

Universidad de las Ciencias Informáticas

Facultad 3



**Desarrollo del módulo Control de Lubricantes para
el Sistema de 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: Luis Ángel Rojas Peña

Tutores: Ing. Miguel Ángel Sánchez Palmero
Ing. Leidy Ramos González
Ing. Yanerkys Cabrera Martínez

La Habana, Junio de 2015

“Año 57 de la Revolución”

Declaración de autoría

DECLARACIÓN DE AUTORÍA

Declaro Luis Ángel Rojas Peña 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 junio del año 2015.

Luis Ángel Rojas Peña

Ing. Miguel Ángel Sánchez Palmero

Ing. Leidy Ramos González

Ing. Yanerkys Cabrera Martínez

DATOS DE CONTACTO

Autor:

- **Nombre y apellidos:** Luis Ángel Rojas Peña
- **Email:** lapena@estudiantes.uci.cu
- **Institución:** Universidad de las Ciencias Informáticas (UCI)
- **Dirección:** Carretera a San Antonio de los Baños, Km 2 1/2, Reparto Torrens, Boyeros

Tutores:

- **Nombre y apellidos:** Ing. Miguel Ángel Sánchez Palmero
- **Email:** masanchez@uci.cu
- **Situación laboral:** Especialista B en Ciencias Informáticas
- **Institución:** Universidad de las Ciencias Informáticas (UCI)
- **Dirección:** Carretera a San Antonio de los Baños, Km 2 1/2, Reparto Torrens, Boyeros

- **Nombre y apellidos:** Ing. Leidy Ramos González
- **Email:** lramosq@uci.cu
- **Situación laboral:** Especialista A en Ciencias Informáticas
- **Institución:** Universidad de las Ciencias Informáticas (UCI)
- **Dirección:** Carretera a San Antonio de los Baños, Km 2 1/2, Reparto Torrens, Boyeros

- **Nombre y apellidos:** Ing. Yanerkys Cabrera Martínez
- **Email:** ycabrera@uci.cu
- **Situación laboral:** Recién Graduado en Adiestramiento
- **Institución:** Universidad de las Ciencias Informáticas (UCI)
- **Dirección:** Carretera a San Antonio de los Baños, Km 2 1/2, Reparto Torrens, Boyeros



“No creo que seamos parientes muy cercanos, pero si usted es capaz de temblar de indignación cada vez que se comete una injusticia en el mundo, somos compañeros, que es más importante.”

Ernesto “Che” Guevara

Agradecimientos

AGRADECIMIENTOS

Una meta más en mi vida se ha cumplido y quisiera agradecer a todas aquellas personas que de una forma u otra ayudaron a allanar mi camino para que me encontrara hoy a estas instancias.

Tengo dos agradecimientos muy especiales con los que doy inicio a la larga lista de personas a las que debo gratitud por este maravilloso resultado.

El primero es para las dos personas más importantes en mi vida: mis padres. Quienes me han heredado el tesoro más valioso que puede dársele a un hijo: su amor. A quienes sin estimar esfuerzo alguno han dedicado gran parte de su vida para formarme y educarme. Nunca podré pagar todos sus desvelos ni aún con las riquezas más grandes del mundo. Por todo esto y mucho más, muchas gracias mamá y papi.

El segundo está dedicado a mi prima Yanet y su familia por todo su apoyo durante estos 5 años. Por abrirme las puertas de su casa y hacerme sentir que estaba en mi propia casa. Por estar al tanto de mí en todo momento y estar presente cada vez que la necesité. No tengo palabras para agradecer tanto amor y dedicación, gracias mi prima y gracias también a Adriana, Alejandra y Luis.

A mis tutores, a Yanerkys y Leidý por todo su apoyo, sacrificio y paciencia, en especial a Migue, que me ayudó incondicionalmente, sacrificando incluso horas de descanso para lograr culminar este trabajo con la calidad requerida. Sin su ayuda no hubiese sido posible este resultado. Mil palabras no bastarían para agradecerles todo su esfuerzo, muchas gracias.

Al tribunal y a mi oponente Olga que sus señalamientos me ayudaron a estar más preparado y lograr un trabajo de mayor calidad.

A Martha y Mato, muchas gracias por atenderme cada vez que tuve alguna duda o problema con el proyecto.

A todos mis compañeros de grupo por compartir conmigo ese espacio donde, además de recibir las clases, fueron creciendo innumerables sentimientos hermosos que voy a llevar junto a mí toda mi vida y donde he encontrado

Agradecimientos

muy buenos amigos. En especial a aquellos que han compartido más de cerca conmigo tantos momentos buenos y otros no tan buenos, llegando a ser como hermanos para mí. Muchas gracias a: Carlos José, Leo, Roberto, Chandý y Luísky.

Muchas gracias a Rigo y Alfredo por compartir apartamento conmigo y estar presente en cada momento que los necesité.

A mis amigas Yuliet, Rosalía y Dayanis. Muchas gracias por todas las cosas lindas que vivimos en este último año.

A mis compañeros del equipo de pelota, tanto de la Facultad como el UCI, por compartir momentos inolvidables conmigo y hacer mi estancia en la universidad un poco más amena, haciendo una de las cosas que más me gusta en la vida: jugar al béisbol.

A todos los profesores de mi larga vida de estudiante que de una forma u otra han aportado su granito de arena para formarme como un buen profesional.

A mi hermana Yuri, mi sobrino Yoelito y mi cuñado Joel por todo el esfuerzo y apoyo incondicional que me han brindado en el transcurso de mi vida y mis estudios. Muchas gracias.

A mi tía Edilia y mi prima Yamila por estar siempre al tanto de mi carrera y ser una fuente de apoyo para mis padres mientras estuve lejos todo este tiempo.

A toda mi familia, en especial a mis primos: Yoy, Mary, Egnier, Yaniel, Elaine, Javier, Dary y Deno por todo su apoyo, y su preocupación, por estar siempre al tanto de mi desempeño en la universidad.

A todos mis amigos del barrio, especialmente para mis hermanos: Katia, Yadier, Rebeca y Ediel que siempre han estado presente brindándome todo su apoyo incondicional y compartiendo maravillosos momentos en el poco tiempo que estuve en mi casa durante estos 5 años. Muchas gracias a todos.

DEDICATORIA

A mis padres por todo su amor, comprensión, apoyo y sacrificio durante estos 23 años. Gracias a ustedes he conseguido llegar hasta aquí y convertir en una realidad, lo que hasta ayer fue un sueño. Todos mis logros los debo a ustedes y este resultado es suyo también. Es un privilegio ser su hijo, son los mejores padres del mundo.

RESUMEN

El control de lubricantes aunque constituye una parte pequeña dentro de la actividad de mantenimiento, tiene una gran relevancia pues representa un factor vital para el correcto funcionamiento de la maquinaria. Su adecuado tratamiento asegura la calidad de los activos y genera beneficios tales como: el incremento de producción, la disminución de averías y la reducción de consumos.

En el departamento de Aplicaciones de Gestión Empresarial perteneciente al Centro de Informatización de Entidades, se desarrolla el Sistema de Control de Flotas y Mantenimiento para la gestión de los procesos que se realizan en la Dirección de Transporte de la Universidad de las Ciencias Informáticas. El presente trabajo contribuye a la realización del mismo a través del desarrollo del módulo Control de Lubricantes de dicho sistema, con el objetivo de obtener funcionalidades que permitan la inserción, análisis y recuperación de toda la información referente a los rellenos de lubricantes realizados a los vehículos de esta área.

El desarrollo del módulo Control de Lubricantes contribuye al ahorro de cuantiosos recursos materiales por parte del cliente, facilitando su trabajo y la toma de decisiones mediante reportes que le posibilitan controlar la información referente a los rellenos de lubricantes.

Palabras clave: control, flota, lubricante, mantenimiento, vehículo.

ÍNDICE

INTRODUCCIÓN.....	1
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA	4
Introducción	4
1.1. Gestión de lubricantes.....	4
1.2. Análisis del estado del arte	5
1.3. Metodología de desarrollo de software	8
1.4. Técnicas de captura de requisitos	9
1.5. Técnicas de validación de requisitos	10
1.6. Arquitectura de software basada en componentes	11
1.7. Patrones de diseño	11
1.8. Tecnologías	13
1.9. Herramientas	16
1.10. Métricas para la validación del diseño	18
1.11. Pruebas de software.....	19
Conclusiones parciales del capítulo	20
CAPÍTULO 2. MODELADO DE NEGOCIO Y REQUISITOS	21
Introducción	21
2.1. Mapa de procesos	21
2.2. Descripción de procesos de negocio	22
2.3. Validación del modelado de negocio.....	23
2.4. Modelo conceptual.....	24
2.5. Requisitos de software	25
Conclusiones parciales del capítulo	35
CAPÍTULO 3. DISEÑO, IMPLEMENTACIÓN Y PRUEBAS	36
Introducción	36
3.1. Diseño	36
3.2. Diseño del sistema.....	40
3.3. Implementación	46
3.4. Pruebas.....	50
Conclusiones parciales del capítulo	56
CONCLUSIONES GENERALES.....	57
RECOMENDACIONES.....	58
REFERENCIAS BIBLIOGRÁFICAS	59
ANEXOS	62

INTRODUCCIÓN

En la actualidad la sociedad avanza vertiginosamente impulsada por el desarrollo de las Tecnologías de la Información y las Comunicaciones, donde paulatinamente se torna más necesario informatizar los procesos que se realizan cotidianamente. De esta manera la información se encuentra más centralizada, segura y de fácil acceso, lo cual contribuye a la reducción de los tiempos de respuesta de las organizaciones, así como a la toma de sus decisiones.

Los sistemas de gestión vehicular no están ajenos a esta tendencia. Controlar el mantenimiento de cada uno de los vehículos de la flota es de gran importancia pues evita averías, reduce las probabilidades de accidentes y alarga su vida útil, de ahí el significado que tiene para las entidades que poseen una flota de vehículos, llevar estos procesos al mundo de los ordenadores. Dentro de las áreas con que estos cuentan se encuentra el control de lubricantes. Su adecuado tratamiento asegura la calidad de sus activos, genera beneficios tales como: el incremento de producción, la disminución de averías y la reducción de consumos, además garantiza el cumplimiento de las políticas de medio ambiente, seguridad y salud. El control de lubricantes aunque constituye una parte pequeña dentro de la actividad de mantenimiento, tiene una gran relevancia pues representa un factor vital para el correcto funcionamiento de la maquinaria. Una correcta gestión y sistematización del plan de lubricación asegura la disponibilidad de los equipos y ayuda a reducir los costes de mantenimiento y consecuentemente los de producción de la empresa (1).

Como parte de la política de informatización de la sociedad cubana, el Centro de Informatización de Entidades (CEIGE), perteneciente a la Universidad de las Ciencias Informáticas (UCI) se encuentra desarrollando el Sistema de Control de Flotas y Mantenimiento para la Dirección de Transporte de la propia universidad. Dicho sistema pretende informatizar los principales procesos que se llevan a cabo en esta área. Uno de los procesos que aún no cuenta con funcionalidades para su apoyo es el proceso de Control de Lubricantes.

La Dirección de Transporte de la UCI actualmente cuenta con su propio parque vehicular, lugar en el cual se dificulta el control del proceso por la gran flota de vehículos que presenta. Los trabajadores que allí laboran desarrollan esta tarea de forma manual y en algunos casos con iniciativas en formato Excel. Esto, además de tornarse demasiado engorroso, trae como consecuencia la pérdida de información, ya sea por errores humanos o por el deterioro de los archivos. Además hace que el flujo del proceso sea lento debido al gran volumen de información que se maneja. Muchos de estos problemas provocan que los vehículos reciban el relleno de lubricantes fuera de fecha o en ocasiones cuando ocurre alguna rotura, lo que

Introducción

atenta contra la vida útil del vehículo y provoca gastos no planificados para la entidad por concepto de lubricantes y piezas.

A partir de la situación antes planteada se pudo definir el siguiente **problema a resolver**: ¿cómo mejorar la gestión del proceso Control de Lubricantes de la Dirección de Transporte de la UCI?

Se plantea como **objeto de estudio**: Control de Lubricantes en los sistemas de gestión de flotas y mantenimiento vehicular.

Y el **campo de acción**: proceso Control de Lubricantes en la Dirección de Transporte de la UCI.

Para dar solución al problema anterior se define como **objetivo general**: desarrollar el módulo Control de Lubricantes del Sistema de Control de Flotas y Mantenimiento de la Dirección de Transporte de la UCI.

Del cual se derivan los siguientes **objetivos específicos**:

- Elaborar el marco teórico de la investigación relativo al control de lubricantes en los sistemas de control de flotas de vehículos.
- Desarrollar el módulo Control de Lubricantes del sistema Control de Flotas y Mantenimiento.
- Validar la solución mediante la aplicación de técnicas, métricas y pruebas.

Analizados los aspectos anteriores se puede plantear la siguiente **idea a defender**: si se desarrolla el módulo Control de Lubricantes del Sistema de Control de Flotas y Mantenimiento entonces se mejora la gestión del proceso Control de Lubricantes en la Dirección de Transporte de la UCI.

Para el desarrollo del presente trabajo se utilizaron los siguientes **métodos científicos de investigación**:

Histórico Lógico: a través de este método se pudo realizar un estudio de los antecedentes de la gestión del proceso de Control de Lubricantes, centrándose en el desarrollo de este proceso en la Dirección de Transporte de la UCI.

Analítico-Sintético: posibilitó separar la información obtenida sobre el proceso Control de Lubricantes en la Dirección de Transporte de la UCI y de esta forma facilitar su estudio y procesamiento, permitiendo el descubrimiento de sus características generales y relaciones esenciales.

Entrevista: se utilizó como técnica cuantitativa para interactuar con el personal calificado y conocedor del proceso Control de Lubricantes en la Dirección de Transporte de la UCI,

Introducción

ayudando así a la adquisición de información detallada del mismo y a la captura de requisitos.

Observación: sirvió de base para revelar las relaciones esenciales y las características fundamentales del proceso Control de Lubricantes en la Dirección de Transporte de la UCI que fueran accesibles a la percepción.

Modelación: este método permitió crear una representación del flujo de las principales actividades del proceso Control de Lubricantes de la Dirección de Transporte de la UCI, permitiendo un mejor entendimiento del mismo por parte del autor.

Este documento está compuesto de los siguientes capítulos:

Capítulo 1. Fundamentación teórica

En este capítulo se analiza el control de lubricantes en varios sistemas de gestión de flotas vehiculares tanto nacionales como internacionales. Arribando, a partir de este análisis, a conclusiones importantes acerca de sus principales deficiencias y de 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 desarrollo, marcos de trabajo y tecnologías para el desarrollo del módulo Control de Lubricantes del Sistema de Control de Flotas y Mantenimiento.

Capítulo 2. Modelado del negocio y requisitos

En el mismo se realiza la modelación del proceso de Control de Lubricantes del área de Transporte de la UCI. Se hace alusión además a las técnicas mediante las cuales se identificaron, priorizaron y validaron los requisitos.

Capítulo 3. Diseño, implementación y pruebas

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 aseguran el correcto funcionamiento de las funcionalidades implementadas para el apoyo del proceso Control de Lubricantes del Sistema Control de Flotas y Mantenimiento para la Dirección de Transporte de la UCI.

Capítulo 1. Fundamentación teórica

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Introducción

Un lubricante es una sustancia líquida que se aplica entre dos o más piezas mecánicas o móviles para minimizar su degradación. Su gestión se basa en la creación y puesta en marcha de los planes de lubricación. A nivel mundial el proceso de control de lubricantes se maneja de diferentes maneras. El presente capítulo muestra un estudio acerca de los principales sistemas de gestión de flotas vehiculares que tiene un espacio dedicado a esta área, arribando a conclusiones que permiten esclarecer aspectos de interés para la informatización del proceso en la Dirección de Transporte de la UCI. Se realiza además un estudio de las técnicas, metodología, arquitectura, patrones, métricas, tecnologías y herramientas a utilizar, que constituyen el basamento teórico necesario para el desarrollo de la propuesta de solución.

1.1. Gestión de lubricantes

La gestión de lubricantes se basa en la confección y ejecución de los planes de lubricación. De esta manera se consigue una lubricación preventiva que cumpla con las situaciones más prácticas. Sin embargo, la gestión de lubricantes es un proceso mucho más amplio. Incluye las siguientes actividades: planificación, coordinación de las acciones de una lubricación, control de plazos y calidad de ejecución de las acciones de lubricación, control del funcionamiento real del sistema y control del costo de lubricación (2).

Al analizar las fuentes bibliográficas se encontraron diferentes conceptos utilizados para definir qué es un lubricante. Se seleccionaron los que a continuación se muestran, pues son los que más se acercan a la situación en cuestión.

Un lubricante es una sustancia líquida que una vez colocada entre dos o más piezas mecánicas o móviles, minimiza su degradación y al mismo tiempo permite la formación de una película que evita el contacto directo entre las piezas para que puedan ser usadas o funcionar sobre todo con altas temperaturas y también bajo mucha presión física o roce (3).

Se conoce como lubricante a toda sustancia sólida, semisólida o líquida, de origen animal, mineral o sintético que, puesto entre dos piezas con movimiento entre ellas, reduce el rozamiento y facilita dicho movimiento (4).

Partiendo de las definiciones anteriores se puede concluir para la presente investigación que un lubricante es una sustancia que se aplica entre las piezas móviles de los vehículos para facilitar el movimiento y reducir el rozamiento entre las mismas, permitiendo que estas trabajen a altas temperaturas y soporten una elevada carga física.

Capítulo 1. Fundamentación teórica

A partir del concepto brindado por el autor anteriormente, se puede definir en qué consiste el proceso Control de Lubricantes en la Dirección de Transporte de la UCI.

En dicha entidad el proceso se realiza utilizando los documentos Solicitud de lubricantes, Orden de trabajo y Modelo de salida de lubricantes. Siendo la Orden de trabajo el documento rector de proceso y el Modelo de salida de lubricantes donde quedan registrados los datos del relleno de lubricantes. Durante el desarrollo de este proceso intervienen: el chofer del vehículo, el especialista en lubricantes, el subdirector técnico del taller y el engrasador. A través del mismo, la entidad controla que se realicen los rellenos de lubricantes a los vehículos en el momento indicado y de forma correcta, además de conocer los índices de consumo de estas sustancias (5).

Los sistemas de gestión de flotas a nivel mundial permiten manejar el proceso de control de lubricantes de diferentes maneras. Además difieren en los recursos utilizados para mostrar los resultados que se obtienen a partir del mismo. Dada esta diversidad, es de primordial interés para el desarrollo del presente trabajo realizar un estudio de los principales sistemas de gestión vehicular que presentan un espacio dedicado al control de lubricantes.

1.2. Análisis del estado del arte

Una adecuada gestión del control de lubricantes en las empresas que cuenten con una flota vehicular asegura la calidad de sus activos y genera innumerables beneficios. A continuación se hace una breve reseña de sistemas tanto nacionales como internacionales de gestión vehicular que presentan características similares.

