Universidad de las Ciencias Informáticas

Facultad 1

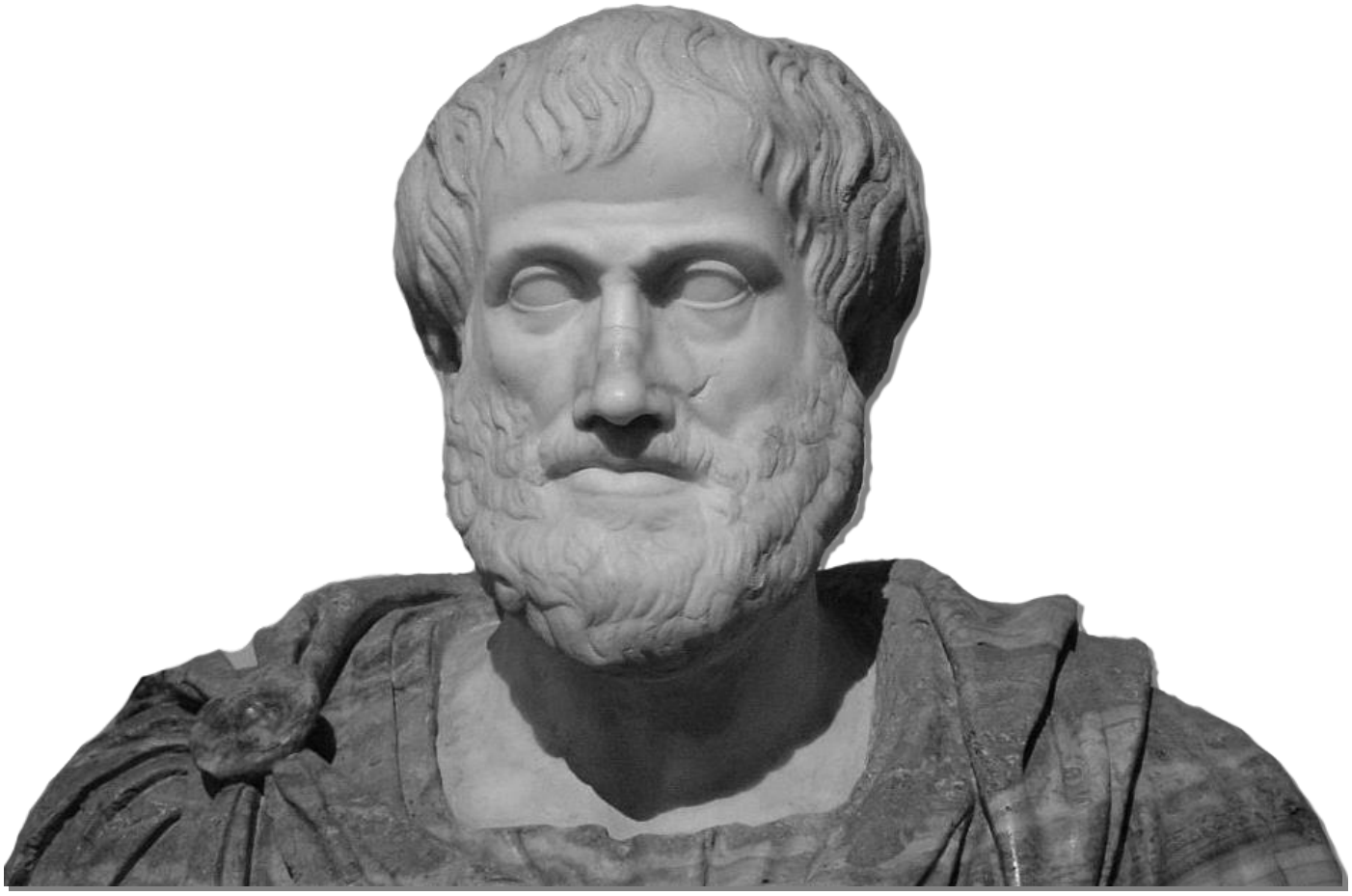
Módulo de contabilidad para el Sistema de Gestión de Información de la asignación, distribución y consumo de Combustible en la UCI

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

Autor: Geinel Dorta Guevara

Tutores: MSc. Radel Calzada Pando
MSc. Osiris Perez Moya
Ing. Raciel Roche Escobar

La Habana, Junio de 2015



"La inteligencia consiste no sólo en el conocimiento, sino también en la destreza de aplicar los conocimientos en la práctica."

Aristóteles

DECLARACIÓN DE AUTORÍA

Declaración de autoría

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

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

_____

Autor: Geinel Dorta Guevara.

