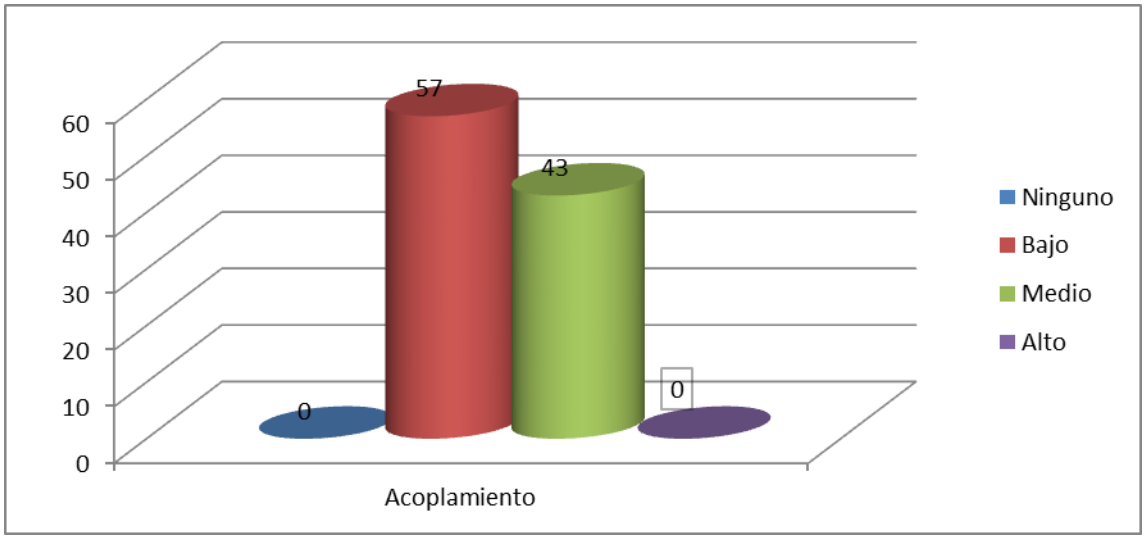


Figura 20 Gráfica que representa el % de clases por categorías del atributo Acoplamiento obtenidos en la aplicación de la métrica RC

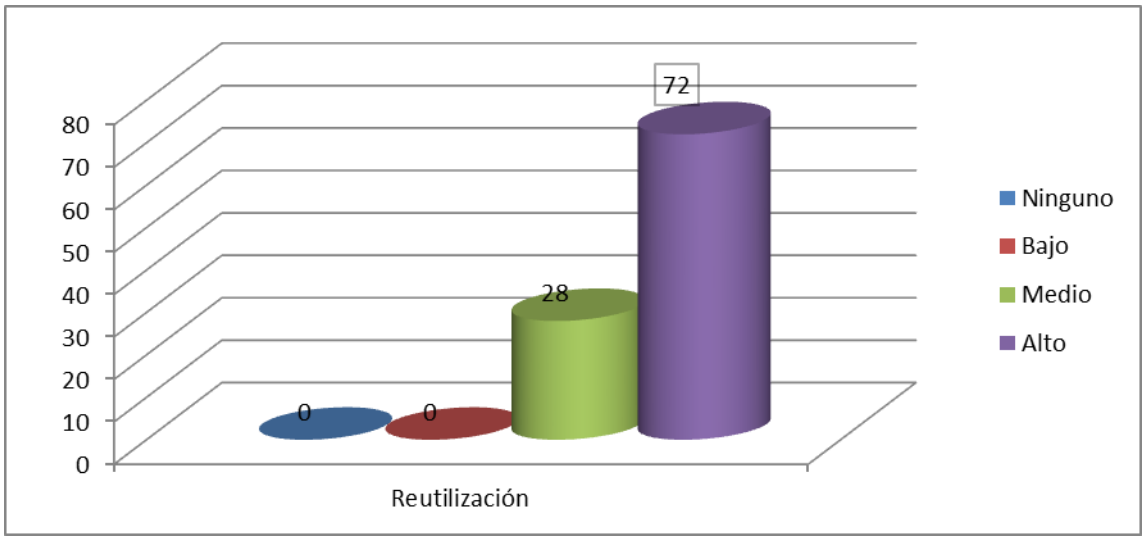


Figura 21. Gráfica que representa el % de clases por categorías del atributo Reutilización obtenidos en la aplicación de la métrica RC