1.2.1. Sistemas internacionales

Internacionalmente existen muchas aplicaciones que permiten la gestión de flotas vehiculares. El presente estudio se centró en aquellos que cuentan con un espacio dedicado al control de lubricantes. Entre estos se pueden mencionar los que siguen:

ANAGO

ANAGO es un software de gestión de mantenimiento de flotas de vehículos y control de costos. Permite administrar detalladamente las reparaciones, mano de obra, pañol de repuestos, facturas de talleres de terceros, control de neumáticos, abastecimiento y consumo de combustibles y lubricantes. Administra las revisiones técnicas, el mantenimiento preventivo y vencimientos, a través de una agenda que genera alertas automáticas. Los vencimientos pueden programarse por fecha, contratos, kilómetros u horas de uso. Cuenta con un centro de reportes y estadísticas configurables que permite visualizar, imprimir y exportar informaciones tales como: consumo de combustibles y lubricantes, desgaste de neumáticos, costos de reparaciones por proveedor, historial de reparaciones, utilización de

Capítulo 1. Fundamentación teórica

repuestos y otro tipo de información útil para reducir los costos y mejorar la administración de la flota (6).

SISTEMA DE GESTIÓN DE FLOTA DE VEHÍCULOS

El Sistema de Gestión de Flota de Vehículos, desarrollado por la empresa SUPPLEST, controla todo el proceso de ingreso, mantenimiento y egreso de una flota de vehículos o un conjunto de maquinarias; abarcando los siguientes aspectos: consumo de combustible y lubricantes, mantenimiento preventivo, correctivo y mayor, impuestos y seguros y requisitos de piezas. El sistema está compuesto por los siguientes módulos: administración de usuarios y permisos, gestión de vehículos, gestión de proveedores, consumo y mantenimiento, áreas y personas, configuración de listas y registro de talonarios de consumo. Es una aplicación fácil de aprender y de utilizar, tiene conexión con otros sistemas, maneja usuarios y permisos, utiliza base de datos relacional y exporta las planillas a Microsoft Excel y a otros formatos (7).

1.2.2. Sistemas nacionales

Cuba no está ajena a los procesos de gestión vehicular, por lo que también se pueden mencionar algunos sistemas de factura nacional que permiten a las entidades cubanas manejar de forma más fácil el control y administración de las tareas relacionadas con sus vehículos, entre estas aplicaciones se encuentran:

GCARS: SISTEMA DE GESTIÓN VEHICULAR V. 1.0

GCARS es una aplicación web desarrollada por la empresa productora de software DATYS. Es un producto diseñado para el control de los vehículos y equipos tecnológicos. Cuenta con un conjunto de funcionalidades y reportes orientados a mejorar la gestión del parque automotriz, del combustible y de los recursos complementarios (accesorios, agregados y gestión de lubricantes). Ofrece un conjunto de funcionalidades y permite además la generación de reportes tanto en formato PDF, como a través de gráficos (8).

GCARS lleva a cabo el control de lubricantes a través de la funcionalidad Gestión de Combustibles y Lubricantes. Entre sus operaciones permite ver el consumo de lubricantes, haciendo reportes de operaciones y reportes gráficos por varios criterios, entre los que se encuentran: centro de costo, vehículo y clasificación.

SISCOMPA

Siscompa.Net es un software de control de flotas creado por la empresa TranSoft. Brinda información confiable sobre todo el parque automotor que poseen las bases, con respecto a su estado técnico, historial y explotación. Planificando el control de toda la operación en taller, mantenimientos, actividad de tráfico, seguridad automotor y el control de los portadores energéticos. Permite potenciar la actividad de ahorros en combustibles,

Capítulo 1. Fundamentación teórica

lubricantes, gomas y baterías. Ofrece a los directivos la información necesaria para la toma de decisiones, siendo una poderosa herramienta de dirección (9).

1.2.3. Conclusiones del análisis del estado del arte

A partir de la información obtenida acerca del proceso Control de Lubricantes en la Dirección de Transporte de la UCI, se realizó un estudio de los sistemas anteriormente mencionados. El mismo tuvo como objetivo conocer de qué manera realizan este proceso. Además de analizar entre sus características, las que representaran condiciones deseables para la propuesta de solución, así como aquellas que indiquen deficiencias que la misma no debe presentar. Dicho estudio arrojó los siguientes resultados:

Tabla 1. Criterios comparativos a evaluar en los sistemas

Sistema / Criterio	ANAGO	SGFV	GCARS	SISCOMPA
Multiplataforma	No	No	No	No
Modularidad	No	No	No	No
Licencia privativa	Sí	Sí	Sí	Sí
Gestión de lubricantes	Sí	Sí	Sí	Sí
Control del consumo de lubricantes	Sí	Sí	Sí	Sí
Generación de reportes	Sí	No	Sí	No

Los sistemas en cuestión presentan funcionalidades y características de gran relevancia para la propuesta de solución las cuales que son necesarias para realizar un correcto control de lubricantes:

- Todos los sistemas analizados presentan funcionalidades para realizar la gestión de lubricantes.
- Los sistemas estudiados, en su totalidad, cuentan con funcionalidades que brindan la posibilidad de mantener, de un modo u otro, un control exhaustivo del consumo de lubricantes.
- Los sistemas **ANAGO** y **GCARS** permiten generar reportes de consumo de lubricantes atendiendo a diferentes criterios.

Capítulo 1. Fundamentación teórica

Sin embargo, el estudio minucioso de dichos sistemas permitió encontrar características presentes en los mismos que indican deficiencias que deben evitarse en el desarrollo de la propuesta de solución:

- Ninguno de los sistemas estudiados presenta un módulo que se dedique específicamente al control de lubricantes, es común en todos encontrar las funcionalidades destinadas a dicho proceso dentro de otro módulo.
- Los sistemas en cuestión se ejecutan solamente sobre el sistema operativo Windows y utilizan tecnologías privativas, aspecto que afecta el paradigma de soberanía tecnológica y migración a software libre por el que apuesta el país.
- Los sistemas analizados, en su totalidad, trabajan bajo licencia privativa.

Teniendo en cuenta lo antes expuesto, se plantea que se necesita desarrollar un módulo para el Sistema de Control de Flotas y Mantenimiento que permita llevar a cabo el proceso Control de Lubricantes en la Dirección de Transporte de la UCI. El mismo debe contar con un espacio dedicado al control del consumo de los lubricantes y debe permitir generar reportes atendiendo a los criterios que requiere la entidad. Además debe ser desarrollado utilizando tecnologías libres y que pueda ser ejecutado sobre las mismas, para así dar continuidad a la política de soberanía tecnológica vigente en el país.

1.3. Metodología de desarrollo de software

Para guiar el proceso de desarrollo de software se utilizó una variación realizada por la UCI a la metodología Proceso Unificado Ágil (AUP, por sus siglas en inglés). Esta variación cuenta con 8 disciplinas: Modelado de Negocio, Requisitos, Análisis y Diseño, Implementación, Pruebas Internas, Pruebas de Liberación, Pruebas de Aceptación y el Despliegue, que se considera opcional. De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) el ciclo de vida de los proyectos de la UCI mantiene la fase de Inicio, unifica las restantes 3 fases en una sola, llamada Ejecución y se agrega la fase de Cierre (10).

Inicio: se llevan a cabo las actividades relacionadas con la planeación del proyecto. 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 (10).

Ejecución: 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 modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto. Durante esta fase el producto

Capítulo 1. Fundamentación teórica

es transferido al ambiente de los usuarios finales o entregado al cliente. Además, en dicha transición se capacita a los usuarios finales sobre la utilización del software (10).

Cierre: en esta fase se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto (10).

1.4. Técnicas de captura de requisitos

La captura de requisitos es el proceso de recoger información sobre el sistema propuesto y los existentes y extraer los requisitos del usuario y del sistema a partir de esta información (11). Son varios los autores que han dado su criterio acerca de las técnicas a utilizar para la captura de requisitos, por ejemplo:

Ian Sommerville plantea que se puede realizar a través de:

Entrevistas: consisten en una serie de preguntas realizadas por el equipo de ingeniería de requisitos a los stakeholders sobre el sistema que utilizan y el sistema a desarrollar. Estas entrevistas se dividen en dos grupos: cerradas, donde existe un grupo de preguntas predefinidas y abiertas donde no hay un programa definido (11).

Escenarios: las personas encuentran más fácil dar ejemplos de la vida real que definiciones abstractas. Pueden entender y criticar un escenario de cómo podría interactuar con un software. Los ingenieros de requisitos pueden utilizar esta información para formular los requisitos reales del sistema (11).

Casos de uso: es una técnica que se basa en escenarios para la obtención de requisitos. Identifican el tipo de interacción y los actores involucrados (11).

Roger S. Pressman hace alusión a las siguientes técnicas:

Reunión o entrevista preliminar: es la primera interacción entre el cliente y el ingeniero de software (12).

Técnicas para facilitar las especificaciones de la aplicación (TFEA): este enfoque es partidario de la creación de un equipo conjunto de clientes y desarrolladores que trabajan juntos para identificar el problema, proponer soluciones, negociar diferentes enfoques y especificar un conjunto preliminar de requisitos de la solución (12).

Despliegue de la funcionalidad calidad (DFC): es una técnica de gestión de calidad que traduce las necesidades del cliente en requisitos técnicos del software (12).

Casos de uso: una vez recopilados los requisitos, el ingeniero del software puede crear un conjunto de escenarios que identifiquen una línea de utilización para el sistema que va a ser construido (12).

Capítulo 1. Fundamentación teórica

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 (13).

Existen otras técnicas que no se encuentran entre las planteadas por estos grandes de la Ingeniería de Software como es el caso de:

Análisis de documentación: permite analizar sistemas anteriores similares, convirtiéndolos en fuente de información relacionada a los procedimientos de las operaciones. También existen otros documentos en las organizaciones que posibilitan obtener información sobre prácticas corrientes de las actividades que el sistema debe hacer y que serán útiles al escribir una especificación de requisitos (14).

De las técnicas anteriormente mencionadas se decidió que para la captura de los requisitos de la propuesta de solución se utilizarán la entrevista, la tormenta de ideas y el análisis de documentación.

1.5. Técnicas de validación de requisitos

La validación de requisitos examina las especificaciones para asegurar que todos los requisitos del sistema han sido establecidos sin ambigüedad, sin inconsistencias, sin omisiones, que los errores detectados hayan sido corregidos, y que el resultado del trabajo se ajusta a los estándares establecidos para el proceso, el proyecto y el producto (12). A continuación se relacionan las técnicas que han propuesto diferentes autores.

Ian Sommerville propone las siguientes:

Revisiones de requisitos: los requisitos son analizados sistemáticamente por un equipo de revisores (11).

Construcción de prototipos: se muestra un modelo ejecutable del sistema a los usuarios finales y a los clientes. Éstos pueden experimentar con este modelo para ver si cumple sus necesidades reales (11).

Generación de casos de prueba: los requisitos deben poder probarse. Si las pruebas para éstos se conciben como parte del proceso de validación, a menudo revela los problemas en los requisitos (11).

Roger S. Pressman plantea la técnica siguiente:

Revisión técnica formal: el equipo de revisión incluye ingenieros del sistema, clientes, usuarios y otros intervinientes que examinan la especificación del sistema buscando errores en el contenido o en la interpretación, áreas donde se necesitan aclaraciones, información

Capítulo 1. Fundamentación teórica

incompleta, inconsistencias, requisitos contradictorios, o requisitos imposibles o inalcanzables (12).

Se seleccionaron las técnicas construcción de prototipos, generación de casos de prueba y revisión técnica formal para validar los requisitos capturados.

1.6. Arquitectura de software basada en componentes

Para el desarrollo del Sistema de Control de Flotas y Mantenimiento fue definido, por la dirección del CEIGE, la utilización de la arquitectura basada en componentes. La misma describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas. Esto provee un nivel de abstracción mayor que los principios de orientación por objetos y no se enfoca en asuntos específicos de los objetos como los protocolos de comunicación y la forma como se comparte el estado (15).

1.6.1. Patrón 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. Estos 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 (16).

El patrón arquitectónico Modelo-Vista-Controlador (MVC) 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 transmitirlos hacia la vista (17).

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 (16).

Capítulo 1. Fundamentación teórica

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 (18). De éstos se utilizaron los que a continuación se relacionan:

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 la clase que lo implementa recibe los datos del usuario y los envía a las distintas clases según el método llamado (18).

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 (19).

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 (19).

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 (19).

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 (19).

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 (20). De ellos se utilizaron los que se muestran a continuación:

Capítulo 1. Fundamentación teórica

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 (20).

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 (20).

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 (20).

Fachada: define una interfaz de alto nivel para un conjunto de interfaces de un subsistema que hace que éste sea más fácil de utilizar (20).

Comportamiento

Ofrecen soluciones respecto a la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan (20).

Cadena de responsabilidad: permite establecer la línea que deben llevar los mensajes para que los objetos realicen la tarea indicada (20).

1.8. Tecnologías

A la hora de desarrollar la propuesta de solución se utilizaron un conjunto de tecnologías, las mismas se relacionan a continuación.

1.8.1. Notación de Modelado de Procesos de Negocio: BPMN 2.0

BPMN (*Business Process Model and Notation*, por sus siglas en inglés) es una notación gráfica que describe la lógica de los pasos de un proceso de negocio. Esta notación ha sido especialmente diseñada para coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes de las diferentes actividades. Proporciona un lenguaje común para que las partes involucradas puedan comunicar los procesos de forma clara, completa y eficiente. De esta forma define la notación y semántica de un diagrama de procesos de negocio. Es un estándar internacional de modelado de procesos muy aceptado por la comunidad. Es independiente de cualquier metodología de modelado de procesos. Crea un puente estandarizado para disminuir la brecha entre los procesos de negocio y la implementación de estos. Permite modelar los procesos de una manera unificada y estandarizada brindando un entendimiento a todas las personas de una organización (21).

Capítulo 1. Fundamentación teórica

1.8.2. Lenguaje Unificado de Modelado: UML 2.0

UML (*Unified Modeling Language*, por sus siglas en inglés) es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema de software. Proporciona una forma estándar de escribir los planos de un sistema, cubriendo elementos conceptuales y concretos tales como: procesos del negocio, funciones del sistema, clases escritas en un lenguaje de programación específico, esquemas de bases de datos y componentes de software reutilizables. Además brinda las siguientes ventajas: mayor rigor en la especificación, verificación y validación del modelo realizado, automatización de procesos y generación de código a partir de los modelos y viceversa (a partir del código fuente generar los modelos). Esto permite que el modelo y el código estén actualizados, manteniendo la visión en el diseño de la estructura de un proyecto (22).

1.8.3. AJAX

AJAX es un acrónimo de *Asynchronous JavaScript + XML*, que se puede traducir como "JavaScript asíncrono + XML". Ajax no es una tecnología en sí mismo. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes, estas son: XHTML y CSS, para crear una presentación basada en estándares, DOM, para la interacción y manipulación dinámica de la presentación, XML, XSLT y JSON, para el intercambio y la manipulación de información, XMLHttpRequest, para el intercambio asíncrono de información y JavaScript, para unir todas las demás tecnologías. Permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano (23).

1.8.4. Lenguajes de programación

Un lenguaje de programación es aquella estructura que, con una cierta base sintáctica y semántica, imparte distintas instrucciones a un programa de computadora (24). Para el desarrollo de la propuesta de solución se utilizaron los que siguen:

1.8.4.1. Lenguaje de programación del lado del servidor

PHP acrónimo de *Hypertext Preprocessor*, es un lenguaje de script incrustado dentro del HTML. Una de sus características más potentes es su soporte para gran cantidad de bases de datos, entre ellas pueden mencionarse InterBase, mSQL, MySQL, Oracle, Informix y PostgreSQL. Permite la integración con las varias bibliotecas externas, que permiten que el desarrollador haga casi cualquier cosa, desde generar documentos en pdf hasta analizar código XML. Ofrece una solución simple y universal para las paginaciones dinámicas de la Web de fácil programación. Debido a su amplia distribución PHP está perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto,

Capítulo 1. Fundamentación teórica

PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparen rápidamente. Además permite el uso de la programación orientada a objetos (25).

Su empleo para el desarrollo de la propuesta de solución se debe a que el marco de trabajo Sauxe, sobre el cual se implementa la misma, fue realizado con este lenguaje de programación.

1.8.4.2. Lenguaje de programación del lado del cliente

El lenguaje de programación del lado del cliente empleado para el desarrollo del módulo Control de Lubricantes es JavaScript, el mismo es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientada a objetos e imperativa (26).

1.8.5. Marco de trabajo Sauxe

El marco de trabajo Sauxe desarrollado por el departamento de Tecnología del centro CEIGE, da solución a un sin número de escenarios o aspectos arquitectónicos como: gestión y configuración dinámica de caché, integración de componentes de forma distribuida o no distribuida, acceso a bases de datos a través de una capa de abstracción, gestión de concurrencia de recursos, administración centralizada de transacciones, gestión dinámica de las trazas generadas por los sistemas, implementación de mecanismos de autenticación y autorización, implementación de mecanismos de mensajería y control de excepciones, gestión y configuración dinámica de pre condiciones, pos condiciones y validaciones de variables, gestión y configuración de flujos de trabajo, visualización de las funcionalidades de un sistema, entre otros escenarios de alta complejidad, garantizando los atributos de calidad de los sistemas que se desarrollen con el mismo. Cuenta con una arquitectura en capas que a su vez presenta en su capa superior una implementación Modelo-Vista-Controlador (27).

Siguiendo el paradigma de independencia tecnológica por el cual apuesta el país, Sauxe reutiliza las siguientes tecnologías libres:

1.8.5.1. Zend Framework 1.2

Zend Framework es un framework de código abierto para el desarrollo de aplicaciones y servicios web con PHP, desarrollado por la empresa Zend Technologies. Se implementa utilizando el 100% de código orientado a objetos. Su estructura de componentes es algo único; cada componente está diseñado con pocas dependencias de otros componentes. Esta arquitectura de acoplamiento flexible permite a los desarrolladores usar los

Capítulo 1. Fundamentación teórica

componentes de forma individual. Mientras que pueden ser utilizadas por separado, sus componentes en la biblioteca estándar forman un marco de aplicación web potente y extensible cuando se combinan. Ofrece una aplicación robusta y de alto rendimiento Modelo Vista Controlador y una abstracción de base de datos que es fácil de usar (28).

1.8.5.2. ExtJS 3.4

ExtJS fue desarrollado por un equipo unificado de programadores. Brilla en la fabricación de aplicaciones web fáciles de usar e intuitivas. Es una de las bibliotecas de presentación más poderosas para el desarrollo de aplicaciones. Proporciona una interfaz de usuario fácil de usar, rica, muy parecida a lo que encontraría en una aplicación de escritorio. Esto permite que los desarrolladores web se centren en la funcionalidad de las aplicaciones web en lugar de las advertencias técnicas (29).