_____ _____

Tutor: Ing. Raciel Roche Escobar. **Tutor: MSc**: Osiris Perez Moya.

AGRADECIMIENTOS

A mis padres por sus consejos y su apoyo en todo momento.

A mi hermano por su apoyo y confianza.

A mis abuelos que siguieron en cada momento de sus vidas mi trayectoria como estudiante.

A mis tutores que fueron siempre un apoyo en la elaboración del trabajo.

A mi oponente y miembros del tribunal que contribuyeron a la realización de este trabajo

A mis amigos por estar siempre cuando se les necesita.

A mis compañeros de aula que hemos compartido cinco años de momentos inolvidables.

A todos los profesores que han participado en mi formación académica.

A todas las personas que de una forma u otra han contribuido a mi carrera.

Geinel

DEDICATORIA

A mi mamá, mi papá y mi hermano por ser mi ejemplo y guía.

Geinel

RESUMEN

Resumen

Los sistemas contables han permitido optimizar todos los procesos en los cuales han estado vinculados, permitiendo una mayor organización y control de toda la información. Para lograr un adecuado control del consumo de combustible dentro la Universidad de las Ciencias Informáticas (UCI) es necesaria la interacción de diferentes áreas mediante el envío y respuesta de planificaciones y solicitudes garantizando el correcto flujo y control de la información y los recursos. Este proceso se realiza de forma manual, provocando deficiencias en su ejecución. Después de un estudio realizado se determinó que para gestionar estas informaciones es necesario el uso de aplicaciones informáticas que mejoren la gestión de estos procesos. En la actualidad existe un sistema desarrollado en la UCI que no se encuentra en uso por falta de un módulo que permita tener un registro y control contable de los flujos de información relacionados de la asignación, distribución y consumo de todo el combustible por vehículo. Para darle cumplimiento al desarrollo del módulo mencionado anteriormente es necesario realizar un estudio detallado del proceso a informatizar así como de los sistemas homólogos, herramientas, metodologías y lenguajes de programación que permitieran el posterior desarrollo de una propuesta de solución acorde a las necesidades existentes. Como resultado de la investigación se obtuvo un módulo que, luego de pasar por un proceso de pruebas en el que fue detalladamente revisado, garantiza una mejor gestión del flujo de información referente a la asignación, distribución y consumo de combustible.

Palabras clave: combustible, gestión contable, gestión económica, registros contables.

ÍNDICE

ÍNDICE

ÍNDICE

Índice de figuras

ÍNDICE

ÍNDICE DE TABLAS

Introducción

Con el paso del tiempo, el hombre ha ido dependiendo cada vez más de los recursos energéticos para la satisfacción de necesidades como: iluminación, calefacción, refrigeración y transporte. Esta dependencia energética ha acarreado un abuso de consumo de combustibles fósiles y recursos no renovables. La sociedad de consumo se extiende cada vez más utilizando recursos energéticos para disfrutar de un mayor confort, y a esta calidad de vida son llevados los países en vías de desarrollo donde la mayor parte de la estructura de oferta de energía primaria está basada en petróleo y gas en casi el 90 % a nivel mundial (PEREZ, GONZALEZ y MARTINEZ, 2013)

A nivel internacional la gestión energética se ha convertido en una pieza clave para que una organización, independientemente de su naturaleza o tamaño, pueda alcanzar niveles deseables de eficiencia y ahorro de energía; de compromiso con el medio ambiente y con la sociedad; así como de mejora de la competitividad en sus procesos productivos (SANCHEZ, GARCIA y WILSON, 2011).

La economía cubana enfrenta un déficit de suministro energético, lo cual se extiende en mayor o menor grado a todos los sectores de la actividad económica. Esta situación obliga a la dirección del país a tomar medidas y adoptar programas para enfrentar la crisis energética, cuyo alcance es sectorial y a nivel nacional. En el caso del sector empresarial son priorizadas las empresas exportadoras y de servicios sociales (SANCHEZ, 2011).

Dentro de las prioridades de asignación de suministro energético se encuentran las instituciones docentes y de investigación como es la Universidad de las Ciencias Informáticas (UCI). Tanto fuera como dentro de la UCI, se desarrollan actividades que demandan el traslado diario de un grupo de personas, ocasionando que se requiera distribuir y consumir combustible. Para lograr un eficiente consumo de combustible es necesaria la interacción de diferentes áreas mediante el envío y respuesta de planificaciones y solicitudes garantizando el correcto flujo y control de la información y los recursos. La Vicerrectoría Económica gestiona el proceso de controlar el correcto flujo de información asociado al gasto de combustible de toda la UCI.