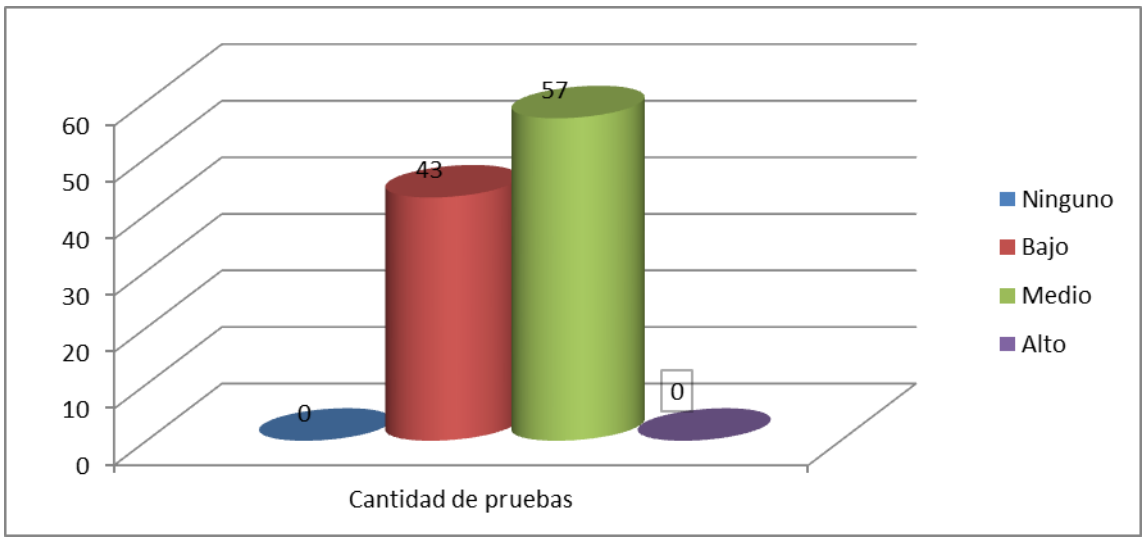


Figura 22 Gráfica que representa el % de clases por categorías del atributo Cantidad de Pruebas obtenidos en la aplicación de la métrica RC

Haciendo un análisis de los resultados obtenidos en la evaluación de la métrica RC, se puede concluir que el diseño del sistema tiene una calidad aceptable teniendo en cuenta que el 57% de las clases tienen 3 o menos dependencias de otras clases, además el 57% de las clases posee índices aceptables en cuanto a Acoplamiento. Así mismo los atributos de Complejidad de Mantenimiento, Cantidad de pruebas y Reutilización se comportan satisfactoriamente.

Los resultados obtenidos en la aplicación de las métricas TOC y RC demuestran la realización de un buen diseño.

3.2.5. Modelo de datos

El modelo de datos es un conjunto de conceptos, reglas y convenciones que permite describir y manipular los datos de un mundo real que deseamos almacenar en la base datos. (43)

En el modelo de datos elaborado para el proceso Control de Neumáticos y Baterías del Sistema de Gestión de Flota y Mantenimiento se visualizan las entidades, sus atributos y las relaciones entre ellas. Son las encargadas de almacenar todos los datos necesarios para que las funcionalidades que fueron definidas en el análisis puedan ser llevadas a cabo.

A continuación se presenta el modelo de datos realizado:

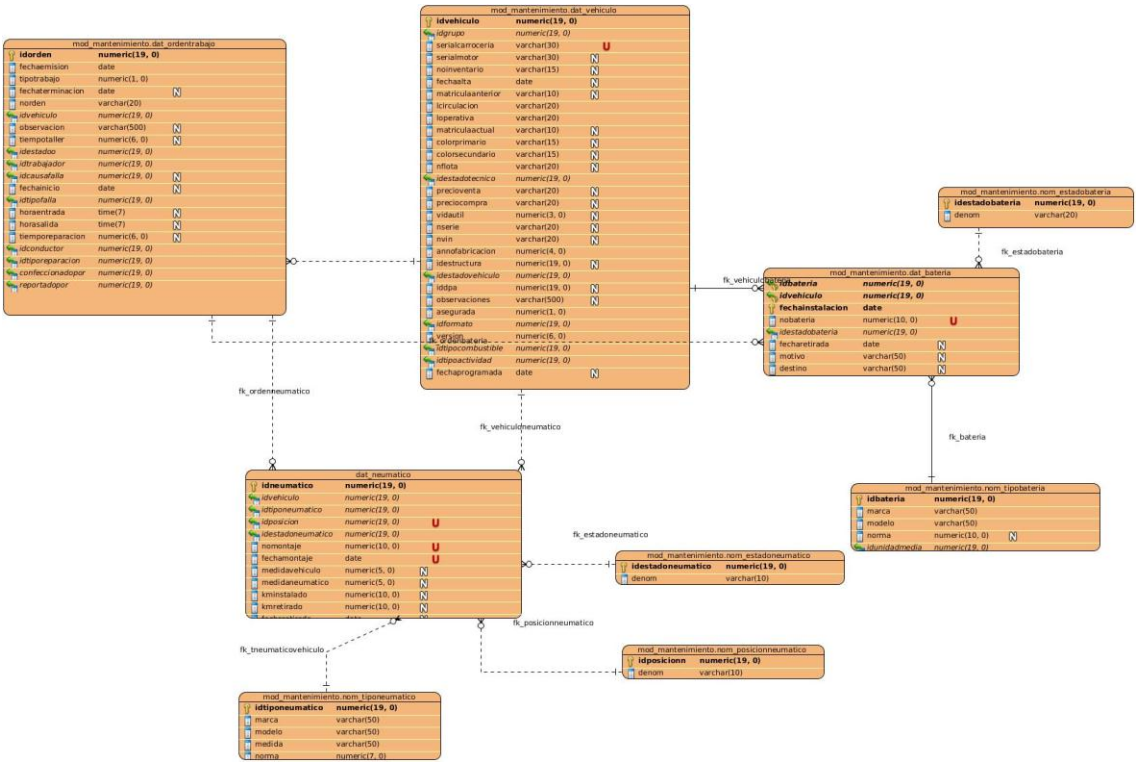


Figura 23 Modelo de datos

3.3. Implementación

La implementación de las funcionalidades de apoyo al proceso Control de Neumáticos y Baterías proporcionará como resultado un producto que cumpla las exigencias y necesidades existentes en el área de Transporte de la UCI.