1.8.5.3. Doctrine Framework 1.2.2

Doctrine es un potente sistema ORM (*Object Relational Mapper*, por sus siglas en inglés) para PHP. Una de sus principales características es la opción de escribir consultas de bases de datos orientadas a objetos en su propio dialecto SQL llamado DQL (del inglés, *Doctrine Query Language*). Proporciona a los desarrolladores una poderosa alternativa a SQL que mantiene la flexibilidad sin necesidad de duplicación de código innecesaria. Cuenta con una gran comunidad y la integración con muchos marcos de trabajo diferentes (30).

1.9. Herramientas

Para el desarrollo de la propuesta de solución se utilizaron las herramientas que se relacionan a continuación.

1.9.1. Repositorio para el control de versiones Subversion 1.8

Subversion un sistema de control de versiones libre y de código abierto. Maneja ficheros y directorios a través del tiempo, tiene un árbol de ficheros en un repositorio central, permitiendo recuperar versiones antiguas de los datos o examinar el historial de cambios de los mismos. Es un sistema general que puede ser usado para administrar cualquier conjunto de ficheros ya sea código fuente o de otro tipo. Implementa un sistema de ficheros de versionado virtual que sigue los cambios sobre árboles de directorios completos a través del tiempo. Permite añadir, borrar, copiar y renombrar ficheros y directorios. Los ficheros nuevos que son añadidos comienzan con un historial nuevo y limpio. Posibilita a los desarrolladores construir y enviar los cambios como fragmentos lógicos e impide que ocurran problemas cuando sólo una parte de los cambios enviados lo hace con éxito (31).

Capítulo 1. Fundamentación teórica

1.9.2. Biblioteca de clases TCPDF 2.1

TCPDF es una biblioteca de clases para el lenguaje de programación PHP la cual permite crear ficheros PDF de forma dinámica y rápida. Dos de las cualidades más apreciadas de esta clase, es su simplicidad a la hora de crear archivos PDF y la capacidad de interpretar código XHTML. Contiene métodos y funciones para justificar el texto del documento, manipular los encabezados y numeración automática de las páginas, alinear, incluir imágenes, aplicar determinados colores, incluir enlaces web, cifrar el documento e incluir código en JavaScript (32).

1.9.3. Herramienta CASE Visual Paradigm 8.0

Visual Paradigm es una herramienta para desarrollo de aplicaciones utilizando modelado UML de alta confiabilidad y estabilidad en el desarrollo orientado a objetos. Tiene soporte multiplataforma y 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. Permite modelar el negocio orientado a procesos usando la notación BPMN (33).

1.9.4. Servidor de aplicaciones Apache 2.2

El servidor Apache HTTP, también llamado Apache, es un servidor web HTTP de código abierto para la creación de páginas y servicios web. La utilización del mismo se debe a que es un servidor multiplataforma, gratuito, muy robusto y que destaca por su seguridad y rendimiento (34).

1.9.5. Sistema gestor de bases de datos PostgreSQL 9.1

PostgreSQL es un potente servidor de base de datos, de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación por su fiabilidad, integridad de datos y corrección. Es multiplataforma y tiene soporte completo para claves foráneas, combinaciones, vistas, triggers y procedimientos almacenados. Presenta además interfaces de programación nativas para varios lenguajes y una documentación excepcional (35).

1.9.5.1. PgAdmin III

PgAdmin es un sistema integral de diseño y gestión de bases de datos PostgreSQL. Es una aplicación multiplataforma que puede utilizarse en Linux, FreeBSD, Solaris, Mac OSX y Windows. Está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL sencillas hasta el desarrollo de bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y hace fácil su administración. También incluye un editor de resaltado de sintaxis SQL, un editor de código del lado del

Capítulo 1. Fundamentación teórica

servidor y un agente de planificación de tareas de SQL. La conexión del servidor puede hacerse mediante TCP/IP o Unix Domain sockets (en plataformas Unix), y puede ser encriptado el SSL para la seguridad. PgAdmin es desarrollado por una comunidad de expertos de PostgreSQL en todo el mundo y está disponible en más de una docena de idiomas. Es un software libre publicado bajo la licencia PostgreSQL (36).

1.9.6. Entorno de desarrollo integrado Netbeans 7.3

Netbeans es un entorno de desarrollo gratuito y de código abierto. Permite el uso de un amplio rango de tecnologías de desarrollo tanto para escritorio, como aplicaciones Web, o para dispositivos móviles. Da soporte a varias tecnologías, entre ellas se pueden mencionar: Java, PHP, C/C++ y HTML5. Puede instalarse en varios sistemas operativos. Simplifica la gestión de grandes proyectos con el uso de diferentes vistas, asistentes de ayuda, y estructurando la visualización de manera ordenada, lo que ayuda en el trabajo diario (37).

1.9.7. Navegador

Un Navegador Web es un programa que permite visualizar páginas web en la red además de acceder a otros recursos, documentos almacenados y guardar información. El Navegador se comunica con el servidor a través del protocolo HTTP y solicita el archivo en código HTML, después lo interpreta y muestra en pantalla para el usuario (38).

Mozilla Firefox 4.0 o superior

Firefox es un navegador de Internet con interfaz gráfica de usuario desarrollado por la Corporación Mozilla y un gran número de voluntarios. Permite añadir funcionalidades mediante extensiones, Firebug es precisamente el complemento que se integrará a Mozilla Firefox para ayudar a desarrollar, evaluar y depurar la propuesta de solución, 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 (39).

1.10. Métricas para la validación del diseño

Las métricas de diseño permiten medir de forma cuantitativa la calidad de los atributos internos del software. A nivel de componentes se centran en las características internas de los componentes del software con medidas que pueden ayudar al desarrollador a juzgar la calidad de un diseño a nivel de componente (40).

Lorenz y Kidd concentran las métricas basadas en clases en tres grupos: tamaño de la clase, herencia y características internas de las clases. Estas métricas están enfocadas a las características internas del diseño orientado a objeto y de esta manera, contribuyen a

Capítulo 1. Fundamentación teórica

asegurar la mantenibilidad de los productos de software. Las métricas basadas en el tamaño de la clase se enfocan en el número de variables y métodos que posee la clase, por su parte las métricas orientadas a herencia se centran en la cantidad de métodos que una subclase hereda, que añade o que reemplaza, así como su índice de especialización y por último las métricas basadas en las características internas de la clase se apoyan en el cociente entre el número total de parámetros por método y el número total de métodos (41).

1.11. Pruebas de software

El único instrumento adecuado para determinar el estatus de la calidad de un producto de software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que este cumple con los requisitos. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante técnicas de prueba (42).

Para probar la propuesta de solución con el fin de encontrar errores durante el desarrollo de la misma se emplearán las técnicas de caja blanca y caja negra, las cuales consisten en:

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 usa la estructura de control del diseño procedimental para obtener los casos de prueba. Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejerciten por lo menos una vez: todos los caminos independientes de cada módulo, todas las decisiones lógicas en sus vertientes verdadera y falsa, los bucles en sus límites y con sus límites operacionales y las estructuras internas de datos para asegurar su validez (12).

Camino Básico

Esta técnica permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecuten por lo menos una vez cada sentencia del programa (12).

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. La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca. Más bien se trata de un enfoque

Capítulo 1. Fundamentación teórica

complementario que intenta descubrir tipos de errores diferentes a los encontrados en los métodos de caja blanca. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación. A diferencia de la prueba de caja blanca, que se lleva a cabo previamente en el proceso de prueba, la prueba de caja negra tiende a aplicarse durante fases posteriores de la prueba, ya que la prueba de caja negra ignora intencionadamente la estructura de control y centra su atención en el campo de la información (12).

Partición Equivalente

Esta técnica 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. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar (12).

Conclusiones parciales del capítulo

Tras el estudio realizado en el presente capítulo se arribó a las siguientes conclusiones:

- A partir del estudio de los sistemas de gestión de flotas de vehículos, tanto nacionales como internacionales, se obtuvo información valiosa y necesaria para la informatización del proceso de Control de Lubricantes de la Dirección de Transporte de la UCI.
- Es necesario el desarrollo de un módulo para el Sistema de Control de Flota y Mantenimiento que cumpla con las necesidades de la entidad, así como con los principios de soberanía tecnológica vigentes en el país.
- Quedaron sentadas las bases teóricas necesarias para el desarrollo de la propuesta de solución.

Capítulo 2. Modelado de negocio y Requisitos

CAPÍTULO 2. MODELADO DE NEGOCIO Y REQUISITOS

Introducción

El modelado de negocio es la representación de los procesos de negocio de una empresa u organización con el objetivo de que puedan ser analizados y mejorados. En el presente capítulo quedan plasmadas las descripciones de los principales artefactos generados en esta etapa, entre ellos se encuentran: el mapa de procesos, la descripción de procesos de negocio, la validación del modelado de procesos de negocio y el modelo conceptual. También se hace alusión a los requisitos de software, refiriéndose además a los artefactos y técnicas fundamentales que se usaron para lograr que los mismos alcanzaran un estado óptimo antes de arribar a la fase de diseño de la solución.

2.1. Mapa de procesos

Se utilizó el mapa de procesos para crear una representación gráfica que permitió visualizar el proceso de Control de Lubricantes, así como sus relaciones principales. A su vez posibilitó documentar el proceso para así facilitar su descripción y entendimiento.

Los mapas de procesos se confeccionaron con el fin de representar 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. El contenido de este trabajo se enmarca en el proceso Control de Lubricantes del área de Transporte de la UCI perteneciente al nivel 1 del proceso Control de Reparaciones y Mantenimiento.

Tabla 2. Dependencia del proceso Control de Lubricantes

Nº	Nombre del proceso	Breve descripción del proceso	Nivel	Proceso padre
1	Control de Lubricantes	Este proceso es el que maneja toda la información referente a la asignación y control de los lubricantes en los vehículos.	1	Control de reparaciones y mantenimiento

Para la realización del mapa del proceso Control de Lubricantes se identificaron los artefactos de entrada y salida de este, así como el formato en que se manejan actualmente en el área de Transporte de la UCI. Se creó la matriz de procesos que contiene la relación que existe entre los mismos, exponiendo los artefactos de salida de unos que constituyen entradas para otros y viceversa (consultar documento entregable CIG-CFM-N-MTTO-i1105).

Capítulo 2. Modelado de negocio y Requisitos

Tabla 3. Artefactos generados en el proceso Control de Lubricantes

Nº	Nombre del artefacto	Breve descripción del artefacto	Formato
1	Solicitud de lubricantes	Documento que genera el chofer cuando su vehículo necesita de un relleno de lubricantes.	.doc
2	Orden de trabajo	Documento que se genera para cada vehículo, en el mismo se registran los trabajos a realizar ya sean preventivos o correctivos, los recursos materiales y piezas utilizadas y los recursos humanos que ejecutaron el trabajo.	.doc
3	Modelo de salida de lubricantes	Documento que se genera cada vez que se realiza un relleno de lubricantes a un vehículo, en este queda registrada la cantidad de lubricantes por tipo que le fue suministrada al mismo.	.doc

2.2. Descripción de procesos de negocio

El proceso de Control de Lubricantes en la Dirección de Transporte de la UCI tiene como objetivo principal registrar toda la información referente al relleno de lubricantes a los vehículos de esta área. El mismo comienza a partir de una solicitud de relleno de lubricantes realizada por un chofer para un determinado vehículo. En este momento el especialista general emite una orden de trabajo, si el propio vehículo ya ha sido lubricado en la semana en curso, esta acción debe ser autorizada por el subdirector técnico. Si es aprobada la solicitud, el especialista general procede a realizar una nueva orden de trabajo. Con la orden de trabajo realizada el chofer se presenta en el área de engrase, donde el engrasador mide los lubricantes del vehículo y según el resultado de la medición procede a suministrar la cantidad de lubricantes que este necesita. Terminado el trabajo, el engrasador procede a crear el modelo de salida de lubricantes. Finalmente se cierra la orden de trabajo con la firma del chofer como constancia de que está de acuerdo con el trabajo realizado y de esta forma concluye el proceso (5).

A continuación se muestra el diagrama de procesos de negocio correspondiente al proceso Control de Lubricantes. Para más información acerca de la descripción de procesos de negocio consultar el Anexo 1.

Capítulo 2. Modelado de negocio y Requisitos

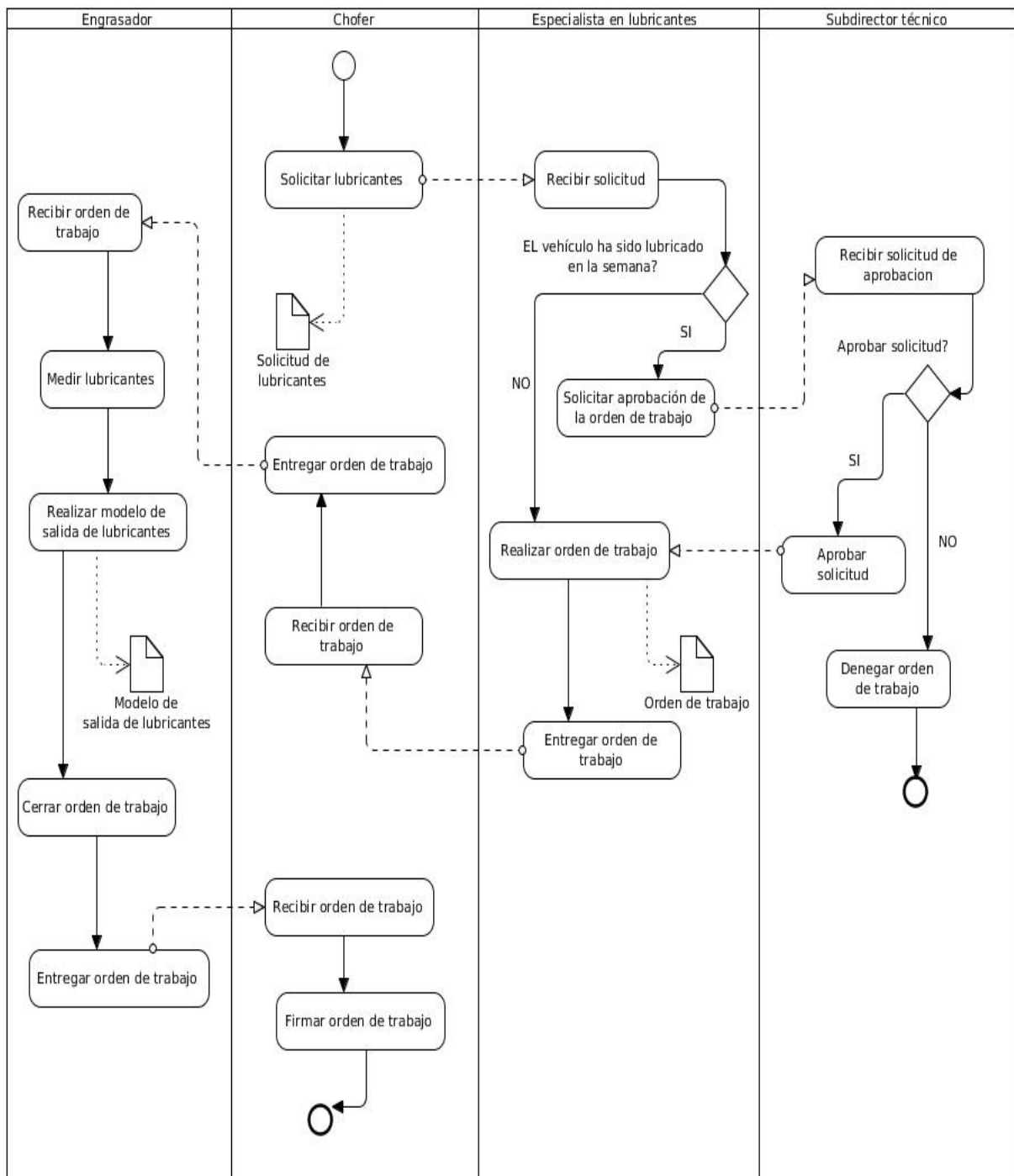


Figura 1. Diagrama de procesos de negocio del proceso Control de Lubricantes

2.3. Validación del modelado de negocio

La validación del modelado de procesos de negocio se llevó a cabo a través de la técnica Revisión Técnica Formal, donde los clientes revisaron el documento de descripción de procesos de negocio realizado, con el objetivo de validar su correcta elaboración y que el flujo de actividades del proceso estuviera en correspondencia con la información brindada.

Capítulo 2. Modelado de negocio y Requisitos

Se realizaron además varias reuniones con los especialistas de la Dirección de Transporte de la UCI y miembros del equipo de desarrollo para lograr una correcta interpretación de la información transmitida y de esta forma corregir cualquier inconformidad por parte de los clientes, los señalamientos planteados fueron recogidos y corregidos posteriormente.

2.4. Modelo conceptual

La realización del modelo conceptual del proceso de Control de Lubricantes permitió obtener una representación gráfica de los conceptos que se manejan en el mismo. Este permite documentar los principales atributos y relaciones de estos conceptos para facilitar su entendimiento. En dicho modelo se representan en forma de diagrama todos los conceptos con sus respectivos atributos, así como las relaciones que existen entre ellos. A su vez se describen por separado cada uno de los conceptos involucrados incluyendo cada uno de sus atributos.