Las planificaciones y solicitudes son almacenadas en formato duro, lo que trae consigo que: la información que exista esté duplicada, la respuesta ante la necesidad de obtener datos sea lenta y, existe un riesgo de deterioro asociado al almacenamiento.

El proceso de distribución y consumo del combustible en la UCI se realiza mensualmente, esto provoca un retardo en la elaboración y entrega de esta información al tener que pasar por varias áreas de la UCI para su aprobación. Esto ocasiona que se retrase la información antes de ser entregado el combustible. Esta lentitud dificulta la disponibilidad de la información para realizar la carga de las tarjetas y el consumo que cada vehículo hace por la tarjeta que tiene asignada. Al no conocer el consumo que cada vehículo tiene por día se dificulta conocer el monto que se alcanza en el gasto por cada vehículo. Además no posee un registro contable por cada vehículo del consumo que realiza de forma semanal y mensual, y no se tiene comunicación inmediata entre la caja y el grupo de finanzas principales encargados de la distribución y consumo del combustible.

Para gestionar estas informaciones es necesario el uso de aplicaciones informáticas que mejoren la gestión de estos procesos.

Todo lo anteriormente planteado induce a determinar el siguiente **problema de la investigación**: ¿cómo contribuir a la gestión de la información contable referente a la asignación, distribución y consumo de combustible de la UCI?

El **objeto de estudio de la investigación**, se enmarca en los sistemas de gestión de la información contable y el **campo de acción** se enfoca en los sistemas para la gestión de la información contable sobre la asignación, distribución y consumo del combustible en la UCI.

Para dar solución al problema antes descrito se tiene como **objetivo general**: desarrollar el módulo de contabilidad para el sistema de gestión de la información de la asignación, distribución y consumo del combustible de la UCI, mediante el uso de las tecnologías de la información, permitiendo la agilidad y protección de la información contable asociada a la asignación, distribución y consumo del combustible.

Se define como **objetivos específicos:**

1. Diagnosticar el estado actual y marco teórico referencial de la gestión de la información contable relacionada con la asignación, distribución y consumo de combustible.
2. Desarrollar un módulo para la gestión de la información contable relacionada con la asignación, distribución y consumo de combustible en la UCI.
3. Validar la solución desarrollada.

Se define como **idea a defender** que: el desarrollo de un módulo para la gestión de la información contable de la asignación, distribución y consumo de combustible en la UCI, permitirá una mayor agilidad y protección de la información contable asociada a la asignación, distribución y consumo del combustible y de los datos económicos que se generan.

Para darle cumplimiento a los objetivos específicos se plantean las siguientes **tareas de la investigación**:

1. Revisar los documentos y entrevistas a directivos y trabajadores de la Dirección de Planificación y Estadísticas así como de la Dirección de Contabilidad y Finanzas y Vicerrectoría Económica.
2. Buscar los elementos comunes y necesarios que se ajustan a las necesidades de la gestión del combustible de forma contable.
3. Diseñar el módulo para el sistema de Gestión de la Información Contable referente al combustible.
4. Implementar el módulo para el sistema de Gestión de la Información Contable, referente al combustible de forma contable en la UCI.
5. Elaborar los casos de prueba para la propuesta de solución.
6. Ejecutar los casos de pruebas.

Con el propósito de desarrollar las tareas planteadas para la investigación se utilizaron los siguientes métodos científicos:

- **Métodos teóricos:**

Analítico-Sintético: con la utilización del método se han buscado, investigado y analizado documentos para extraer los elementos del proceso de asignación, distribución y consumo del combustible a nivel nacional e internacional, que pudieran estar relacionados con el objeto de estudio; realizando consultas a diversas fuentes bibliográficas.

Análisis Histórico-Lógico: se utilizó en el estudio de los antecedentes, la evolución y el desarrollo que han tenido los sistemas de asignación, distribución y consumo del combustible.
Modelación: se utilizó para representar los procesos definidos por el negocio mediante la construcción de modelos y diagramas a lo largo del desarrollo de la investigación, simplificando la realidad y facilitando la comprensión de los mismos.

- **Métodos empíricos:**

Entrevista: se realiza una entrevista a los especialistas de las áreas: Dirección de Transporte, Grupo de Gestión Energética y Vicerrectoría Económica, con el objetivo de lograr una mayor comprensión de las particularidades y características del proceso de distribución del combustible. De la entrevista, se obtuvo la lista de requisitos funcionales del sistema.