3.3.1. Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Se realizan con el objetivo de poseer una vista de forma general del sistema a partir de las dependencias e integraciones de los componentes y módulos.

En la figura 24 se muestra el diagrama de componentes del Sistema de Gestión de Flota y Mantenimiento.

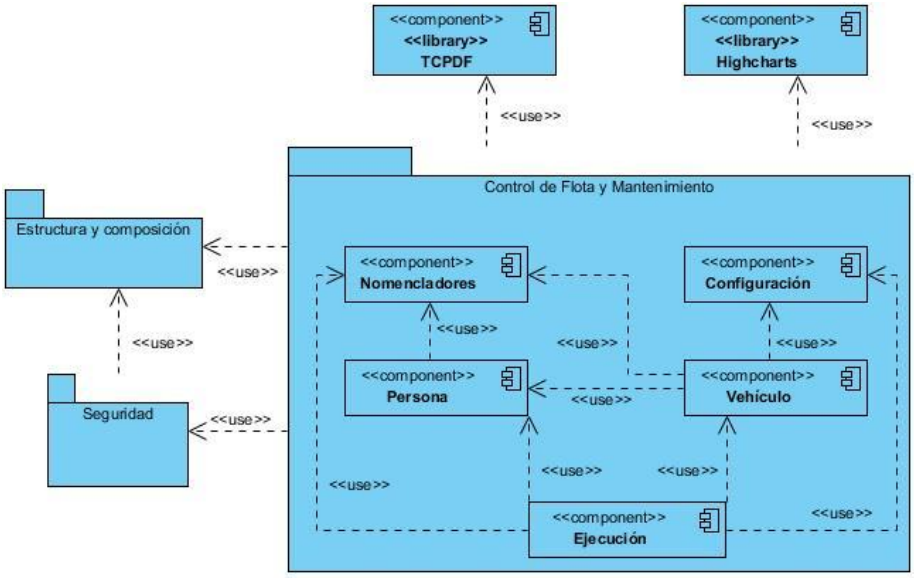


Figura 24 Diagrama de componentes

En este se representa la integración entre los mismos, o sea los servicios que brindan unos y son consumidos por otros y viceversa. El proceso Control de Neumático y Baterías se encuentra distribuido en el componente Nomencladores donde se encuentran los nomencladores de tipo de neumático y batería. Los datos de los vehículos se encuentran el componente Vehículo y el componente fundamental donde se encuentran las clases y los métodos que soportan los requisitos están en el componente Ejecución.

3.3.2. Modelo de despliegue

Los diagramas de despliegue visualizan la disposición física de los distintos recursos computacionales que componen un sistema y el reparto de los componentes sobre dichos nodos. (44)

Para el modelo realizado se definió una arquitectura cliente-servidor de la siguiente manera:

Cliente: accede a la aplicación a través de un computador, donde es ejecutada mediante el navegador Mozilla Firefox 16.0, sobre cualquier sistema operativo.

Servidor Web: el servidor de aplicaciones empleado donde radica la lógica de negocio de la aplicación es el Servidor Web Apache 2.2 utilizando el lenguaje PHP.

Servidor de Base de datos: servidor de Datos PostgreSQL 9.1.

A continuación se presenta el modelo de despliegue elaborado para el Sistema de Gestión de Flota y Mantenimiento:

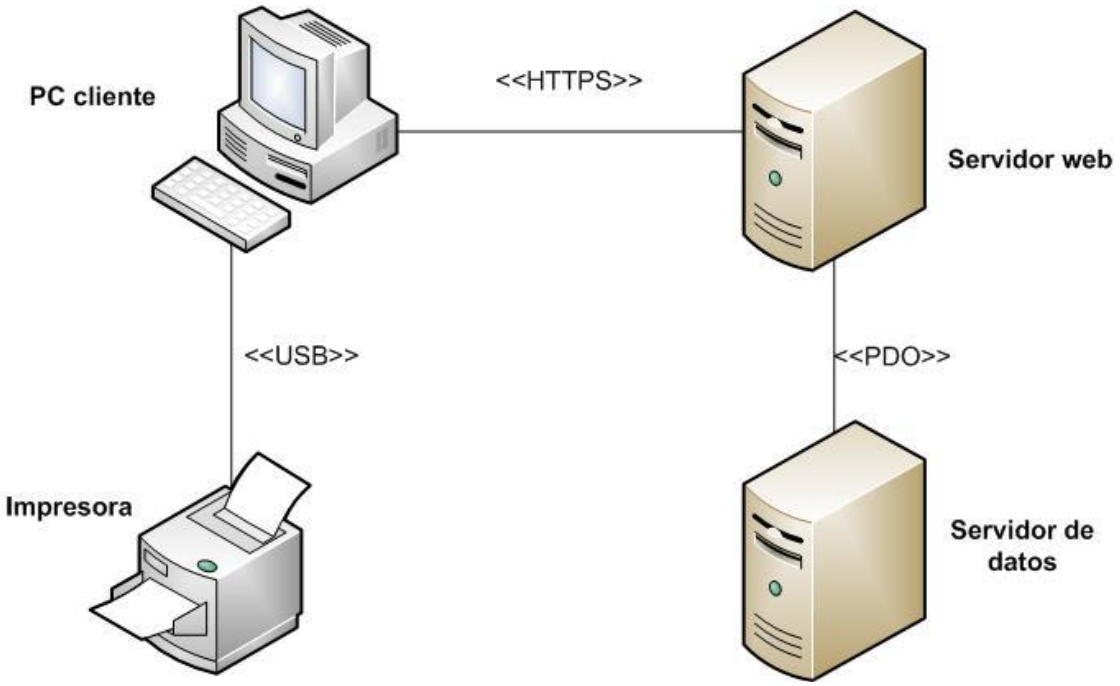


Figura 25 Modelo de despliegue

3.3.3. Estándares de codificación

Los estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, la comprensión y mantenimiento del código. (45)

A continuación se especifican los estándares de codificación que fueron utilizados en el desarrollo del Sistema Gestión de Flotas y Mantenimiento:

3.3.3.1. Estándares de codificación

- Para las clases controladoras el nombre comenzará con la primera letra en mayúscula y el resto en minúscula, además estará dado según la función que realiza y se le adicionará al final “Controller”. Ejemplo: ControlBateriasController.
- El nombre de las clases del modelo que se encuentran dentro de la carpeta “business” comenzará con la primera letra en mayúscula y el resto en minúscula. A este nombre se le agregará al final “Model” y al inicio se le colocará “Nom”, en caso de que sea un nomenclador, de lo contrario será Dat, por ejemplo: DatBateriaModel, NomTipobateriaModel.
- Las clases del dominio que se encuentran dentro de la carpeta “domain”, son generadas mediante un mapeo a la base de datos realizado por la herramienta Doctrine Generator, y son nombradas igual que las tablas de la base de datos. Ejemplo: DatBateria.

- Las clases contenidas en la carpeta “generated” son generadas mediante un mapeo a la base de datos realizado por la herramienta Doctrine Generator y delante del nombre se le coloca “Base”.
- EL nombre de las clases de la vista comenzará con la primera letra en mayúscula y el resto en minúscula, en caso de que este sea compuesto, se utilizará la notación Pascal Casing. (46) A este nombre se le agregaran al principio, las letras en mayúscula de las funciones que se realizarán a través de estas clases. Ejemplo: LBVehiculos (listar y buscar), AMETipobateria (adicionar, modificar y eliminar).

3.3.3.2. Nomenclatura de los métodos o funciones

El nombre de los métodos o funciones de una clase comenzará en minúscula, en caso de que sea compuesto se utilizará la notación Camel Casing, o sea, seguido del primer nombre, comenzará el segundo con mayúscula. En caso de que sea una función de una clase controladora se le agregará al final la palabra “Action”. Ejemplo: asociarBateriaAction, listarBateriaAction.

3.3.3.3. Nomenclatura de las variables

Las variables serán nombradas comenzando en minúscula, en caso de que el nombre de estas sea compuesto, se utilizará la notación Camel Casing, o sea, seguido del primer nombre, comenzará el segundo en mayúscula. En la capa de la Vista, las variables que contengan componentes que formen parte de la interfaz, se les colocará delante un sufijo que indique el tipo de componente que contiene. Ejemplo: btnAsociarNeumatico, storeNeumatico.

3.3.4. Publicación de servicios entre componentes

Para cada uno de los componentes fueron publicados en sus clases Services, los mismos métodos que son utilizados en estos para la recuperación de los datos en las clases del modelo y del dominio, realizándose de esta manera con el objetivo de reutilizar código existente. Los servicios fueron implementados de modo que reciban muchos o ningún parámetro, siempre que fuera posible, disminuyendo con esto la cantidad de servicios a publicar.

El proceso Control de Neumáticos y Baterías emplea las clases Services, DatBateriaService, DatNeumaticoService, NomTipobateriaService, NomTiponeumaticoService y DatVehiculoService las cuales contienen todos los servicios que son consumidos en el proceso Control de Neumáticos y Baterías.