La siguiente tabla representa un fragmento del diccionario de datos correspondiente al concepto Modelo de salida de lubricantes:

Tabla 4. Fragmento del diccionario de datos del concepto Modelo de salida de lubricantes

Descripción		Documento donde se registra la salida de los lubricantes		
Atributos				
Nombre	Descripción	Tipo	¿Puede ser nulo?	¿Es único?
nmodelosalida lubricante	Número del modelo de salida del lubricante	numérico	No	Si
fechaentrada	Fecha de entrada al taller	date	No	No
fechasalida	Fecha de salida del taller	date	No	No
tiempoentaller	Tiempo en el taller	time	No	No
horaentrada	Hora de entrada al taller	time	No	No
horasalida	Hora de salida del taller	time	No	No

A continuación se presenta una imagen del modelo conceptual del proceso Control de Lubricantes:

Capítulo 2. Modelado de negocio y Requisitos

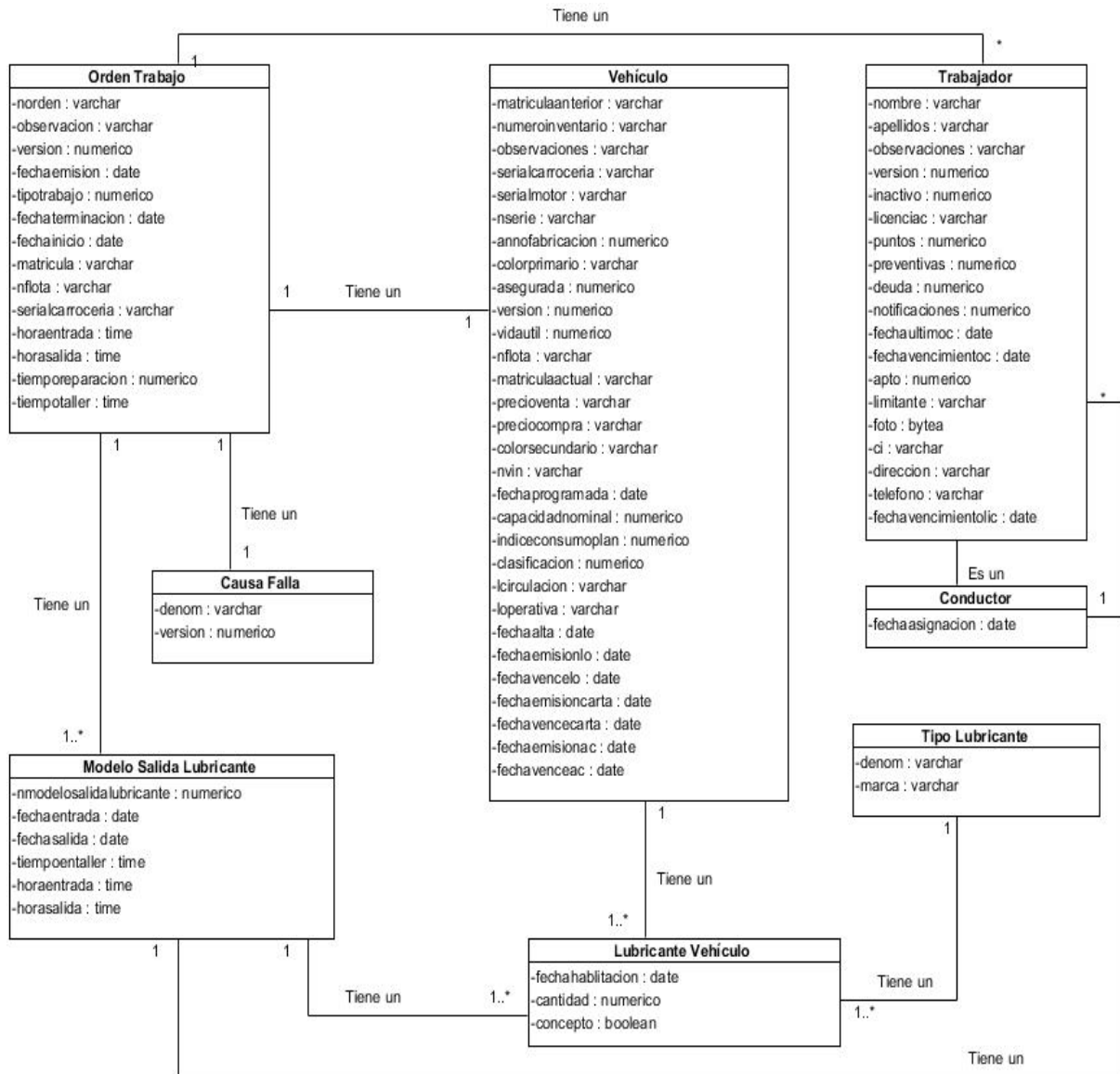


Figura 2. Modelo conceptual del proceso Control de Lubricantes

Para más información sobre la confección del modelo conceptual del proceso de Control de Lubricantes consultar el documento entregable CIG-CFM-N-MTTO-i1301.

2.5. Requisitos de software

Los requisitos de software describen los servicios ofrecidos por el sistema y sus restricciones. Estos reflejan las necesidades de los clientes (43). Los mismos pueden dividirse en dos categorías: requisitos funcionales y requisitos no funcionales.

2.5.1. Técnicas para la captura de requisitos

Para la captura de los requisitos del módulo Control de Lubricantes se utilizaron las técnicas que se relacionan a continuación:

Capítulo 2. Modelado de negocio y Requisitos

Entrevista: la misma se aplicó realizando preguntas relacionadas con el proceso Control de Lubricantes al personal conocedor del mismo en la Dirección de Transporte de la UCI en aras de obtener las necesidades que tenían de informatización. Esto permitió que fueran identificados los requisitos necesarios para el desarrollo de las funcionalidades que permitan llevar a cabo la gestión de este proceso.

Tormenta de ideas: en cada uno de los encuentros realizados se debatieron los requisitos identificados, debate que permitió que fluyeran diversas ideas acerca de los mismos, lo cual permitió obtener una visión más amplia de lo que se quería implementar, favoreciéndose de esta forma el avance de futuras etapas en el desarrollo del módulo Control de Lubricantes.

Análisis de documentación: esta técnica se utilizó para realizar el análisis de distintos sistemas existentes, relacionados con el control de flotas y mantenimiento, específicamente los que realizaran acciones destinadas al control de lubricantes. De estos se observó el tipo de información que manejan y la manera en que realizan dicho control, obteniéndose la posibilidad de seleccionar características y arribar a conclusiones importantes que fueron de gran utilidad para el desarrollo de la solución.

2.5.2. Requisitos funcionales

Los requisitos funcionales son declaraciones de los servicios que el sistema debe proporcionar, cómo este debe reaccionar ante entradas específicas y cómo debe comportarse en determinadas situaciones (43). Los requisitos funcionales identificados en el proceso Control de Lubricantes son los siguientes:

RF 1. Agrupación de requisitos: Gestionar tipo de lubricantes:

- RF 1.1. Adicionar tipo de lubricante
- RF 1.2. Modificar tipo de lubricante
- RF 1.3. Eliminar tipo de lubricante
- RF 1.4. Listar tipo de lubricante
- RF 1.5. Buscar tipo de lubricante

RF 2. Generar modelo de salida de lubricantes

RF 3. Agrupación de requisitos: Gestionar modelo de salida de lubricantes:

- RF 3.1. Modificar modelo de salida de lubricantes
- RF 3.2. Cancelar modelo de salida de lubricantes
- RF 3.3. Listar modelo de salida de lubricantes
- RF 3.4. Buscar modelo de salida de lubricantes
- RF 3.5. Imprimir modelo de salida de lubricantes

Capítulo 2. Modelado de negocio y Requisitos

RF 4. Gestionar lubricantes vehículo:

RF 4.1. Adicionar lubricantes vehículo

RF 4.2. Modificar lubricantes vehículo

RF 4.3. Eliminar lubricantes vehículo

RF 4.4. Listar lubricantes vehículo

RF 5. Autorizar orden de relleno extra de lubricantes

RF 6. Rechazar orden de relleno extra de lubricantes

RF 7. Limpiar búsqueda

RF 8. Salida del sistema control de consumo de lubricantes por tipo de vehículo en un período

RF 9. Salida del sistema comportamiento de consumo por tipo de lubricantes en un período

RF 10. Salida del sistema control de consumo de lubricantes por tipo de vehículo en un año

RF 11. Salida del sistema comportamiento de consumo por tipo de lubricantes en un año

RF 12. Salida del sistema historial de habilitación de lubricantes por vehículo

RF 13. Salida del sistema historial de habilitación de lubricantes por vehículo en un período

RF 14. Salida del sistema vehículos que se les venció el relleno de lubricantes

RF 15. Salida del sistema modelos de salida de lubricantes cancelados

Reutilización de requisitos

Para el desarrollo del módulo Control de Lubricantes se contó con la posibilidad de reutilizar requisitos que ya se encontraban implementados en el Sistema de Control de Flotas y Mantenimiento, estos requisitos son: Generar Solicitud, Generar Orden de Trabajo y la agrupación de requisitos Gestionar Orden de Trabajo.

En consecuencia, aumentó la fiabilidad del producto pues cuenta con funcionalidades que han sido verificadas, validadas y en ocasiones avaladas por usuarios que las han utilizado en otro momento. Además, permitió que se trabajara de forma más rápida, hábil y eficiente, ya que se evita la duplicación de código y se controla mejor el proceso de desarrollo de software.

2.5.3. Descripción de requisitos funcionales

La descripción de los requisitos funcionales del módulo Control de Lubricantes permitió realizar una descripción textual de cada uno de los requisitos identificados, donde se reflejan los flujos por los que transita el requisito, la información que muestra, así como las

Capítulo 2. Modelado de negocio y Requisitos

restricciones que posee. También queda plasmado en este documento un prototipo no funcional de interfaz de usuario del requisito descrito.

La siguiente descripción corresponde al requisito funcional Generar modelo de salida de lubricantes:

Tabla 5. Descripción del requisito funcional Generar modelo de salida de lubricantes

Precondiciones		Se ha registrado una orden de trabajo para un vehículo con causa de falla sistema de lubricación
Flujo de eventos		
Flujo básico		
1	Se selecciona la orden de trabajo.	
2	Se introducen los siguientes datos: Número de modelo Fecha de entrada Hora de entrada Fecha de salida Hora de salida Confeccionado Reportado Despachado Recibido	
3	Se asocian los lubricantes al vehículo.	
4	Concluye el requisito.	
Pos-condiciones		
1	NA	
Flujos alternativos		
Flujo alternativo		
1	NA	
Pos-condiciones		
1	NA	
Validaciones		
1	NA	
Relaciones	Requisitos Incluidos	NA
	Extensiones	NA
Conceptos	Modelo de salida de lubricantes	Visibles en la interfaz: Número de modelo Número de orden Vehículo

Capítulo 2. Modelado de negocio y Requisitos

	Marca
	Fecha de entrada
	Hora de entrada
	Fecha de salida
	Hora de salida
	Tiempo en taller
	Confeccionado
	Reportado
	Despachado
	Recibido
	Utilizados internamente:
	idorden
	idvehículo
	idtipolubricante

Requisitos especiales	NA
Asuntos pendientes	NA

Prototipo no funcional de interfaz de usuario

Generar modelo de salida de lubricantes

No. Modelo:

No. Orden:

Vehículo:

Marca:

Fecha entrada:

Hora entrada:

Fecha salida:

Hora salida:

Tiempo en taller:

Confeccionado por:

Reportado por:

Despachado por:

Recibido por:

Asociar lubricantes

+ M -

Tipo de lubricante	Cantidad	Unidad de medida

Cancelar Aceptar

Figura 3. Prototipo de interfaz de usuario del requisito funcional Generar modelo de salida de lubricantes

La restante información sobre la descripción del requisito anteriormente abordado se encuentra en el documento entregable CIG-CFM-R-MTTO-i2649 y las descripciones correspondientes a los demás requisitos funcionales del módulo Control de Lubricantes se

Capítulo 2. Modelado de negocio y Requisitos

localiza en los documentos entregables: CIG-CFM-R-MTTO-i2647, CIG-CFM-R-MTTO-i2648, CIG-CFM-R-MTTO-i2650, CIG-CFM-R-MTTO-i2651, CIG-CFM-R-MTTO-i2652, CIG-CFM-R-MTTO-i2653, CIG-CFM-R-MTTO-i2946, CIG-CFM-R-MTTO-i2947, CIG-CFM-R-MTTO-i2948, CIG-CFM-R-MTTO-i2949, CIG-CFM-R-MTTO-i2950, CIG-CFM-R-MTTO-i2951, CIG-CFM-R-MTTO-i2952 y CIG-CFM-R-MTTO-i2953.

2.5.4. Trazabilidad de requisitos

Para controlar la trazabilidad de los requisitos del proceso de Control de Lubricantes se tuvieron en cuenta los siguientes elementos:

- Requisitos
- Modelo conceptual
- Diseños de casos de pruebas

Se realizó el diagrama de requisitos mediante el cual se logró definir la relación entre los elementos de trazabilidad mencionados anteriormente (consultar Anexo 2).

Realizando las siguientes matrices de trazabilidad:

- Requisitos – Modelo conceptual
- Requisitos – Diseño de casos de pruebas

A continuación se presentan fragmentos de las mismas:

(16) Requirement		Autorizar orden de relleno extra	Comportamiento del consumo por tipo de lub...	Comportamiento del consumo por tipo de lub...	Consumo de lubricantes por grupo de vehic...	Consumo de lubricantes por grupo de vehic...	Generar modelo de salida de lubricantes	Generar orden de trabajo	Gestionar lubricante vehículo	Gestionar modelo de salida de lubricantes	Gestionar orden de trabajo	Gestionar tipo de lubricante	Historial de habilitación de lubricante por ve...	Historial de habilitación de lubricante por ve...	Modelos de salida de lubricantes cancelados	Rechazar orden de relleno extra	Vehiculos que se les venció el relleno de lu...
(7) Class																	
<input type="checkbox"/>	Conductor						✓	✓		✓	✓						
<input type="checkbox"/>	Lubricante Vehículo		✓	✓	✓	✓	✓			✓			✓	✓	✓		✓
<input type="checkbox"/>	Modelo Salida Lubricante						✓		✓	✓					✓		
<input type="checkbox"/>	Orden Trabajo	✓						✓			✓					✓	
<input type="checkbox"/>	Tipo Lubricante		✓	✓	✓	✓	✓		✓	✓		✓	✓	✓			
<input type="checkbox"/>	Trabajador						✓	✓		✓	✓						
<input type="checkbox"/>	Vehículo		✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓			✓

Figura 4. Matriz de trazabilidad Requisitos – Modelo conceptual

Capítulo 2. Modelado de negocio y Requisitos

(16) Test Case	Autorizar orden de relleno...	Comportamiento del con...	Comportamiento del con...	Consumo de lubricantes...	Consumo de lubricantes...	Generar modelo de sali...	Generar orden de trabajo	Gestionar lubricante ve...	Gestionar modelo de sal...	Gestionar orden de trab...	Gestionar tipo de lubric...	Historial de habilitación ...	Historial de habilitación ...	Modelos de salida de lu...	Rechazar orden de relleno...	Vehículos que se les ve...
Autorizar orden de relleno extra	«V» ...>															
Comportamiento del consumo por tipo de lubricante en un año	«V» ...>															
Comportamiento del consumo por tipo de lubricante en un periodo		«V» ...>														
Consumo de lubricantes por grupo de vehículo en un periodo				«V» ...>												
Consumo de lubricantes por tipo de vehículo en un año				«V» ...>												
Generar modelo de salida de lubricantes						«V» ...>										
Gestionar lubricante vehículo								«V» ...>								
Gestionar modelo de salida de lubricantes									«V» ...>							
Gestionar orden de trabajo										«V» ...>						
Gestionar orden de trabajo							«V» ...>									
Gestionar tipo de lubricante											«V» ...>					

Figura 5. Matriz de trazabilidad Requisitos – Diseños de casos de pruebas

2.5.5. Requisitos no funcionales

Los requisitos no funcionales describen las restricciones de los servicios o funciones ofrecidas por el sistema. Pueden incluir restricciones de tiempo, de las normas de desarrollo y de procesos. Generalmente se aplican al sistema como un todo (43). Dentro de los requisitos no funcionales del Sistema de Control de Flotas y Mantenimiento se encuentran:

RNF 1. Usabilidad

- 1.1. El idioma de todas las interfaces de la aplicación será el español.
- 1.2. 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.
- 1.3. 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.
- 1.4. Los errores cometidos por el usuario les serán notificados y los mensajes incluirán sugerencias de las posibles soluciones.

RNF 2. Seguridad

- 2.1. El sistema manejará la seguridad de acceso y administración de usuarios mediante el otorgamiento de privilegios y roles, así como la asignación de perfiles.

Capítulo 2. Modelado de negocio y Requisitos

- 2.2. Se concederá acceso al sistema a partir de un nombre de usuario y una contraseña.
- 2.3. 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

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

RNF 4. Hardware

- 4.1. El sistema interactuará con impresoras para imprimir los diferentes documentos que genere la aplicación, como respuesta a las funcionalidades del sistema.
- 4.2. El sistema para su instalación en las máquinas clientes requiere:
 - 4.2.1. Procesador: 1.40 GHZ
 - 4.2.2. RAM: 512 MB (recomendado 1024 GB)
 - 4.2.3. Tarjeta de Red: 1
- 4.3. El sistema para su instalación en el servidor de aplicación requiere:
 - 4.3.1. Procesador: 3.00 GHZ
 - 4.3.2. RAM: 1GB (recomendado 4GB)
 - 4.3.3. Disco duro: 160 GB (recomendado 500GB)
 - 4.3.4. UPS: 1
 - 4.3.5. Lector de CD: 1
 - 4.3.6. Tarjeta de Red: 1
- 4.4. El sistema para su instalación en el servidor de base de datos requiere:
 - 4.4.1. Procesador: 3.00 GHZ
 - 4.4.2. RAM: 1GB (recomendado 4GB)
 - 4.4.3. Disco duro: 160 GB (recomendado 500GB)
 - 4.4.4. UPS: 1
 - 4.4.5. Lector de CD: 1
 - 4.4.6. Tarjeta de Red: 1

2.5.6. Priorización de requisitos

La priorización de requisitos de software, hace referencia a una actividad crítica dentro de la Ingeniería de Requisitos, a partir de la cual, se establece un proceso de decisión preferentemente consensuado, tendiente a establecer el orden de implementación de una serie de requisitos de software, tomando en consideración, algún conjunto definido de

Capítulo 2. Modelado de negocio y Requisitos

criterios. Asimismo, su correcta implementación, trae aparejado una serie de beneficios concretos que se materializan cuando resulte indispensable estimar el uso de los recursos disponibles, el tiempo comprometido y los costos incurridos en un proyecto de software (44).

Durante la priorización de los requisitos del módulo Control de Lubricantes se evaluaron todos los requisitos funcionales atendiendo a una serie de criterios definidos y según el resultado de esta evaluación, se le otorga al requisito una prioridad que puede ser Alta, Media o Baja.

Para la determinación de la complejidad de los requisitos se analizan individualmente los criterios siguientes:

- La complejidad por interfaces
- Los diferentes comportamientos
- Las formas de inicialización
- Las consultas a fuentes de datos
- Las restricciones de validación
- El grado de reutilización
- La lógica de negocio

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. Esta arrojó como resultado que de un total de 26 requisitos, 9 presentan complejidad Baja, 7 complejidad Media y 10 complejidad Alta. Ver tabla 6.

Tabla 6. Priorización de los requisitos del proceso de Control de Lubricantes

Requisitos	Complejidad
Listar tipo de lubricante	Baja
Adicionar tipo de lubricante	Baja
Modificar tipo de lubricante	Baja
Eliminar tipo de lubricante	Baja
Buscar tipo de lubricante	Baja
Generar modelo de salida de lubricantes	Alta
Modificar modelo de salida de lubricantes	Alta
Cancelar modelo de salida de lubricantes	Baja
Listar modelo de salida de lubricantes	Baja
Buscar modelo de salida de lubricantes	Media

Capítulo 2. Modelado de negocio y Requisitos

Imprimir modelo de salida de lubricantes	Media
Adicionar lubricante vehículo	Media
Modificar lubricante vehículo	Media
Eliminar lubricante vehículo	Baja
Listar lubricante vehículo	Media
Autorizar relleno extra de lubricantes	Media
Rechazar relleno extra de lubricantes	Media
Limpiar búsqueda	Baja
Control de consumo de lubricantes por tipo de vehículo en un período	Alta
Control de consumo de lubricantes por tipo de vehículo en un año	Alta
Comportamiento de consumo por tipo de lubricantes en un período	Alta
Comportamiento de consumo por tipo de lubricantes en un año	Alta
Historial de habilitación de lubricantes por vehículo	Alta
Historial de habilitación de lubricantes por vehículo en un período	Alta
Vehículos que se les venció el relleno de lubricantes	Alta
Modelos de salida de lubricantes cancelados	Alta

Para más información acerca de la priorización de requisitos funcionales del módulo Control de Lubricantes, consultar el documento entregable CIG-CFM-R-MTTO-i6103.