Los aportes prácticos esperados con la solución que se implementa son los siguientes:

- Brindar funcionalidades que permitan la gestión del combustible en la UCI desde el punto de vista de registro y control contable.
- Garantizar la seguridad e integridad de los datos así como que no exista informaciones duplicadas.
- Proporcionar la información necesaria con rapidez y calidad entre las áreas involucradas en el proceso.

La estructura del trabajo de diploma está compuesta por un resumen, una introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y los anexos. A continuación, se describen los principales aspectos abordados en cada uno de los capítulos:

- **Capítulo I: Fundamentación teórica de los Sistemas Contables de asignación, distribución y consumo de combustible.**

En este capítulo se puntualizan los principales conceptos relacionados con el tema que se aborda en la investigación. Se realiza el estudio del estado del arte a nivel nacional e internacional sobre los sistemas contables, además de definirse las tecnologías, metodologías de desarrollo de software y herramientas a utilizar durante el desarrollo de la aplicación.

- **Capítulo II: Diseño del Módulo de contabilidad para el Sistema de Gestión de Información de la asignación, distribución y consumo de combustible en la UCI.**

En este capítulo se realiza la modelación de la solución de propuesta así como su funcionamiento y se hace un análisis del modelo del negocio correspondiente al sistema. Se identifican los requisitos funcionales y no funcionales, se muestran los artefactos de la metodología de desarrollo de software seleccionada y se describen los diagramas correspondientes al diseño del sistema, así como el modelo de datos.

- **Capítulo III: Implementación y pruebas del Módulo de contabilidad para el Sistema de Gestión de Información de la asignación, distribución y consumo de combustible en la UCI.**

En este capítulo se abordan aspectos relacionados con la implementación del sistema en base a la arquitectura de desarrollo de software definida. Se documentan las pruebas de software realizadas al Sistema de Gestión de la Información Contable de la asignación, distribución y consumo del combustible en la UCI.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA DE LOS SISTEMAS CONTABLES DE ASIGNACIÓN, DISTRIBUCIÓN Y CONSUMO DE COMBUSTIBLE

1.1. Introducción

En el capítulo se presentan los conceptos fundamentales que constituyen la base teórica de los procesos de gestión contable del combustible. Se realiza un estudio de los sistemas homólogos tanto a nivel nacional como internacional. Se realiza un análisis de las metodologías de desarrollo de software, tecnologías, y herramientas actuales que guiarán el proceso de desarrollo de software y se seleccionan las adecuadas a utilizar durante el desarrollo de la solución que se propone.

1.2. Conceptos asociados a la gestión de distribución y consumo del combustible

De los conceptos asociados a la gestión de distribución y consumo del combustible fueron seleccionados los siguientes:

1.2.1. Recursos energéticos

Se denomina recursos energéticos a los medios o recursos que ofrece la naturaleza, y a partir de los cuales, mediante un proceso industrial, se obtiene alguna forma de energía que puede ser directamente utilizada por el consumidor o por alguna actividad productiva. Los recursos energéticos pueden ser: sólidos, como el carbón o la biomasa y líquidos, como el petróleo o el gas natural (E-DUCATIVA, 2014).

1.2.2. Combustible

El combustible es cualquier material que es capaz de liberar energía una vez que se oxida de manera violenta y con desprendimiento de calor. Los combustibles fluidos son mayormente empleados a instancias de motores de combustión interna, destacándose el gasóleo, el querosene, la gasolina o nafta. Otro combustible es el llamado combustible fósil, que es aquel que se ha formado hace millones de años en el planeta a partir de los restos orgánicos de plantas y animales muertos, y tras miles de años se produjeron las reacciones químicas que transformaron tales restos en carbón, gas y petróleo (COMBUSTIBLES, 2004).

1.2.3. Sistema de gestión

Un sistema de gestión ayuda a lograr los objetivos de una organización mediante una serie de estrategias que incluyen la optimización de procesos, centrándose en la gestión de la información (RODRIGUEZ y DIAZ, 2013).

1.2.4. Gestión económica

La gestión económica es la encargada de llevar a vía de hecho toda la actividad de una organización con eficiencia y eficacia con vista a que las empresas obtengan resultados que pueden ser positivos o negativos de acuerdo al manejo dicha gestión realizada. Está caracterizada por una visión amplia de las posibilidades reales de una organización para resolver determinada situación o arribar a un fin establecido (VELÁZQUEZ y DARROMÁN, 2011).