A continuación se muestran los servicios que son consumidos en este proceso:

Tabla 8 Servicios consumidos en el proceso Control de Neumáticos y Baterías

Servicios	Componentes que lo brinda	Componentes que lo utilizan	Descripción
Listar_Vehiculo	Vehículo	Ejecución	Muestra un listado de todos los vehículos que se encuentran registradas en el sistema.
Buscar_Vehiculo	Vehículo	Ejecución	A partir del identificador de un vehículo, se obtienen todos los datos del mismo.
Listar_Tipobateria	Nomencladores	Ejecución	Muestra un listado de todos los tipos de baterías que se encuentran registradas en el sistema.
Buscar_ Tipobateria	Nomencladores	Ejecución	A partir del identificador de un tipo de batería, se obtienen todos los datos del mismo.
Listar_Tiponeumatico	Nomencladores	Ejecución	Muestra un listado de todos los tipos de neumáticos que se encuentran registradas en el sistema.
Buscar_ Tiponeumatico	Nomencladores	Ejecución	A partir del identificador de un tipo de neumático, se obtienen todos los datos del mismo.
Buscar_ TipobateriaA	Nomencladores	Nomenclador	A partir del identificador de un tipo de batería, si está asociada a una batería se obtienen los datos de la batería.
Buscar_ TiponeumaticoA	Nomencladores	Nomenclador	A partir del identificador de un tipo de neumático, si está asociado a un neumático se obtienen los datos del neumático.

3.4. Pruebas de software

Una vez implementadas las funciones que deben soportar los requisitos se realizaron un conjunto de pruebas para garantizar la calidad del cumplimiento de los requisitos aceptados. A este conjunto de técnicas experimentales para la búsqueda de fallos en los programas se les denomina pruebas de software.

Para garantizar la calidad de la inserción, análisis, y recuperación correcta de la información del proceso Control de Neumáticos y Baterías de la Dirección de

Transporte de la UCI se hace evidente la necesidad de contar con pruebas de este tipo desde las primeras etapas de concepción de un proyecto.

3.4.1. Pruebas de caja blanca

Para utilizar la **Técnica del Camino Básico**, primeramente se hace necesario el cálculo de la complejidad ciclomática del algoritmo que vaya a ser analizado, para lo cual se deben enumerar las sentencias de código de este y a partir de ahí elaborar el grafo de flujo de esta funcionalidad.

Las sentencias enumeradas del método **listarBaterias** () se encuentran en la figura 26 mostrada a continuación.

```
function listarBateriasAction()
{
    $start = $this->_request->getPost("start");                    //1
    $limit = $this->_request->getPost("limit");                    //1
    $matriculaactual = $this->_request->getPost("matriculaactual"); //1
    $nobateria = $this->_request->getPost("nobateria");            //1
    $marcamodelo = $this->_request->getPost("marcamodelo");        //1
    $tipobateria = explode("/",$marcamodelo);                      //2
    $marca =  $tipobateria[0];                                     //3
    $modelo =  $tipobateria[1];                                    //3

    if($nobateria OR $matriculaactual OR  $marca OR $modelo ){      //4

    $model = new DatBateriaModel();                                //5

    $result = $model->buscarBateria($matriculaactual,$nobateria,$marca,$modelo,$start,$limit);   //6

        }
        else{

    $model = new DatBateriaModel();                                  //7

    $result = $model->listarBateria($start,$limit);                  //8

            }
                //print_r($result);die;
echo json_encode($result);return;                                    //9


    }
```

Figura 26 Sentencias enumeradas al algoritmo listarBaterias ()

A continuación en la figura 27 se muestra el grafo de flujo asociado a la funcionalidad **listarBaterias ()**

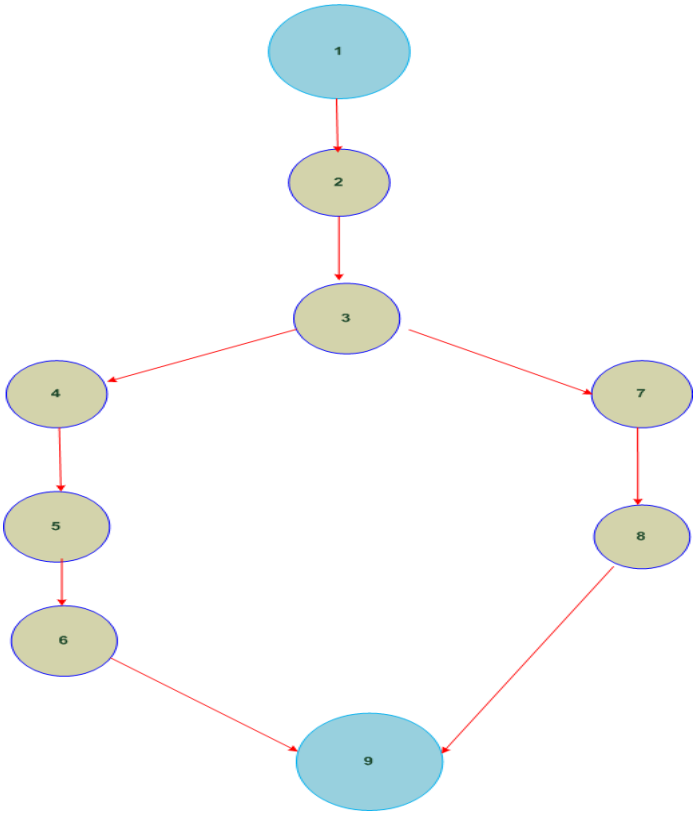


Figura 27 Grafo de flujo asociado al algoritmo de despliegue listarBaterias ()