2.5.7. Validación de requisitos

Para la validación de los requisitos del módulo Control de Lubricantes se utilizaron las siguientes técnicas:

Revisión técnica formal: la revisión de las descripciones de los requisitos del proceso Control de Lubricantes fue realizada con el personal conocedor del mismo en la Dirección de Transporte de la UCI. La utilización de esta técnica permitió validar que no existieran errores en el contenido, malas interpretaciones, información incompleta, inconsistencias y que los requisitos no fueran contradictorios, imposibles o inalcanzables. Se aprobaron los que estaban descritos de forma correcta, clara y consistente.

Casos de prueba: se diseñaron casos de prueba para cada uno de los requisitos, con el objetivo de verificar el correcto funcionamiento de los mismos, identificando los posibles escenarios por los que transitan los requisitos, así como los juegos de datos que pueden

Capítulo 2. Modelado de negocio y Requisitos

recibir los campos contenidos en estos. Se validaron y aprobaron los que estaban bien enunciados y descritos.

Construcción de prototipos: partiendo de las especificaciones de requisitos se conformaron prototipos de interfaz de usuario. Con esto se validó que los requisitos estaban en concordancia con las necesidades plasmadas por los proveedores de requisitos, lo cual ayudó al cliente a tener una idea clara de la estructura de la interfaz de usuario. Además permitió mejorar la comunicación con el mismo, basada en una visión inicial de las funcionalidades a través de las cuales se gestionaría el proceso Control de Lubricantes.

Conclusiones parciales del capítulo

Con la realización del presente capítulo se arribó a las siguientes conclusiones:

- Mediante el modelado de negocio del proceso Control de Lubricantes se obtuvieron artefactos que sirvieron de base para lograr un mejor entendimiento del proceso, para su posterior informatización. Además creó las condiciones para llevar a cabo la captura de requisitos funcionales.
- Los requisitos identificados fueron descritos, validados y priorizados, garantizando que no existieran errores que posteriormente pudieran afectar el Sistema de Control de Flotas y Mantenimiento, sentando las bases para realizar el análisis y diseño de la solución.

Capítulo 3. Diseño, Implementación y Pruebas

CAPÍTULO 3. DISEÑO, IMPLEMENTACIÓN Y PRUEBAS

Introducción

A través del diseño de software se pueden obtener los modelos del sistema a desarrollar. En el presente capítulo se hace referencia a un grupo de aspectos relevantes pertenecientes a esta disciplina que permitieron al mismo arribar en óptimas condiciones a la etapa de implementación, entre estos se encuentran: los mecanismos y patrones de diseño, los diagramas de clases del diseño y las métricas para su validación. Además se hace alusión a los elementos fundamentales abordados durante la implementación de las funcionalidades de apoyo al proceso de Control de Lubricantes, los mismos son: el diagrama de componentes, los estándares de codificación y el modelo de despliegue. Por último el capítulo realiza un recorrido por los tipos de pruebas seleccionadas, con sus respectivas técnicas para garantizar la calidad de las funcionalidades implementadas.

3.1. Diseño

A partir del diseño de software se pueden obtener los modelos del sistema que se desea desarrollar. Estos modelos pueden evaluarse en relación con su calidad y de esta forma lograr mejorarlos antes de pasar a las etapas de implementación y pruebas. A continuación se relacionan los elementos fundamentales que se tuvieron en cuenta durante esta etapa.

3.1.1. Mecanismos de diseño

Los mecanismos de diseño permiten simplificar los diagramas de clases. Para el desarrollo del módulo Control de Lubricantes se utilizaron mecanismos previamente definidos para el Sistema de Control de Flotas y Mantenimiento, entre estos se encuentran:

Mecanismo de diseño para las clases controladoras

Todas las clases controladoras definidas en el diseño heredan de la clase ZendExt_Controller_Secure, ya que en ella se incluyen numerosas funcionalidades comunes en todas las controladoras (45).



Figura 6. Mecanismo de diseño para las clases controladoras

Capítulo 3. Diseño, Implementación y Pruebas

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 (45).

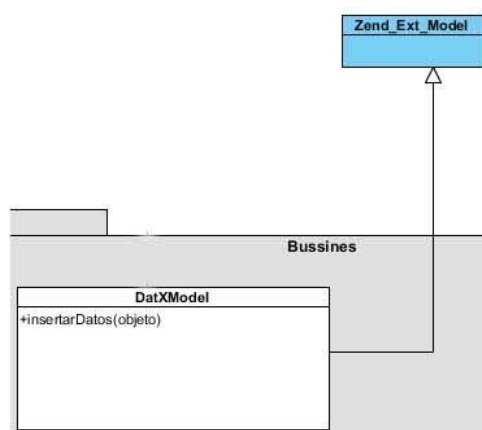


Figura 7. Mecanismo de diseño para las clases modelos

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; estas 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 (45).

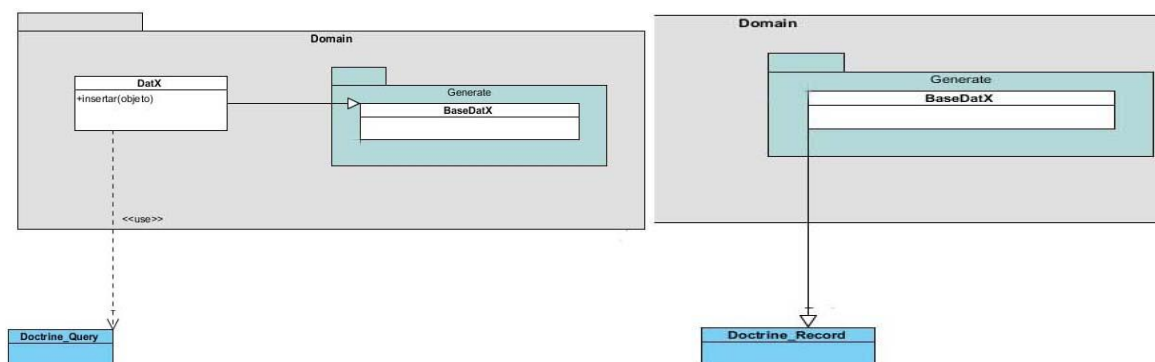


Figura 8. Mecanismo de diseño para las clases del dominio

Mecanismo de diseño para el 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 Business (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

Capítulo 3. Diseño, Implementación y Pruebas

3.1.2. Patrones de diseño

El marco de trabajo Sauxe por su capacidad de combinar buenas prácticas, permite que los desarrolladores no tengan que preocuparse por implementar varios de los patrones de diseño más utilizados en la actualidad, ya que este por sí mismo los maneja. Durante el desarrollo de la solución se utilizaron los siguientes patrones:

Patrones GRASP

Controlador: el mismo se puede observar en la creación de una clase controladora para las funcionalidades que se realizan en el módulo Control de Lubricantes o Clasificadores. Dentro de las clases creadas se encuentran: GestmodelosalidalubricantesController, GenerarmodelosalidalubricantesController y RecuperacionesController.

Experto: fue usado en la mayoría de las clases ya que a cada una le fue asignada la responsabilidad en dependencia de la información que esta posee.

Alta Cohesión: se le asignaron responsabilidades a las clases de forma tal que la cohesión se mantuviera alta, o sea, cada clase se encargará de realizar solamente las funciones que estén en correspondencia con la responsabilidad que esta posee.

Bajo Acoplamiento: se observa en la mayor parte del diseño pues las clases fueron creadas teniendo en cuenta que las mismas presentaran la menor cantidad de relaciones de dependencia posible.

Creador: este patrón se puede apreciar principalmente en las clases controladoras las cuales necesitan crear objetos de las clases del modelo para poder acceder a las funcionalidades que estas poseen.

Patrones GoF

Singleton: este patrón fue utilizado cuando alguna clase del módulo hace uso de la clase IOC dada la necesidad de obtener información de otro componente. Todas las clases que realicen esta operación llevan a cabo sus peticiones a través de una instancia única. La instancia se puede obtener realizando la llamada *\$this->plIntegrator->* seguido del nombre del componente y de la funcionalidad de la cual se necesite adquirir información.

Fachada: la utilización de este patrón está reflejada en la creación y empleo de la clase IOC, 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 que se necesitaran utilizar, facilitando su interacción.

Capítulo 3. Diseño, Implementación y Pruebas

Cadena de responsabilidad: este patrón se puede apreciar en la comunicación existente entre las clases controladoras y el modelo. Una vez realizada una petición por parte de la vista, el controlador la delega hacia el modelo y a partir del tipo de petición, se le da respuesta a la misma en la clase que tenga asignada esta responsabilidad.

3.1.3. Modelo Vista Controlador

La implementación de este patrón en el módulo Control de Lubricantes se realizó de la siguiente manera: La Vista abarca la capa de presentación, en la misma se maneja todo el flujo web, brindando al usuario la posibilidad de interactuar con el Controlador mediante los mensajes de eventos y visualizar las respuestas a sus peticiones. El Controlador 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 siguientes paquetes: “business”, el cual engloba las clases donde se realiza la lógica del negocio, y “domain”, donde están comprendidas las clases del dominio, estas se encargan de realizar todas las operaciones sobre la base de datos.

3.2. Diseño del sistema

Durante el ciclo de vida de un software, el diseño del sistema representa un componente fundamental del mismo. Se centra en diseñar una solución que permita convertir los requisitos del cliente en un sistema de información real. Los aspectos tenidos en cuenta para el desarrollo del mismo son los siguientes:

3.2.1. Diagrama de clases del diseño con estereotipos web

A continuación se muestra el diagrama correspondiente al requisito “Generar modelo de salida de lubricantes”. Donde el controlador, en este caso representado por la clase `GenerarmodelosalidalubricantesController`, respondiendo a una petición realizada desde la vista principal mediante la relación `<<link>>`, construye la página cliente `generarmodelosalidalubricantes.phtml` a través de la relación `<<build>>`. La misma contiene los script donde se encuentra implementado el código de ejecución de la página y estos a su vez contienen dos formulario que toman los datos ingresados por el usuario en la página y los envían al controlador utilizando 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`, las que utilizan las funcionalidades brindadas por el marco de trabajo `Doctrine` para acceder a las tablas de

Capítulo 3. Diseño, Implementación y Pruebas

la base de datos. En caso de que la información solicitada por la página cliente se encuentre en otro componente, el controlador a través de la relación <<instance>> accede a la clase IOC, la cual accede a las clases de servicio del componente y estas a partir de la información obtenida, realizan las operaciones necesarias y le envían los resultados al controlador.

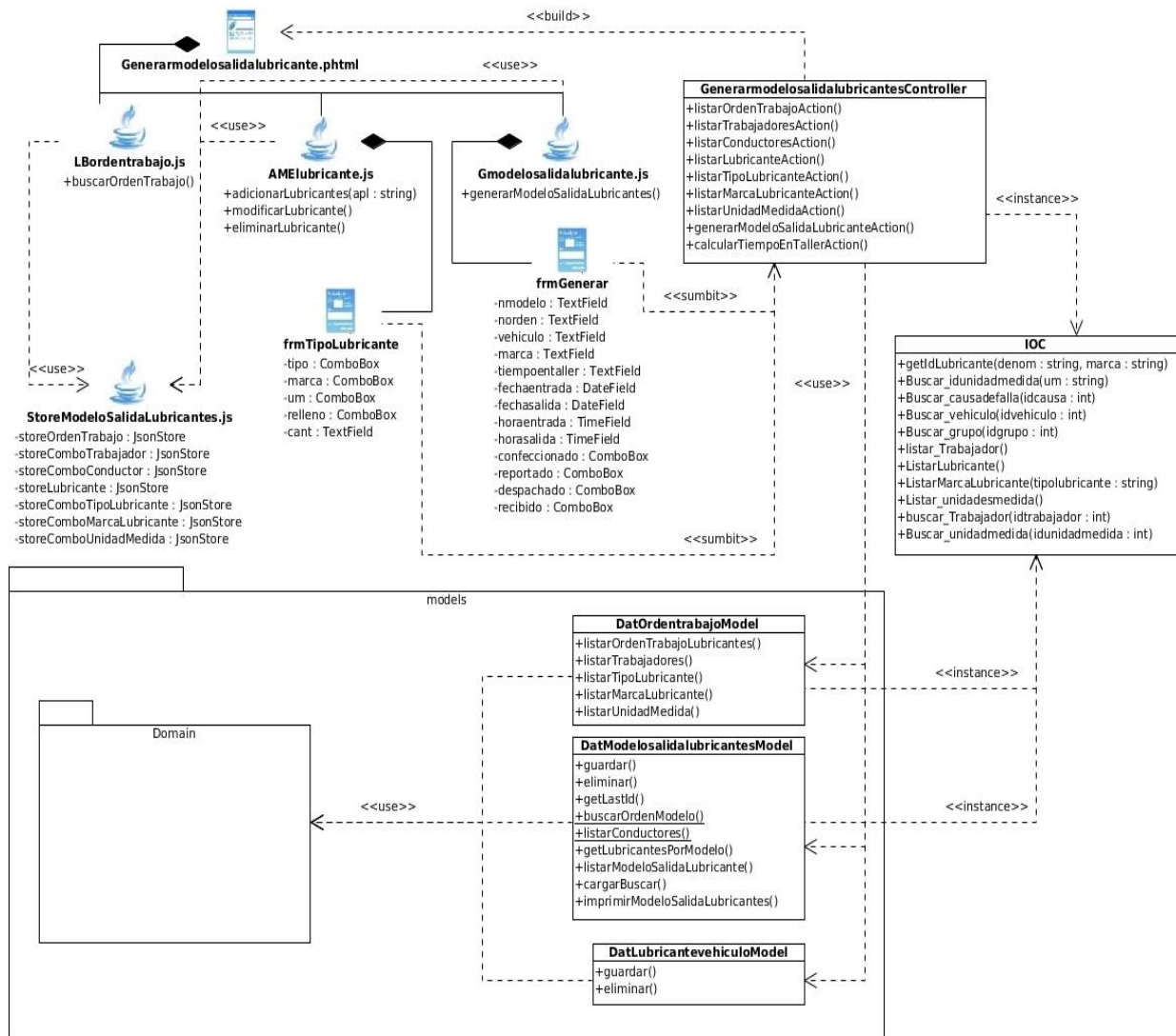


Figura 11. Diagrama de clases del diseño del requisito Generar modelo de salida de lubricantes

3.2.2. Diagrama de secuencia

El diagrama de secuencias permite crear una representación de los objetos que se encuentran en el escenario, así como la secuencia de mensajes que intercambian los mismos para llevar a cabo la funcionalidad descrita por una transacción del sistema.

A continuación se muestra el diagrama de secuencia, perteneciente al requisito funcional Generar modelo de salida de lubricantes.

Capítulo 3. Diseño, Implementación y Pruebas

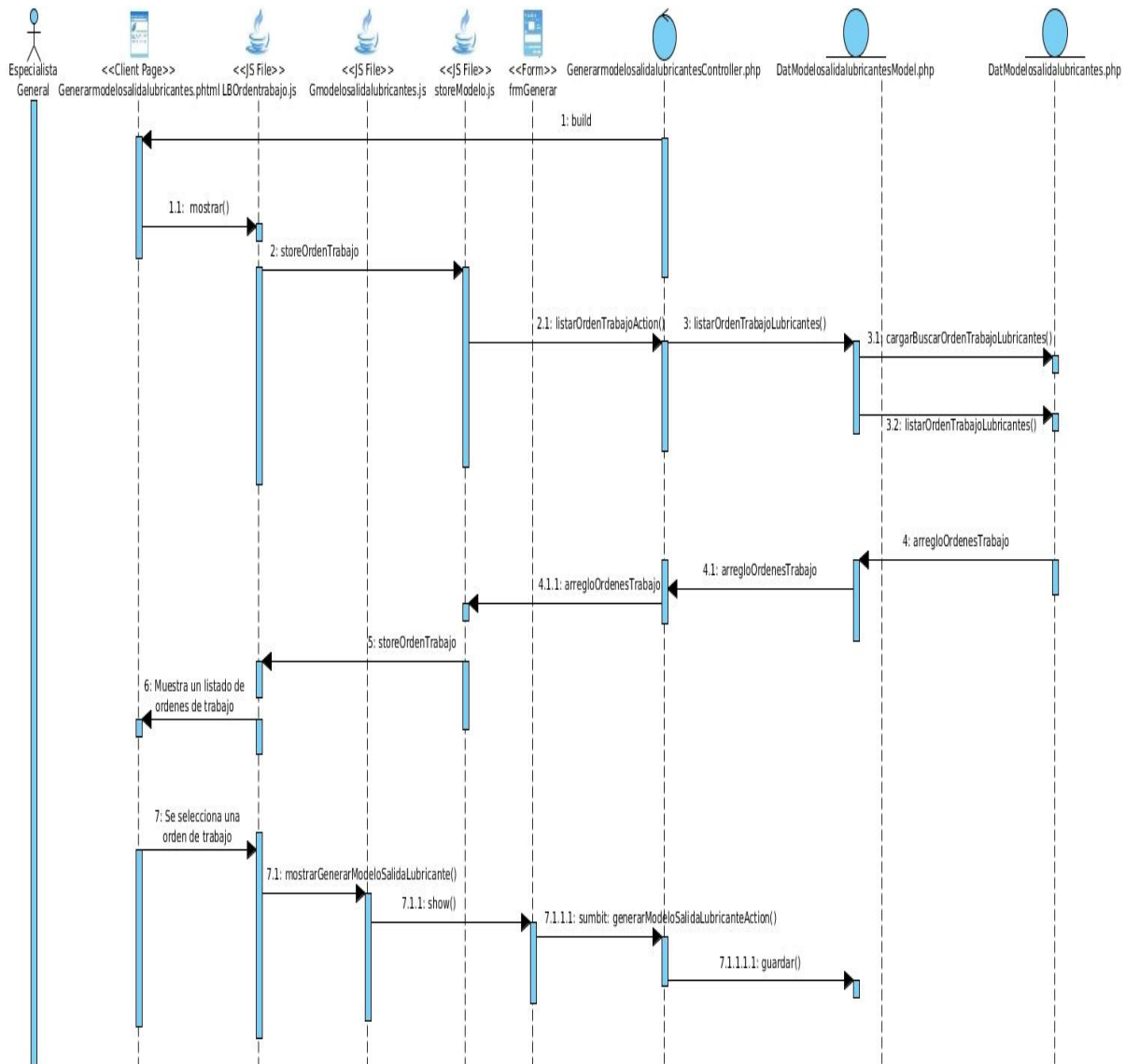


Figura 12. Diagrama de secuencia del requisito Generar modelo de salida de lubricantes

3.2.3. Métricas para validar el diseño

Para realizar la validación del diseño se escogieron las métricas que a continuación se relacionan:

Métrica de Tamaño Operacional de Clases (TOC)

Esta métrica se encuentra entre las planteadas por Lorenz y Kidd y pertenece al grupo de las métricas de tamaño de clase. La misma se centra en la cantidad de operaciones para cada clase individual y los valores promedio para el sistema como un todo. 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:

Capítulo 3. Diseño, Implementación y Pruebas

- **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. Los umbrales para esta métrica se basan 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 7. Criterios para evaluar los atributos según el umbral de TOC

Atributos	Categoría	Criterio
Responsabilidad	Baja	\leq Promedio
	Media	Entre Promedio y $2 \times$ Promedio
	Alta	$>$ Promedio
Complejidad de implementación	Baja	\leq Promedio
	Media	Entre Promedio y $2 \times$ Promedio
	Alta	$>$ Promedio
Reutilización	Baja	\leq Promedio
	Media	Entre Promedio y $2 \times$ Promedio
	Alta	$>$ Promedio

A partir de la evaluación de cada una de las clases según los atributos anteriormente mencionados se obtuvo el siguiente resultado:

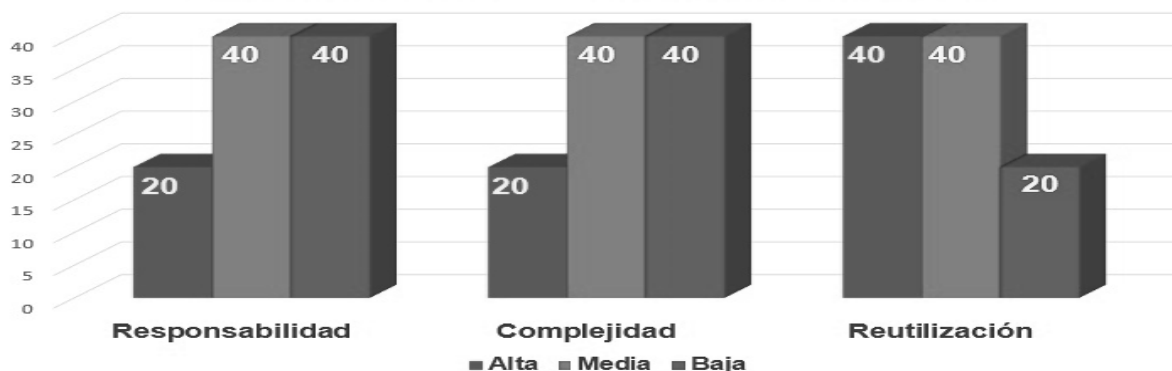


Figura 13. Gráfica con los resultados de la evaluación del diseño según TOC

Capítulo 3. Diseño, Implementación y Pruebas

El análisis de los resultados obtenidos al realizar la evaluación del diseño mediante la métrica TOC demuestra ser aceptable, pues el 40% de las clases poseen evaluación baja en los atributos de calidad Responsabilidad y Complejidad de Implementación. Mientras que la variable Reutilización se comporta de igual manera para una evaluación alta.

Métrica de Relaciones entre Clases (RC)

Esta métrica se basa en la cantidad de relaciones de uso que presenta una clase determinada con las demás clases. Utiliza los atributos Acoplamiento, Complejidad de Mantenimiento, Reutilización y Cantidad de Pruebas para medir la calidad del diseño de la clase.

Los umbrales para evaluar el diseño según esta métrica se basan en el promedio de relaciones por clases y el caso del atributo Acoplamiento, utiliza la propia cantidad de relaciones, estos valores fueron los aplicados en el diseño del sistema y los mismos son reflejados en la siguiente tabla.

Tabla 8. Criterios para evaluar los atributos según el umbral de RC

Atributos	Categoría	Criterio
Acoplamiento	Bajo	1
	Medio	2
	Alto	> 2
Complejidad de mantenimiento	Baja	\leq Promedio
	Media	Entre Promedio y $2 \cdot$ Promedio
	Alta	> Promedio
Reutilización	Baja	\leq Promedio
	Media	Entre Promedio y $2 \cdot$ Promedio
	Alta	> Promedio
Cantidad de pruebas	Baja	\leq Promedio
	Media	Entre Promedio y $2 \cdot$ Promedio
	Alta	> Promedio

Haciendo un análisis del resultado que se muestra a continuación, se puede concluir que el diseño del sistema es aceptable, teniendo en cuenta que el 60% de las clases presentan una evaluación baja en cuanto a la variable Acoplamiento. Además, los atributos de Complejidad de mantenimiento, Cantidad de pruebas y Reutilización se comportan satisfactoriamente.

Capítulo 3. Diseño, Implementación y Pruebas

3.3. Implementación

La implementación de las funcionalidades de apoyo al proceso de Control de Lubricantes permitió obtener como resultado un producto que cumple con las necesidades de la Dirección de Transporte de la UCI. Dentro de los principales aspectos analizados durante esta disciplina se encuentran los que a continuación se abordan.

3.3.1. Diagrama de componentes

A través del diagrama de componentes se describen los elementos físicos del sistema y sus relaciones. El mismo se realizó con el objetivo de crear una vista general del sistema a partir de las dependencias e integraciones de los componentes y módulos. A continuación se muestra el diagrama de componentes del Sistema de Control de Flotas y Mantenimiento:

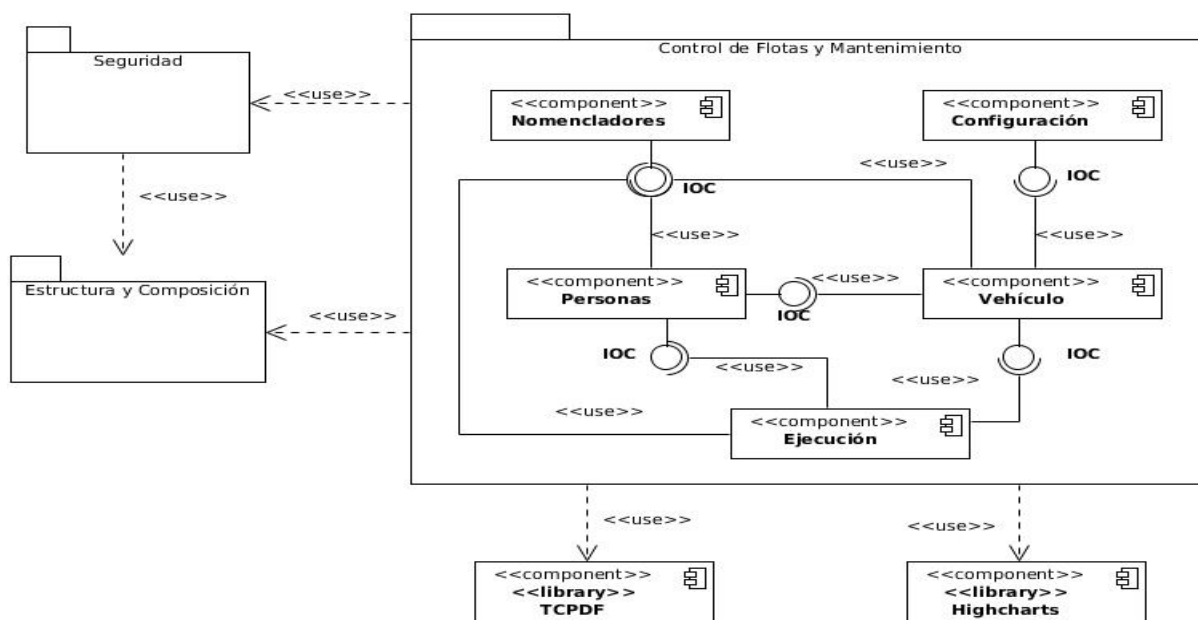


Figura 16. Diagrama de componentes del Sistema de Control de Flotas y Mantenimiento

En este diagrama se puede apreciar la integración entre los componentes, la cual se realiza mediante los servicios que brindan unos y son consumidos por otros y viceversa, siendo la clase IOC utilizada como interfaz para garantizar dicha integración. El módulo Control de Lubricantes, que tiene distribuidas las clases que permiten satisfacer sus requisitos en los componentes Nomencladores y Ejecución, además de hacer uso de los servicios brindados por el componente Nomencladores, utiliza algunos de los que brindan los componentes: Persona y Vehículo. Entre estos servicios aparecen: listarTrabajador y buscarTrabajador del componente Persona, listarVehículo y buscarVehículo, del componente Vehículo y listarLubricante, listarMarcaLubricante, getIdLubricante, listarUnidadMedida, buscarCausaFalla y BuscarUnidadMedida, del componente Nomencladores.

Capítulo 3. Diseño, Implementación y Pruebas

3.3.2. Estándares de codificación

A continuación se presentan los estándares de codificación que se utilizaron para definir el modo en que se nombrarán las clases, funciones y variables de la solución:

Nomenclatura según el tipo de clases

Los nombres de las clases comienzan con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing (46).

- Las clases controladoras después del nombre llevan la palabra: “Controller” (46). Por ejemplo: GestmodelosalidalubricantesController.
- Las clases que se encuentran dentro de la carpeta “business” después del nombre llevan la palabra: “Model” (46). Por ejemplo: DatModelosalidalubricantesModel.
- Las clases que se encuentran dentro de la carpeta “domain” el nombre que reciben es el de la tabla en la Base de Datos (46). Por ejemplo: DatModelosalidalubricantes.
- Las clases que se encuentran dentro de la carpeta “generated” el nombre comienza con la palabra: “Base” y seguido el nombre de la tabla en la Base de Datos (46). Por ejemplo: BaseDatModelosalidalubricantes.

Nomenclatura de las funciones

El nombre a emplear para las funciones se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing y este debe sugerir el propósito de la misma (46). Por ejemplo: listarModeloSalidaLubricantes.

En caso de ser una acción de la clase controladora se asigna el nombre y seguido la palabra: “Action” (46). Por ejemplo: modificarModeloSalidaLubricantesAction.

Nomenclatura de las variables

El nombre a emplear para las variables se escribe con la primera palabra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamelCasing, y comenzando con un prefijo según el tipo de datos (46). Por ejemplo: modeloSalida.

3.3.3. Interfaces de la solución

La solución brindada permite, de manera sencilla, tener un control exhaustivo de la información referente al relleno de lubricantes de los vehículos de la entidad. Facilitando la obtención de reportes con datos relevantes para apoyar la toma de decisiones. A continuación se muestran varias interfaces del módulo desarrollado donde se puede apreciar los elementos antes abordados:

Capítulo 3. Diseño, Implementación y Pruebas

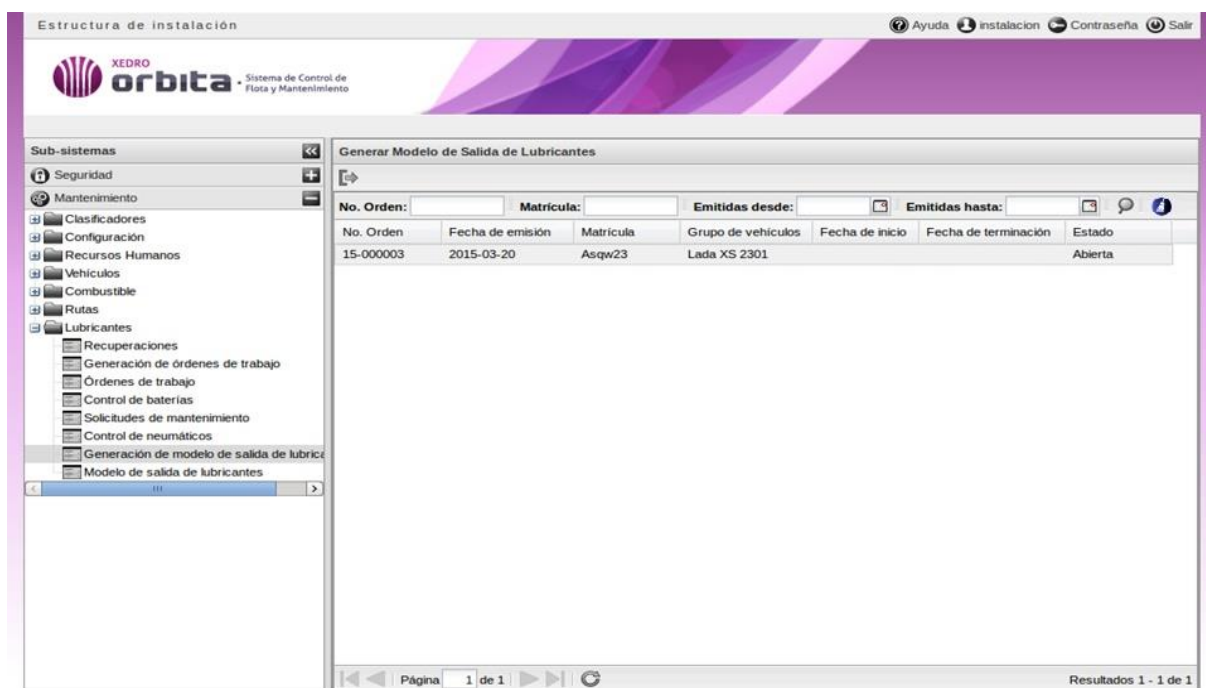


Figura 17. Interfaz de usuario de la funcionalidad Generar modelo de salida de lubricantes

Una vez emitida una orden de trabajo que indique el relleno de lubricantes a un vehículo, esta se lista en la interfaz de la funcionalidad Generar modelo de salida de lubricantes, representada en la figura anterior (Figura 17). Desde dicha funcionalidad el usuario puede generar el documento donde se le da salida a los lubricantes suministrados al vehículo.

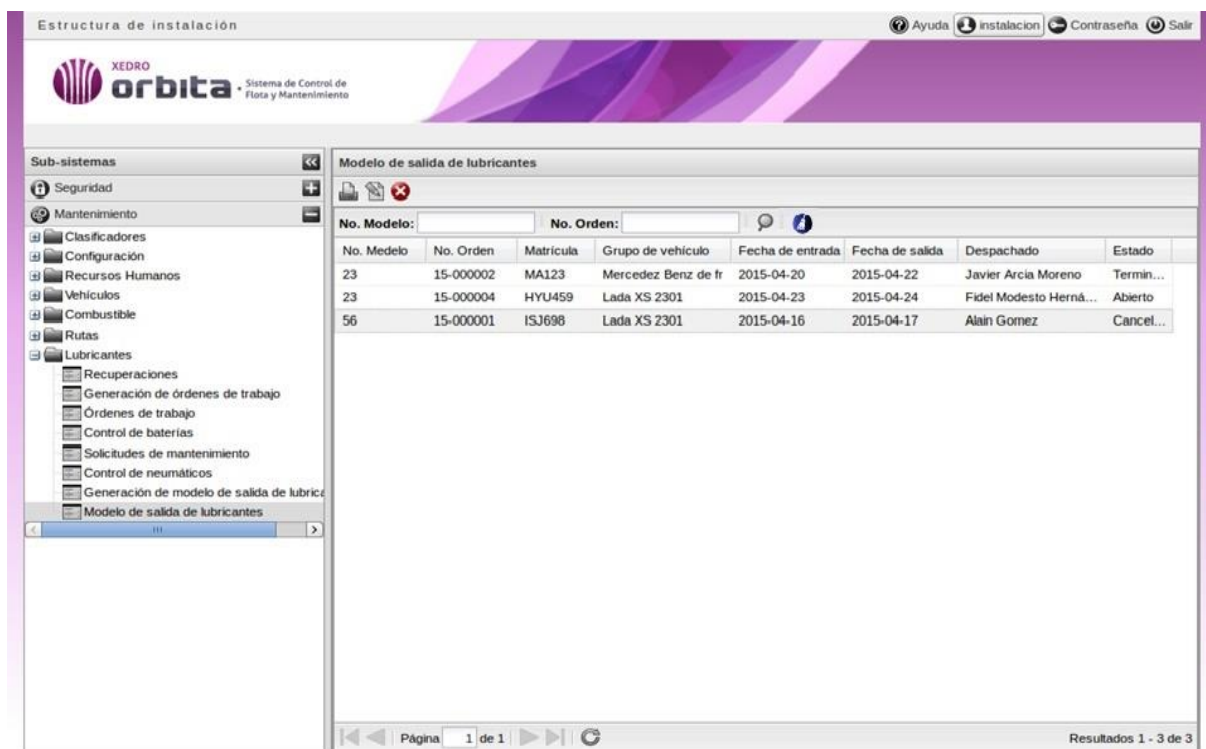


Figura 18. Interfaz de usuario de la funcionalidad Modelo de salida de lubricantes

Capítulo 3. Diseño, Implementación y Pruebas

La figura anterior (Figura 18) muestra la funcionalidad Modelo de salida de lubricantes, la cual permite al usuario gestionar dicho modelo. Esta brinda la opción de modificar en caso de que se necesite agregar o cambiar algún dato del mismo y permite realizar acciones como: exportar el modelo a formato pdf o cancelarlo si su generación no procede.

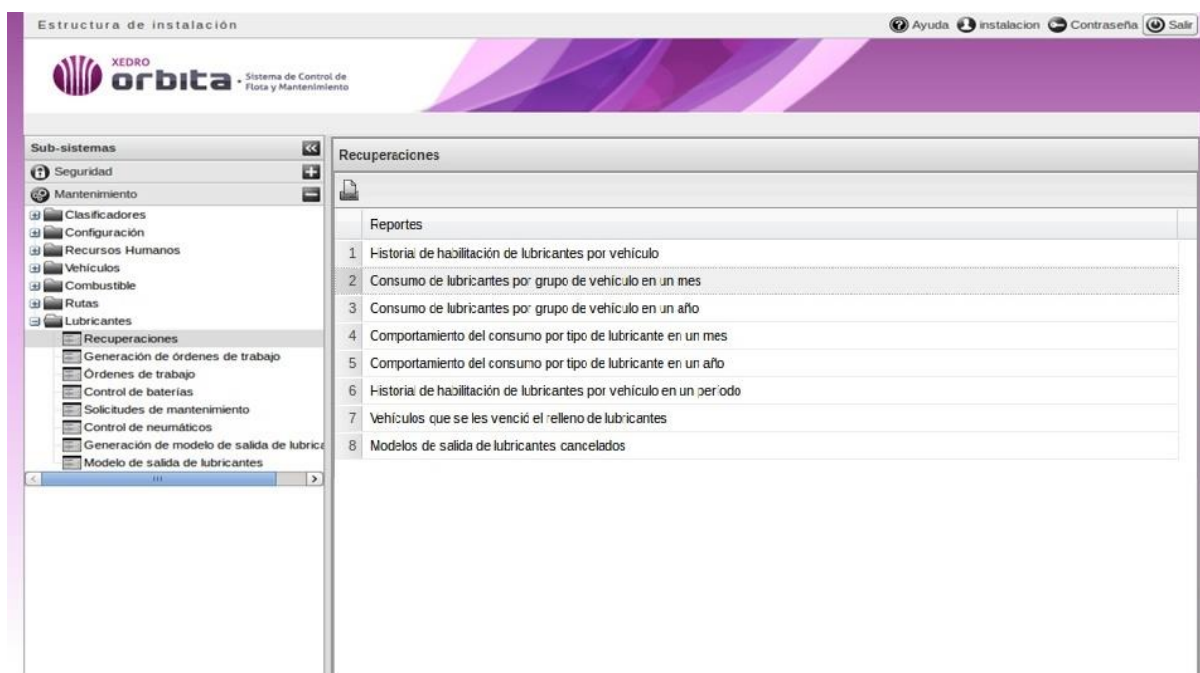


Figura 19. Interfaz de usuario de la funcionalidad Recuperaciones

La figura anterior (Figura 19) muestra la interfaz de la funcionalidad Recuperaciones. Desde dicha funcionalidad el usuario puede generar reportes en formato pdf con información necesaria para apoyar la toma de decisiones en la entidad. Entre los reportes que brinda la misma se encuentran: Consumo de lubricantes por grupo de vehículo en un período, Comportamiento del consumo por tipo de lubricante en un período, Historial de habilitación de lubricantes a un vehículo y Vehículos que se les venció el relleno de lubricantes.