3.4.1.1. Cálculo de la complejidad ciclomática a partir de un segmento de código

La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa. El valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y ofrece un límite superior para el número de pruebas que se deben realizar para asegurar que sea ejecutada cada sentencia al menos una vez. (35)

La complejidad ciclomática de un código determinado puede ser calculada de tres maneras diferentes. Al calcular la complejidad del método listarBaterias () fueron utilizadas las tres formas posibles con el objetivo de verificar que el cálculo se había realizado correctamente, si los resultados obtenidos en cada una era el mismo. Las fórmulas para realizar dicho cálculo son:

1. V (G) = (A - N) + 2, V (G)= (9 -9)+2, V (G)= 2

Siendo A la cantidad total de aristas del grafo y N la cantidad de nodos.

2. V (G) = P +1, V (G) = 1+1, V (G) = 2

Siendo P la cantidad de nodos predicado (son aquellos de los cuales parten dos o más aristas)

3. V (G) = R, V (G) = 2
Siendo R la cantidad de regiones que posee el grafo.

En cada una de las fórmulas V (G) ha representado el valor del cálculo. A partir de los resultados obtenidos en cada uno, se puede determinar que la complejidad ciclomática del código analizado es 2, que a su vez es el número de caminos posibles a circular el flujo y el límite superior de casos de prueba que se le pueden aplicar a dicho código.

A continuación se muestran los caminos básicos por donde puede circular el flujo:
Camino básico # 1: 1-2-3-4-5-6-9
Camino básico # 2: 1-2-3-7-8-9

Para cada uno de los caminos obtenidos se realiza un caso de prueba. Los casos de prueba realizados son los siguientes:

Caso de prueba para el Camino básico # 1

Camino: 1-2-3-4-5-6-9
Descripción: Muestra un listado de todas las baterías que se encuentran registradas en el sistema.
Condición de ejecución: Para ejecutar el algoritmo es necesario que los datos de entrada cumplan con los siguientes requisitos: el start y el limit deben estar dados en valores numéricos. Los campos correspondientes al número de batería, matricula actual y marca pueden o no estar vacíos.
Entrada:
$nobateria = 1
$matriculaactual = 9000002
$marca = Laxus
$modelo = Freedom
Resultados esperados: Teniendo en cuenta los datos pasados por parámetro se espera que el algoritmo muestre un listado de las baterías registradas en el sistema que cumplan con los criterios de búsqueda pasados por parámetro.

Caso de prueba para el Camino básico # 2

Camino: 1-2-3-7-8-9
Descripción: Muestra un listado de todas las baterías que se encuentran registradas en el sistema.
Condición de ejecución: Para ejecutar el algoritmo es necesario que los datos de entrada cumplan con los siguientes requisitos: el start y el limit deben estar dados en

valores numéricos. Los campos correspondientes al número de batería, matricula actual y marca no pueden estar vacíos.

Entrada:

$nobateria = Vacío

$matriculaactual = Vacío

$marca = Vacío

$modelo = Vacío

Resultados esperados: Se espera que el algoritmo muestre un listado de las baterías registradas en el sistema.

Con la aplicación de los casos de prueba expuestos anteriormente se comprobó que el flujo de trabajo de las funcionalidades es correcto, se probó que cada sentencia es ejecutada al menos una vez, cumpliéndose así las condiciones de la prueba.

3.4.2. Prueba de caja negra

A este tipo de prueba también se le conoce como Prueba de Caja Opaca o Inducida por los Datos. Se centran en lo que se espera de una funcionalidad, es decir, intentan encontrar casos en que la funcionalidad no se atiene a su especificación, esta prueba se limita a brindar solo datos como entrada y estudiar la salida, sin preocuparse de lo que pueda estar haciendo la funcionalidad internamente, es decir, solo trabaja sobre su interfaz externa.

Para cada uno de los requisitos funcionales fueron definidos casos de prueba, los cuales son los siguientes:

Caso de prueba para el requisito Asociar batería

Condiciones de ejecución

- Se debe identificar y autenticar ante el sistema, además debe tener los permisos para ejecutar esta acción.
- Se debe seleccionar la opción del menú: **Mantenimiento/Taller/Control de baterías**
- Se ha registrado al menos una herramienta en el sistema.

Nombre del requisito	Descripción general	Escenarios de pruebas	Flujo del escenario
1: Asociar batería a vehículo.	El sistema debe permitir asociar baterías a un vehículo.	EP 1.1: Asociar batería a vehículo empleando datos válidos.	Se selecciona el vehículo al cual se le va a asignar la batería. Se introducen los datos de la batería Se presiona el botón **Aceptar**. El sistema valida los datos introducidos. El sistema confirma el registro de los datos.