3.3.4. Modelo de despliegue

Los diagramas de despliegue permiten visualizar la disposición física de los distintos recursos computacionales que componen un sistema. Están compuestos por nodos, dispositivos y conectores. El modelo de despliegue tiene como objetivo capturar la configuración de los elementos de procesamiento y las conexiones entre los mismos en el sistema. A continuación se presenta el diagrama de despliegue perteneciente al Sistema de Control de Flotas y Mantenimiento.

Capítulo 3. Diseño, Implementación y Pruebas

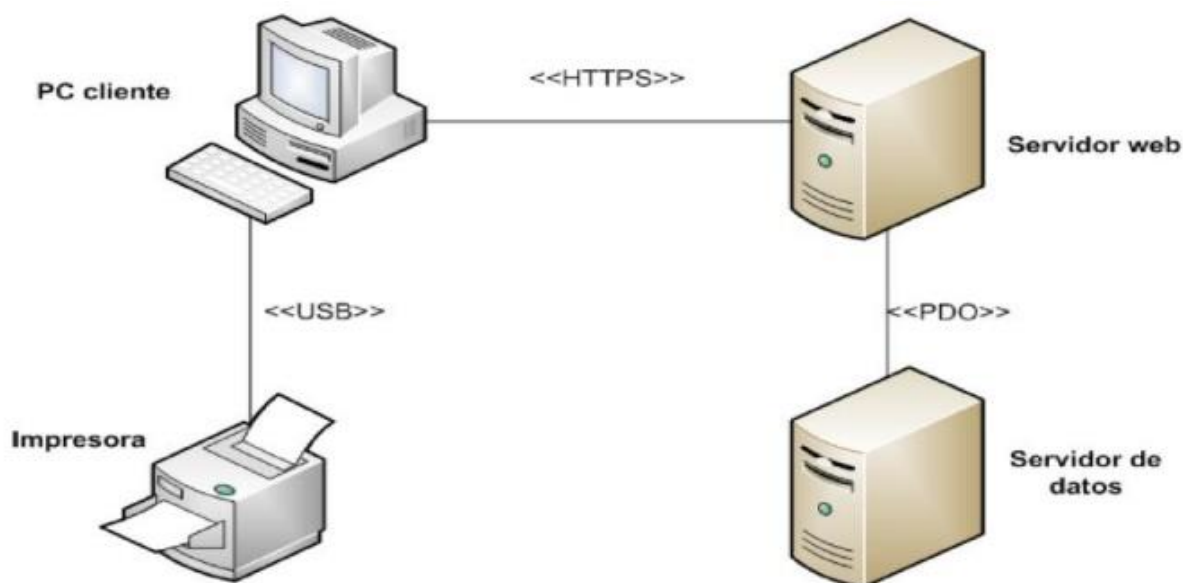


Figura 20. Diagrama de despliegue del Sistema de Control de Flotas y Mantenimiento

PC cliente: accede a la aplicación a través de un computador, donde es ejecutada mediante el navegador Mozilla Firefox versión 4.0 o superior, sobre Windows o Linux.

Servidor web: el servidor de aplicaciones empleado donde radica la lógica de negocio de la aplicación es Apache 2.2 utilizando la biblioteca de clases PHP 5.3.

Servidor de datos: se encuentra el sistema gestor de base de datos PostgreSQL 9.1 donde se encuentra la base de datos utilizada por el sistema.

Impresora: se utiliza para imprimir los reportes que permite generar el Sistema de Control de Flotas y Mantenimiento.

3.4. Pruebas

Una vez implementadas las funcionalidades correspondientes a cada requisito, se realizó una serie de pruebas para garantizar que las mismas cumplieran correctamente con los requisitos aceptados por el cliente. A continuación se hace alusión a los tipos de pruebas, con sus respectivas técnicas, utilizadas para llevar a cabo dicha acción.

3.4.1. Pruebas de caja blanca

Las pruebas de caja blanca se realizaron aplicando la **Técnica del Camino Básico**. Para el desarrollo de la misma es necesario realizar el cálculo de la complejidad ciclomática del algoritmo que será analizado. Partiendo de la enumeración las sentencias de código del mismo, se procede a elaborar el grafo de flujo correspondiente a esta funcionalidad.

Capítulo 3. Diseño, Implementación y Pruebas

La siguiente figura muestra las sentencias enumeradas del código de la funcionalidad `listarModeloSalidaLubricantesAction()`.

```
function listarModeloSalidaLubricanteAction() {  
    $start = $this->_request->getPost("start"); #1  
    $limit = $this->_request->getPost("limit"); #1  
    $nmodelo = $this->_request->getPost('nmodelosalidalubricante'); #1  
    $norden = $this->_request->getPost('norden'); #1  
    $modelo = new DatModelosalidalubricantesModel(); #2  
    if ($nmodelo != null || $norden != null) { #3  
        $datos = $modelo->cargarBuscar($nmodelo, $norden, $limit, $start); #4  
    } else { #5  
        $datos = $modelo->listarModeloSalidaLubricante($limit, $start); #5  
    }  
    echo json_encode($datos); #6  
    return;  
}
```

Figura 21. Sentencias enumeradas de la funcionalidad `listarModeloSalidaLubricantesAction()`

A continuación se muestra el grafo de flujo correspondiente a la funcionalidad `listarModeloSalidaLubricantesAction()`.

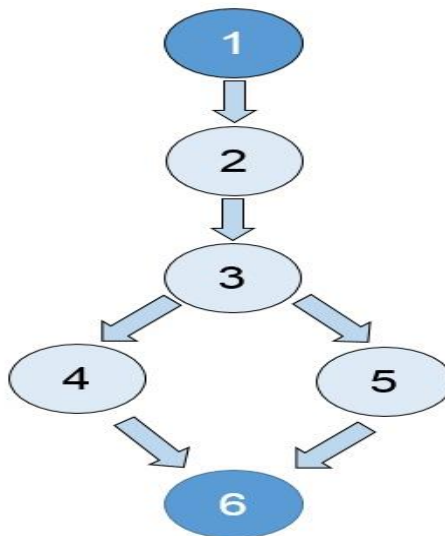


Figura 22. Grafo correspondiente al flujo de la función `listarModeloSalidaLubricantesAction()`

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. Cuando se usa en el contexto del método de prueba del camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y brinda un límite superior para el número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez (12).

Capítulo 3. Diseño, Implementación y Pruebas

La complejidad ciclomática será calculada de tres maneras diferentes. Al calcular la complejidad del método `listarModeloSalidaLubricantesAction ()` fueron utilizadas las tres vías posibles para así verificar que el cálculo se había realizado correctamente. Las fórmulas para realizar dicho cálculo son las siguientes:

1. $V(G) = (A - N) + 2$, $V(G) = (6 - 6) + 2$, $V(G) = 2$.

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 $V(G)$ representa 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 transitar por 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: 1-2-3-4-6.

Camino básico # 2: 1-2-3-5-6.

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: 1-2-3-4-6

Descripción: los datos de entrada cumplirán con los siguientes requisitos:

Al menos uno de los criterios de búsqueda (`$modelo` y `$norden`) no puede ser nulo.

Entrada: `$POST = [[start] => 0, [limit] => 20, [nmodelosalidalubricantes] =>23, [norden] => 15-0000002]`.

Resultados esperados: se espera que se muestre un listado con los datos de los modelos de salida de lubricantes que cumplen con los criterios de búsqueda.

Resultados obtenidos: satisfactorio.

Capítulo 3. Diseño, Implementación y Pruebas

Caso de prueba para el camino básico # 2

Camino: 1-2-3-5-6

Descripción: los datos de entrada cumplirán con los siguientes requisitos:

Los criterios de búsqueda (\$modelo y \$norden) deben ser nulo.

Entrada: \$POST = [[start] => 0, [limit] => 20, [nmodelosalidalubricantes] =>, [norden] =>].

Resultados esperados: se espera que se muestre un listado con los datos de todos modelos de salida de lubricantes registrados en el sistema.

Resultados obtenidos: satisfactorio.

3.4.2. Pruebas de caja negra

Para llevar a cabo las pruebas de caja negra se utilizó la técnica **Partición Equivalente**. Definiendo para cada uno de los requisitos identificados un caso de prueba. A continuación se muestra el flujo central del caso de pruebas del requisito Generar modelo de salida de lubricantes.

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/Lubricantes/Generación de modelo de salida de lubricantes**.
- Se debe haber registrado en el sistema al menos una orden de trabajo con la causa de falla sistema de lubricación.

Tabla 9. Caso de pruebas del requisito Generar modelo de salida de lubricantes

Descripción general	Escenarios de pruebas	Flujo del escenario
Debe permitir generar el modelo de salida de lubricante.	EP 1.1: Generar modelo de salida de lubricante introduciendo datos válidos.	<ul style="list-style-type: none">– Se debe seleccionar una orden de trabajo.– Se debe seleccionar el botón Generar.– Se introducen los datos del modelo de salida de lubricante correctamente.– Se presiona el botón Aceptar.– Se muestra un mensaje de

Capítulo 3. Diseño, Implementación y Pruebas

	información.
	<ul style="list-style-type: none">– Se presiona el botón Aceptar.
EP 1.2: Generar modelo de salida de lubricante dejando campos vacíos.	<ul style="list-style-type: none">– Se debe seleccionar una orden de trabajo.– Se debe seleccionar el botón Generar.– Se introducen los datos dejando algún campo en blanco.– Se presiona el botón Aceptar.– Se muestra un mensaje informando del error.– Se presiona el botón Aceptar
EP 1.3: Generar modelo de salida de lubricante introduciendo datos no válidos.	<ul style="list-style-type: none">– Se debe seleccionar una orden de trabajo.– Se debe seleccionar el botón Generar.– Se introducen los datos del modelo de salida de lubricante incorrectamente.– Se presiona el botón Aceptar.– Se muestra un mensaje de error.– Se presiona el botón Aceptar.
EP 1.4: Generar modelo de salida de lubricante introduciendo horas no válidas.	<ul style="list-style-type: none">– Se debe seleccionar una orden de trabajo.– Se debe seleccionar el botón Generar.– Se introducen las horas de entrada y salida del modelo de salida de lubricantes incorrectamente.– Se muestra un mensaje de error.

Capítulo 3. Diseño, Implementación y Pruebas

	Se presiona el botón Aceptar .
EP 1.5: Generar modelo de salida de lubricante con un número de modelo ya registrado.	<ul style="list-style-type: none">– Se debe seleccionar una orden de trabajo.– Se debe seleccionar el botón Generar.– Se introduce un número de modelo que ya se encuentre registrado en el sistema.– Se presiona el botón Aceptar.– Se muestra un mensaje de error.– Se presiona el botón Aceptar.
EP 1.6: Cancelar.	<ul style="list-style-type: none">– Se introducen o no los datos del tipo de lubricante.– Se presiona el botón Cancelar.

Para más información referente al caso de prueba que se mostró anteriormente consultar el documento entregable CIG-CFM-I-MTTO-i5165. Los restantes casos de prueba se encuentran en los documentos entregables: CIG-CFM-I-MTTO-i5160, CIG-CFM-I-MTTO-i5161, CIG-CFM-I-MTTO-i5162, CIG-CFM-I-MTTO-i5163, CIG-CFM-I-MTTO-i5164, CIG-CFM-I-MTTO-i5166, CIG-CFM-I-MTTO-i5167, CIG-CFM-I-MTTO-i5168, CIG-CFM-I-MTTO-i5169, CIG-CFM-I-MTTO-i5170, CIG-CFM-I-MTTO-i5171, CIG-CFM-I-MTTO-i5172, CIG-CFM-I-MTTO-i5173, CIG-CFM-I-MTTO-i5174, CIG-CFM-I-MTTO-i5175, CIG-CFM-I-MTTO-i5176, CIG-CFM-I-MTTO-i5177, CIG-CFM-I-MTTO-i5178, CIG-CFM-I-MTTO-i5179, CIG-CFM-I-MTTO-i5180, CIG-CFM-I-MTTO-i5181, CIG-CFM-I-MTTO-i5182, CIG-CFM-I-MTTO-i5183, CIG-CFM-I-MTTO-i5184 y CIG-CFM-I-MTTO-i5185.

La solución fue sometida a dos iteraciones de dichas pruebas, donde se evaluaron elementos como ortografía, errores de interfaz, funcionalidades incorrectas o ausentes, errores en estructuras de datos o en el acceso a la base de datos. Obteniendo en la primera iteración un total de 7 no conformidades, divididas en 3 de documentación y otras 4 de aplicación. Al realizar la segunda iteración de pruebas no se encontraron no conformidades.

Con la aplicación de las pruebas de caja negra se demostró el correcto funcionamiento de las funcionalidades implementadas en el módulo Control de Lubricantes pues se obtuvieron los resultados esperados como respuesta al conjunto de condiciones de entrada definidas en cada diseño de caso de prueba.

Capítulo 3. Diseño, Implementación y Pruebas

Conclusiones parciales del capítulo

Tras el desarrollo del presente capítulo se arribó a las siguientes conclusiones:

- A través del análisis y diseño se generaron los artefactos necesarios para el desarrollo de la solución.
- Se validó el diseño utilizando las métricas TOC y RC, obteniendo resultados satisfactorios por cada uno de los indicadores.
- Mediante la implementación de las funcionalidades del módulo se cumplió con los requisitos identificados, satisfaciendo de esta forma las necesidades del cliente.
- Se comprobó el correcto funcionamiento de las funcionalidades implementadas utilizando las pruebas de software, obteniendo los resultados esperados en cada una de estas.

Conclusiones Generales

CONCLUSIONES GENERALES

Mediante la investigación, los resultados obtenidos y las validaciones realizadas se cumplió el objetivo general de dicha investigación, pues se logró desarrollar el módulo Control de Lubricantes para el Sistema de Control de Flotas y Mantenimiento, lo cual permitió arribar a las siguientes conclusiones:

- El estudio del proceso de Control de Lubricantes en diferentes sistemas de gestión de flotas vehiculares, permitió conocer a fondo como estos realizaban dicho proceso, identificando características que constituían condiciones deseadas en la solución, así como dificultades que esta no debía presentar.
- A través del análisis y diseño se obtuvieron una serie de artefactos indispensables para el correcto desarrollo de la solución, permitiendo tener un mejor dominio de sus funcionalidades.
- La implementación de las funcionalidades del módulo permitió cumplir con los requisitos identificados, utilizando para esto herramientas, tecnologías y lenguajes distribuidos bajo licencias de software libre en correspondencias con las políticas vigentes en la universidad y el país.
- El módulo desarrollado satisface las necesidades plasmadas por el cliente, evidenciándose en los resultados obtenidos en las validaciones realizadas en las etapas de modelado de negocio, requisitos, análisis y diseño, implementación y pruebas de la solución.
- Se contribuyó a mejorar el proceso de Control de Lubricantes en la Dirección de Transporte de la UCI.

RECOMENDACIONES

Se recomienda:

- Utilizar el presente trabajo como material de estudio y referencia para el desarrollo de futuras investigaciones relacionadas al control de lubricantes.
- Exhortar a la Dirección de Transporte de la Universidad de las Ciencias Informáticas a usar el módulo desarrollado.

REFERENCIAS BIBLIOGRÁFICAS

1. **Works Gestión de Mantenimiento, S. L.** WGM Division Lubricantes. [En línea] <http://www.wgmlubricantes.com/default.html>.
2. **KRUNIĆ, Vlado, AŠONJA, Aleksandar y KRUNIĆ, Momčilo** . SOFTWARE SUPPORT FOR THE MANAGEMENT OF LUBRICATION. *Bulletin of Engineering* . 2014, Vol. VII.
3. **Definición ABC**. [En línea] <http://www.definicionabc.com/motor/lubricante.php>.
4. **Hernández Centeno, Jérémy** . Monografías. *Lubricantes*. [En línea] Universidad Simón Bolívar, 2007. <http://www.monografias.com/trabajos48/lubricantes/lubricantes.shtml#ixzz3bQ1UHSwY>.
5. **Transporte, Especialistas de la Dirección de**. Universidad de las Ciencias Informáticas, 2014.
6. **System Logística**. System Logística Empresaria. [En línea] <http://www.system.com.ar/>.
7. **La Pampeana**. *Sistema de Gestión de Flotas de Vehículos*. [En línea] 8 de Julio de 2004 . http://www.lapampeana.com.ar/internacional/nov_ampliada.asp?idnovedad=2.
8. **Datys**. *gCARS: Sistema de Gestión Vehicular V. 1.0. Especificaciones Técnicas*. 2010.
9. **TranSoft**. TranSoft. Software y Diseño. [En línea] <http://www.transoft.transnet.cu/?controlador=Producto&accion=index>.
10. **Universidad de las Ciencias Informáticas**. *Metodología de desarrollo para la Actividad productiva de la UCI*. La Habana : s.n., 2014.
11. **Sommerville, Ian**. *Ingeniería de Software 7ma Edición*.
12. **Pressman, Roger S**. *Ingeniería de Software: un enfoque práctico 5ta Edición*.
13. **Pressman, Roger S**. *Ingeniería de Requisito. Un Efoque Practico*. 2005.
14. **Amaolo, Aranda y Noemí, Gabriela**. *Marco para la elitización de requisitos software en procesos de desarrollo global*. 2009.
15. **Peláez, Juan Carlos**. *Arquitectura basada en Componentes*. [En línea] Abril de 2009. <http://geeks.ms/blogs/jkpelaiez/archive/2009/04/18/arquitectura-basada-en-componentes.aspx>.
16. **Buschmann, Frank, y otros**. *A System of Patterns*. 1996.
17. **Bascón Pantoja, Ernesto**. *El patrón de diseño modelo -vista - controlador (MVC) y su implementación el Java Swing*. 2004.
18. **Larman, Craig**. *UML y Patrones, Introducción al Análisis y Diseño Orientado a Objetos*. 2012.
19. **Tabares Botero, Ricardo**. *Grasp Patterns and Anti-Patterns: an Object*. 2012.
20. **Gamma, Erich y Otros**. *Design Patterns. Elements of Reusable*. 1995.
21. **Bizagi. Bizagi Suite**. *BPMN 2.0*. [En línea] 2014. <https://www.bizagi.com/docs/BPMNbyExampleSPA.pdf>.

Referencias Bibliográficas

22. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software*. s.l. : Rational Software Corporation, 2000.
23. **Eguiluz, Javier.** *Introducción a AJAX*. 2008.
24. **Definicion.de.** *Lenguaje de programación*. [En línea] 2008. <http://definicion.de/lenguaje-de-programacion/>.
25. **Van Der Henst, Christian y Heredia Santos, Herminio.** *Maestros del Web*. [En línea] 23 de Mayo de 2001. <http://www.maestrosdelweb.com/phpintro/>.
26. **Mozilla Developer Network y colaboradores individuales.** *Mozilla*. [En línea] 23 de Septiembre de 2014. <https://developer.mozilla.org/es/docs/Web/JavaScript>.
27. **Cañete Pupo, Yanisleydi.** *Libro de Ayuda del Marco de Trabajo Sauxe*. La Habana : s.n., 2010.
28. **Zend Framework.** *Zend Framework*. [En línea] 2012. <http://framework.zend.com/manual/1.12/en/manual.html>.
29. **Blades, Steve, Ramsay, Colin y Frederick, Shea.** *Learning Ext JS*. 2008.
30. **Doctrine Team.** *Doctrine*. [En línea] 2012. <http://www.doctrine-project.org/>.
31. **Collins-Sussman, Ben, Fitzpatrick, Brian W. y Pilato, C. Michael.** *Control de versiones con Subversion*. 2004.
32. **Asuni, Nicola.** *TCPDF. . PHP class for generating PDF documents*. [En línea] 2013.
33. **Visual Paradigm.** *Visual Paradigm*. [En línea] 2012. <http://www.visual-paradigm.com/>.
34. **Ibrugor.** *Apache HTTP Server: ¿Qué es, cómo funciona y para qué sirve?* . [En línea] 2014. <http://www.ibrugor.com/blog/apache-http-server-que-es-como-funciona-y-para-que-sirve/>.
35. **PostgreSQL.** *About PostgreSQL*. [En línea] 2014. <http://www.postgresql.org/about/>.
36. **pgAdmin.** *pgAdmin III*. [En línea] 2015. <http://www.pgadmin.org/docs/1.8/index.html>.
37. **Genbeta: dev desarrollo y software.** *NetBeans*. [En línea] enero de 2014. <http://www.genbetadev.com/herramientas/netbeans-1>.
38. **Masadelante.** *¿Qué es un navegador, explorador o buscador?* . [En línea] <http://www.masadelante.com/faqs/que-es-un-navegador>.
39. **Mozilla Developers.** *Mozilla*. [En línea] 2013. <http://www.mozilla.org>.
40. **IngSoftwareII-CUFM.** *Desarrollo de Software*. [En línea] noviembre de 2012. <https://cufmingsoftware.wordpress.com/estandares-de-diseno>.
41. **Rodríguez Rodríguez, Carlos Rafael.** *Herramienta para aplicar métricas al Diagrama de Clases del Diseño Orientado a Objetos*. 2012.
42. **Pruebas de software.** *Gestión de Calidad y Pruebas de Software*. [En línea] 2014. <http://www.pruebasdesoftware.com/laspruebasdesoftware.html>.
43. **Saraiva de Almeida, Edson.** *Requisitos de software. Engenharia de Software* . s.l. : Universidade Sao Judas Tadeus, 2014.

Referencias Bibliográficas

- 44. Martín Azzolini, Carlos.** *Un Enfoque de Priorización de Requerimientos, a partir de la Segmentación de las Preferencias de los Stakeholders.* La PLata : Universidad Nacional de La Plata, 2011.
- 45. Alás Verdecia, Pedro Manuel .** *Modelo de diseño.* La Habana : Control de Flotas y Mantenimiento, 2013.
- 46. Centro de Informatización de Entidades.** *Estándares de codificación para proyectos con el marco de trabajo Sauxe.* La Habana : s.n., 2013.

ANEXOS

Anexo 1. Descripción de procesos de negocio del proceso Control de Lubricantes

Objetivo	Controlar los lubricantes de un vehículo.
Evento(s) que lo genera(n)	Solicitud de lubricantes
Pre condiciones	N/A
Marco legal	Procedimiento para la entrega de los lubricantes. Código: PE-020-97. Versión: 1.0
Clientes internos	Especialista general Chofer Subdirector técnico Engrasador
Clientes externos	N/A
Entradas	N/A
Flujo de eventos	
Flujo básico Control de lubricantes	
1.	Solicitar lubricantes. El Chofer hace la solicitud de lubricantes para el vehículo.
2.	Recibir solicitud de lubricantes. El Especialista general recibe la solicitud de parte del Chofer.
3.	Realizar orden de trabajo. El Especialista general realiza la orden de trabajo a partir de la solicitud recibida.
4.	Entregar orden de trabajo. El Especialista general procede a entregar la orden de trabajo al Chofer solicitante.
5.	Recibir orden de trabajo. El Chofer recibe la orden de trabajo creada anteriormente.
6.	Entregar orden de trabajo en el área de lubricantes. El Chofer le entrega la orden de trabajo al Engrasador.
7.	Recibir orden de trabajo. El Engrasador recibe la orden de trabajo de manos del Chofer.
8.	Realizar medición de lubricantes. El Engrasador mide los lubricantes del vehículo para saber la cantidad de lubricante a rellenar.
9.	Realizar el modelo de salida de lubricantes. El Engrasador procede a crear el modelo de salida de lubricantes.
10.	Cerrar la orden de trabajo. Terminada la lubricación del vehículo el Engrasador procede a cerrar la orden de trabajo.
11.	Entregar orden de trabajo. El Engrasador le entrega al Chofer la orden de trabajo para que este la firme como constancia de que el proceso se realizó de manera correcta.

12. Recibir orden de trabajo. El Chofer recibe la orden de trabajo para verificar los detalles del trabajo realizado.
13. Firmar orden de trabajo. El Chofer firma la orden de trabajo como constancia de su satisfacción.
14. Volver al flujo básico 1. Se vuelve al flujo básico 1 para que se controlen los lubricantes de la semana al siguiente vehículo.

Pos-condiciones

1. Se ha realizado la orden de trabajo.
2. Se ha realizado el modelo de salida de lubricantes.

Salidas

1. Orden de trabajo (doc)
2. Modelo de salida de lubricantes (doc)

Flujos paralelos

1. N/A

Pos-condiciones

1. N/A

Salidas

1. N/A

Flujos alternos

2. a El vehículo ya ha sido lubricado en la semana.

1. Solicitar aprobación. El Especialista general solicita autorización al Subdirector técnico del taller.
2. Recibir solicitud de aprobación. El Subdirector técnico recibe la solicitud de aprobación realizada por el Especialista general.
3. Aprobar la orden de trabajo. El Subdirector técnico procede a aprobar la realización del proceso.
4. Volver al flujo básico 2. Se vuelve al flujo básico 2 para que el Especialista general proceda a crear la orden de trabajo.

Pos-condiciones

1. Se ha autorizado la orden de trabajo.

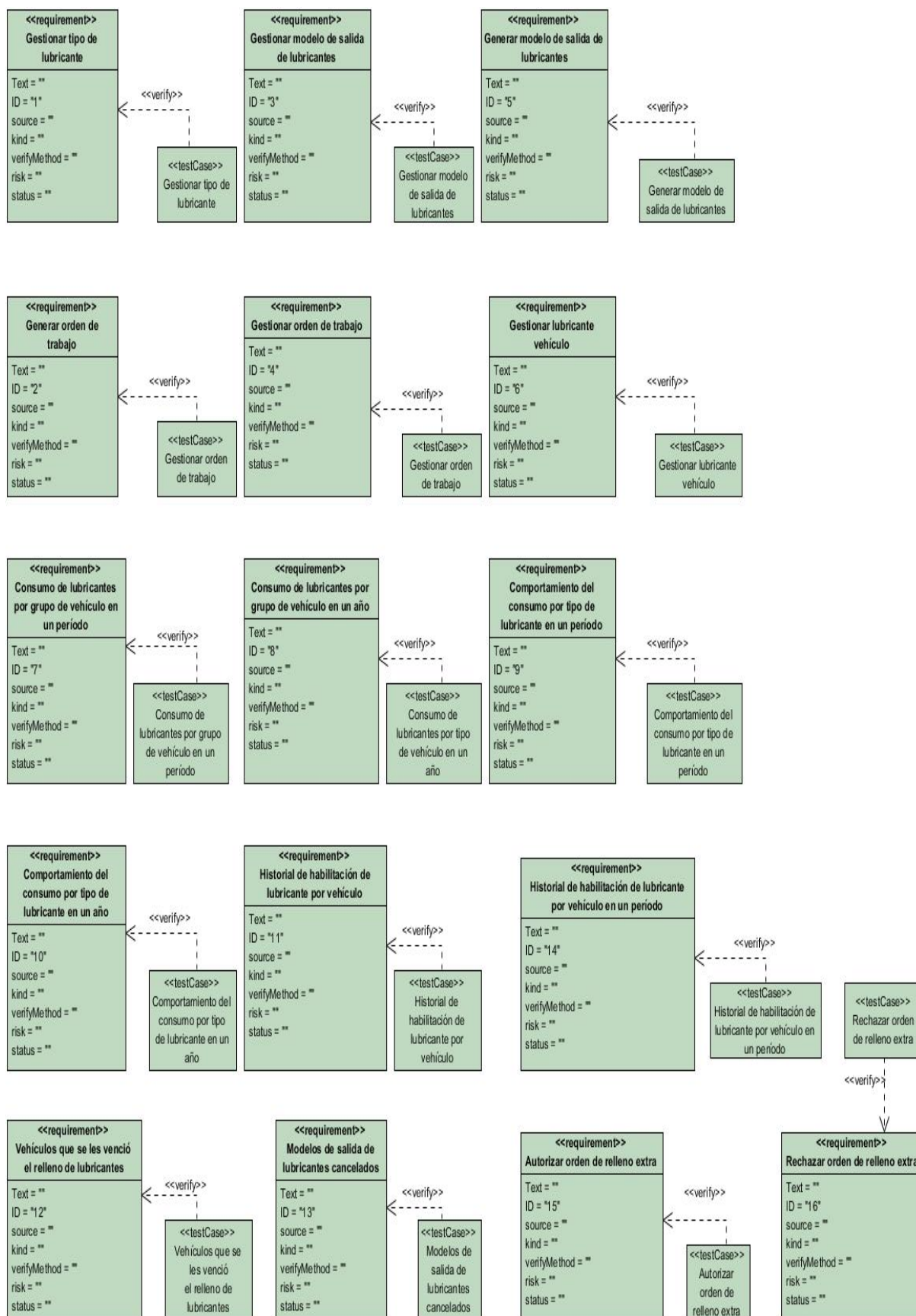
Salidas

1. N/A

Asuntos pendientes

No existe ningún asunto pendiente en este proceso.

Anexo 2. Diagrama de requisitos del proceso Control de Lubricantes



Anexo 3. Resultado obtenido en la Peña Tecnológica 2015



RECONOCIMIENTO

A: ANÁLISIS DE SISTEMAS DE MANTENIMIENTO VEHICULAR Y TECNOLOGÍAS PARA EL DESARROLLO DEL MÓDULO CONTROL DE LUBRICANTES EN EL SISTEMA ÓRBITA.

Por obtener la categoría de *DESTACADO* en el VIII Taller Científico.

Entregado en la Universidad de la Ciencias Informáticas, La Habana, a los 27 días del mes de marzo del año 2015.

TC. Dr. Josué Cardoso Castro

Presidente del Comité Organizador del Evento
Presidente del Consejo Científico DTS-MININT



Anexo 4. Resultado obtenido en la Jornada Científica Estudiantil 2015 a nivel de facultad

III Jornada estudiantil
II Jornada estudiantil
I Jornada científica
Jornada estudiantil
Jornada científica estudiantil
Jornada

UCI Universidad de las Ciencias Informáticas
UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FEDERACIÓN ESTUDIANTIL UNIVERSITARIA

otorga el presente

RECONOCIMIENTO

a: *Luis Ángel Rojas Peña*

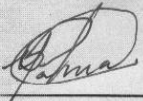
por obtener: *Destacado*


con el trabajo: *Desarrollo del módulo control de subvenciones para el Sistema de Control de Flota y mantenimiento de la Dirección de transporte de la Universidad de las Ciencias Informáticas.*


En la XII edición de la JORNADA CIENTÍFICA ESTUDIANTIL en la FACULTAD 3.

"La tecnología es el reflejo del fanatismo del hombre por sobrevivir"

DADO A LOS 6 DÍAS DEL MES MAYO DEL 2015
"AÑO 57 DE LA REVOLUCIÓN"


YOANDRI GARCÍA PALMA
PRESIDENTE DE LA FEU


UCI Universidad de las Ciencias Informáticas
FACULTAD 3


MSc REINA VICTORIA ESTRADA NELSON
DECANA

Anexo 5. Acta de aceptación del módulo Control de Lubricantes por parte del cliente



Acta de aceptación

Centro de Informatización de Entidades (CEIGE)

20 de Mayo de 2015

En cumplimiento del desarrollo del Sistema de Control de Flota y Mantenimiento para la Dirección de Transporte de la Universidad de las Ciencias Informáticas, se hace entrega del producto que se relaciona a continuación.

— Módulo Control de Lubricantes para el Sistema de Control de Flota y Mantenimiento de la Dirección de Transporte de la Universidad de las Ciencias Informáticas.

<p>Entrega: Módulo Control de Lubricantes para el Sistema de Control de Flota y Mantenimiento de la Dirección de Transporte de la Universidad de las Ciencias Informáticas.</p>	<p>Recibe: Dirección de Transporte de la Universidad de las Ciencias Informáticas.</p>	
<p>Nombre y Apellidos: Luis Ángel Rojas Peña</p>	<p>Nombre y Apellidos: Roberto Hernández González</p>	<p>Cargo: Especialista General</p>
<p>Nombre y apellidos de los tutores: Ing. Miguel Ángel Sánchez Palmero Ing. Leidy Ramos González Ing. Yanerkys Cabrera Martínez</p>	<p>Certifican: Frank Osorio Reinoso Suárez</p>	
<p><i>Declaro que la solución entregada cumple con el marco tecnológico definido y al margen de la Declaración de Autoría es cedida a la Dirección de Transporte de la Universidad de las Ciencias Informáticas.</i></p> <p>Firma: </p>	<p><i>Declaro que la solución entregada cumple con el marco tecnológico definido y ha sido transferida a la Dirección de Transporte de la Universidad de las Ciencias Informáticas.</i></p> <p>Firma: </p>	
<p>Comentarios:</p>		

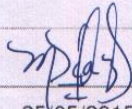
Anexo 6. Acta de liberación del módulo Control de Lubricantes por parte del CEIGE

UCI | CIG-AF-CL09001 : Acta de liberación

Control del documento

Título: Módulo control de lubricantes para el Sistema de Control de Flotas y Mantenimiento.

Versión: 1.0

	Nombre	Cargo
Elaborado por	Ing. Tania Pérez Ramírez	RGA
Revisado por	Ing. Yisel Niño Benitez	Asesor de Calidad
Aprobado por	MSc. Maybel Díaz Capote	Firma 
Cargo	Subdirector I+D+i	Fecha 25/05/2015

Reglas de confidencialidad

Clasificación: Uso Interno
 Forma de distribución: PDF Digital

Este documento contiene información propietaria del **CENTRO DE INFORMATIZACIÓN DE ENTIDADES**, y es emitido confidencialmente para un propósito específico.

El que recibe el documento asume la custodia y control, comprometiéndose a no reproducir, divulgar, difundir o de cualquier manera hacer de conocimientos público su contenido, excepto para cumplir el propósito para el cual se ha generado.

Las reglas son aplicables a las 6 páginas de este documento.

Control de cambios

Versión	Lugar*	Tipo**	Fecha	Autor	Descripción
Versión 1.0	Todo el documento	A	25/05/2015	Ing. Tania Pérez Ramírez	Creación

*Sección del documento, Tabla, Figura.
 ** **A** Alta; **B** Baja; **M** Modificación

CENTRO DE INFORMATIZACIÓN DE ENTIDADES
 Universidad de las Ciencias Informáticas
 Carretera a San Antonio Km 2 ½. Torrens.
 Boyeros. Ciudad de La Habana. Cuba
 Teléfono: + 53 (7) 837 3680
 E-mail: software.gestion@uci.cu

|2

UCI | CIG-AF-CL:09001 : Acta de liberación

2 Datos del producto

Emitida a favor de: Módulo control de lubricantes para el Sistema de Control de Flotas y Mantenimiento.

Fecha de emisión del acta: 25/05/2015

Responsable: Luis A. Rojas Peña

Cargo: Estudiante

2.1 Clasificado como:

→x Aplicación

2.2 Detalle de los elementos probados y su estado final:

Artefacto	Estado Final
Aplicación	0 No Conformidades

2.3 Cantidad de iteraciones:

Para la revisión se emplearon un total de 2 iteraciones de trabajo para lograr el resultado de 0 (cero) No Conformidad.

Artefacto	Versión	Estado final	Cantidad Iteraciones	Tipos de pruebas realizadas	Fecha de liberación
Aplicación	Versión 1.0	No Conformidades	2 iteraciones total	Pruebas Funcionales	25/05/2015

**CENTRO DE INFORMATIZACIÓN
DE ENTIDADES**
Universidad de las Ciencias Informáticas
 Carretera a San Antonio Km 2 ½. Torrens.
 Boyeros. Ciudad de La Habana. Cuba
 Teléfono: + 53 (7) 837 3680
 E-mail: software.gestion@uci.cu

3 Elementos revisados o probados y herramientas utilizadas

Elemento	Herramienta
Aplicación	Diseño de Caso de Pruebas

3.1 Cantidad total de horas empleadas y rango de fechas:

Se emplearon un total de 10 horas efectivas de trabajo del 20/05/2015 y del 25/05/2015.

3.2 Estructura del equipo de prueba empleado y turnos de trabajo:

Las pruebas se realizaron en un total de 2 turnos de trabajo, con 1 probadora, toda la actividad estuvo dirigida por un Jefe de Pruebas.

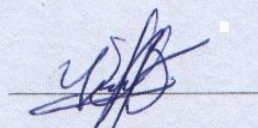
4 Evaluado por:

4.1 Especialista principal Asignado:

Ing. Yisel Niño Benítez

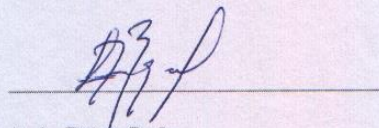
4.2 Otro personal especializado participante:

Nombres y Apellidos	Cargo
Ing. Tania Pérez Ramírez	RGA
Luis A. Rojas Peña	Estudiante



Ing. Yisel Niño Benítez

Asesor de Calidad



Luis A. Rojas Peña

Responsable por el Equipo de Desarrollo

**CENTRO DE INFORMATIZACIÓN
DE ENTIDADES**
Universidad de las Ciencias Informáticas
Carretera a San Antonio Km 2 ½. Torrens.
Boyeros. Ciudad de La Habana. Cuba
Teléfono: + 53 (7) 837 3680
E-mail: software.gestion@uci.cu