EP 1.2: Asociar batería a vehículo empleando datos inválidos.	Se selecciona el vehículo al cual se le va a asignar la batería. Se introducen los datos de la batería Se presiona el botón **Aceptar**. El sistema señala los datos erróneos y permite corregirlos. El sistema confirma el registro de los datos.
EP 1.3: Asociar batería a vehículo dejando campos vacíos.	Se selecciona el vehículo al cual se le va a asignar la batería. Se introducen los datos de la batería Se presiona el botón **Aceptar** El sistema señala los datos vacíos y permite corregirlos. El sistema confirma el registro de los datos.
EP 1.4: Cancelar.	Se seleccionan o no la batería que se va a asociar al vehículo. Se presiona el botón **Cancelar**.

Para más información sobre el caso de prueba para el requisito Asociar batería, consultar el documento entregable CIG-CFM-N-MTTO-i51112. Los casos de prueba elaborados para los restantes requisitos funcionales se encuentran en los documentos entregables recogidos en el repositorio del proyecto de Soluciones Empresariales del centro CEIGE.

3.4.3. Conclusiones parciales del capítulo

Durante el desarrollo del capítulo fueron expuestos los artefactos generados del diseño de las funcionalidades para la gestión del proceso Control de Neumáticos y Baterías del área de Transporte de la UCI, así como las métricas que fueron empleadas para realizar su validación, las que ofrecieron como resultado que el diseño estaba realizado de forma simple y con una calidad aceptable, permitiendo con esto darle paso a las actividades de la disciplina implementación. Además de lo anteriormente especificado, se presentaron las pruebas estructurales y funcionales que les fueron aplicadas, las que dieron como resultado que poseía un correcto

funcionamiento, contaba con las condiciones de calidad necesarias y satisfacía las necesidades plasmadas por el cliente.

Conclusiones generales

Tras haber culminado los tres capítulos con los que cuenta el presente trabajo de diploma se arriba a las siguientes conclusiones:

- Mediante la elaboración del marco teórico y el estudio del estado del arte se puso de manifiesto la necesidad de desarrollar una serie de funcionalidades de apoyo al proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
- La realización del análisis y diseño permitió obtener los requisitos y modelos necesarios para la implementación de funcionalidades de apoyo al proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
- La implementación del diseño propuesto permite que la Dirección de Transporte de la UCI cuente con una versión de funcionalidades de apoyo al proceso Control de Neumáticos y Baterías.
- La validación a través de las pruebas de caja blanca y caja negra permitió asegurar el correcto funcionamiento de las funcionalidades implementadas para apoyar el proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento de la Dirección de Transporte de la UCI.
- El desarrollo de funcionalidades de apoyo al proceso Control de Neumáticos y Baterías del Sistema Control de Flota y Mantenimiento permite la inserción, análisis y recuperación de la información referente a los neumáticos y baterías del Área de Transporte de la UCI, dándole cumplimiento al objetivo planteado en la investigación.

Recomendaciones

- Se recomienda que para el desarrollo de futuros sistemas para la gestión de flotas y mantenimiento sea utilizado el presente trabajo como referencia en cuanto a la gestión de los neumáticos y baterías.
- Se recomienda la puesta en explotación de la herramienta en el Área de Transporte de la UCI.

Bibliografía

1. **UCI, Especialistas del area de Transporte de la.** *Taller.* 11 de 2012.
2. **Solbyte.** www.novatrans.com. [En línea] 2012.
3. **Maximo.** http://www.transportex.net/informacion.html. [En línea] 2012.
4. **Ltda.** www.eugcom.cl/fichas_tecnicas/Flotas.pdf. [En línea] 2012.
5. **Apolo.** *Manual de Usuario Apolo Express 2004.* 2012.
6. **Albet.** *Manual de usuario.* 2012.
7. **Producción, Equipo de.** *Modelo de desarrollo orientado a componentes del proyecto ERP-Cuba.*
8. **CEIGE.** *Modelo de Desarrollo.* 2012.
9. **Tome, Rtzaida.** *Captura de requisitos.* 2011.
10. **Pressman, Roger S.** *Ingeniería del Software Un enfoque práctico.* 2006.
11. [aut. libro] ROGER PRESSMAN. *Ingenieria de Requsito Un Efoque Practico.* 2005.
12. **Fuentes, Lidia, Troya, José M. y Vallecillo, Antonio.** Lenguajes y Ciencias de la Computación. *Lenguajes y Ciencias de la Computación.* [En línea] 2012. [Citado el: 12 de 11 de 2012.] http://www.lcc.uma.es/~av/Docencia/Doctorado/tema1.pdf..
13. **Buschmann, Meunier, Rohnert, Sommerlad, Stal - Wiley.** *A System of Patterns* . 2012.
14. **Bascón Pantoja, Ernesto.** *El patrón de diseño modelo -vista - controlador (MVC) y su implementación el Java Swing.* 2004.
15. **Larman, Craig.** *UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos.* 2012.
16. **Tabares, Ricardo Botero.** *Grasp Patterns and Anti-Patterns: an Object.* 2012.
17. **GAMMA, ERICH, y otros.** *Design Patterns. Elements of Reusable* . 1995.
18. **Pérez, J.E.** http://www.librosweb.es/ajax/index.html. [En línea] 2012.
19. **Corporation, A. PHP v5.2.** http://www.answers.com/topic/php. [En línea] 2012.
20. **spimeframework.** http://wiki.spimebox.com/doku.php?id=framework. *http://wiki.spimebox.com/doku.php?id=framework.* [En línea] 2012.
21. **CODEBOX.** *SAUXE.* 2012.
22. **Cutter' Blades, S., Ramsay, Colin y Frederick, Shea.** *Learning Ext JS.* s.l. : Packt Publishing, 2008.
23. **Framework, Z.** http://framework.zend.com/manual/1.12/en/manual.html. [En línea] Zend Framework, 2012.
24. **doctrine, Doctrine.** http://www.doctrine-project.org/. [En línea] 2012.
25. **Consulting, M. BPMN.** http://www.milestone.com.mx/CursoModeladoNegociosBPMN.htm. [En línea] 2012.
26. **Patricia López, F.R. UML.** http://ocw.unican.es/ensenanzas-tecnicas/ingenieria-del-software-i/materiales-de-clase-1/is1-t02-trans.pdf. [En línea] 2012.
27. **Paradigm, V. Visual Paradigm.** http://www.visual-paradigm.com/. [En línea] 2012.
28. **2.0., apache.org. Documentación del Servidor HTTP Apache.** http://httpd.apache.org/docs/2.0/es/. [En línea] 2012.
29. **Postgresq.** http://www.postgresql.org/. *http://www.postgresql.org/.* [En línea] 2012.
30. **PostgreSQL., postgresql.org.** http://www.postgresql.org/. [En línea] 2012.
31. **definicionabc.** http://www.definicionabc.com/tecnologia/navegador.php#ixzz2ImO523hk.

http://www.definicionabc.com/tecnologia/navegador.php#ixzz2ImO523hk. [En línea] 2013.
32. **Mozilla.** www.mozilla.org. [En línea] 2013.
33. **Peña Lemus, Yudisleidys and Hernández Díaz, Yuniersy.** *SIMETSE–SIstema de METricas para evaluar el diseño.* Ciudad de la Habana : s.n., 2007.
34. **Lorenz, M. y Kidd, J..** *Object-Oriented Software Metrics.* . 1994.
35. **Pressman, Roger S.** *Ingeniería del Software Un enfoque práctico.* 2006.
36. **Kawtar Benghazi, José Luis Garrido Bullejos,Manuel Noguera García,Departamento de Lenguajes y Sistemas Informáticos,Universidad de Granada.** *Introducción al Modelado de Procesos de Negocio.* 2013.
37. **Sommerville.** Aprendizaje.p://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_2_Ingenieria_de_Requisitos/5_Validacion_de_Requisitos/Manual_Validacion.pdf. [En línea] 2005.
38. **Arias Chaves, Michael.** *LA INGENIERÍA DE REQUERIMIENTOS Y SU IMPORTANCIA EN EL DESARROLLO DE PROYECTOS DE SOFTWARE.* : Redalyc, 2005. 1409-4746.
39. **Escalona, María José y Koch, Nora.** Entorno Virtual de Aprendizaje. [En línea] [Citado el: 15 de 2 de 2012.] http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_2_Ingenieria_de_Requisitos/5_Validacion_de_Requisitos/Ingenieria_de_Requisitos_para_Aplicaciones_Web.pdf. [En línea] 2013.
40. **Sandoval Carvajal, Maria Marta y García Vargas, Maria Adilia. LA TRAZABILIDAD EN EL PROCESO DE REQUERIMIENTOS DE SOFTWARE.** http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf. [En línea] 2013.
41. **Cabrera Pereira., Leisniel Ignacio y Hernández Gómez, Maylin.** *Análisis y Diseño del Módulo de Cobros y Pagos del sistema integral de gestión CEDRUX.* . Ciudad de la Habana : s.n., 2009.
42. **Alas Verdecia, Pedro Manuel.** *Modelo de Diseno.* 2012.
43. **Rivera, Javier Fernandez.** http://aurea.es/wp-content/uploads/modelodedatos.pdf. [En línea] 2013.
44. **Mancha, Universidad de Castilla La.** http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf. [En línea] Departamento de Sistemas Informáticos., 2013.
45. **DEPARTAMENTO DE ELECTRÓNICA, SISTEMAS E INFORMÁTICA COORDINACIÓN.** http://pis.unicauca.edu.co/moodle/file.php/291/Patron_Diseno_MVC.pdf. [En línea] Facultad de Ingeniería Electrónica y Telecomunicaciones. , 2013.
46. **Solem.** *Estandares de Codificacion.* 2013.
47. **Atrix.** www.consuman.com. [En línea] 2012.
48. **Ltd, Zend Technologies.** http://framework.zend.com. [En línea] 2012.
49. **SGestMan.** www.SGestMan.cu. *www.SGestMan.cu.* [En línea] 2012.
50. **Visconti, Marcelo y Astudillo, Hernán. Fundamento de Ingeniería de Software.** http://www.magma.com.ni/~jorge/upoli_uml/refs/Introducion_UML.pdf. [En línea] Magma Soft, 2013.