1.2.5. Sistema de gestión de flotas de equipos de transporte

Un sistema de gestión de flotas de equipos de transporte, le permite a una institución controlar los vehículos pertenecientes a la misma, tales como: coches, camiones y motores. Este tipo de sistema de gestión incluye entre sus funciones: el mantenimiento de vehículos, financiación, seguimiento de los mismos, gestión del combustible, gestión de conductores, control de las fechas de vencimiento de las revisiones técnicas de los vehículos y vencimiento de las licencias de conducir de los conductores (IDEA, 2009) y (PRADO, 2010).

1.3. Análisis de soluciones homólogas. Sistemas de gestión económica

1.3.1. Internacional

- **ContaPyme**: (Sistema de gestión empresarial y contable para PYMES[1]), es una herramienta que maneja de forma integrada las áreas económicas de gestión de una organización. Este sistema incorpora un conjunto de herramientas para manipular los datos y la obtención de resultados. Una de las características de este sistema es que utiliza un registro de información sencillo, de tal forma que lo convierten en un instrumento fácil de utilizar y ágil en el seguimiento y en la toma de

[1] Pequeñas y medianas empresas.

medidas, pero dentro de sus funciones realiza el control del combustible de forma sencilla sin crear expedientes económicos de los registro y control de la asignación, distribución y consumo del combustible (CONTAPYME, 2014).

- **SAP:** es uno de los Sistema de Recursos de Empresas (*Enterprise Resource System*, en inglés abreviado ERP) utilizado en el mundo. Este sistema tiene módulos integrados que abarca todas las áreas de una organización. Cada módulo realiza una función diferente, pero está diseñado para que la información adquirida se comparta entre todos sus módulos (INFORMATICA HOY, 2007), y no crea expedientes económicos sobre la asignación, distribución y consumo del combustible.

1.3.2. Nacional

- **CONDOR**: sistema automatizado de alta complejidad y seguridad que abarca todos los aspectos del proceso contable de una entidad. Está formado por varios módulos como activos fijos, contabilidad general, nóminas, control de inventarios, recursos humanos, entre otros. Este sistema no crea registros del proceso contable relacionado con el combustible, además de ser privativo (SANCHEZ, 2011).

- **ASSETS[2]:** es el sistema utilizado en la Vicerrectoría Económica de la UCI para la gestión de los procesos contables. Es un sistema flexible, con ayuda en línea, tiene pantallas de entradas de datos con opciones fáciles de interpretar y ejecutar, facilita el uso de la parametrización para adaptarse a las exigencias de cada cliente que lo utilice, con la emisión de varios reportes que tendrán la forma y el contenido que el usuario defina. También facilita la ejecución de auditorías contables para localizar errores de compatibilidad de datos. Este sistema registra por parte de los clientes que lo trabajan las cifras que se presenten, así como el concepto por el que se utiliza la cifra, nunca registra por concepto de Expediente de Combustible, además de no registrar la asignación, distribución y consumo del combustible de cada vehículo según la demanda asignada y consumida, ya sea por el plan asignado o por el extra que se le asigna (PACHECO, 2010).

[2] Sistema de Gestión Integral

1.4. Análisis de soluciones homólogas. Sistemas de gestión de distribución y consumo del combustible

1.4.1. Internacional

En el mundo existen numerosos sistemas de gestión del combustible que han sido desarrollados con el propósito de satisfacer necesidades existentes en diferentes organizaciones, los cuales permiten controlar de forma efectiva la asignación, distribución y consumo del combustible y otros indicadores de los vehículos asociados a dichas entidades. Algunos de estos son: Gestión de Flota, KreaFlota y FlotaWeb.

- **Gestión de Flota**

El sistema de gestión de flotas EUGCOM [3]es un sistema computacional diseñado para la sólida y correcta administración de flotas de vehículos. Permite administrar de una manera óptima toda la información de los vehículos que la empresa maneja y administra. Algunos de sus módulos son:

- **Combustibles**: módulo de ingreso y mantención de los datos que componen el movimiento de cargas de combustible efectuados a los vehículos.
- **Cargas de combustible:** módulo de búsqueda de datos que permite ejecutar consultas y obtener resultados acerca de las cargas de combustible registradas en el sistema. Genera un informe sumamente personalizado según los requerimientos del usuario y permite calcular los datos importantes por cada vehículo.
- **Reprocesamiento de cargas de combustible:** módulo que permite reprocesar los datos automáticos generados en el movimiento de cargas de combustible como distancia y rendimiento, y genera un informe de errores de las cargas de combustible registradas en el sistema, y es capaz de reparar automáticamente los errores usuales (EUGCOM, 2009).

- **KreaFlota**

[3] Sistema de Gestión Administrativa

Es un software diseñado y concebido para controlar los aspectos técnicos de cualquier flota vehicular, dispone de módulos para la correcta gestión de cada uno de los vehículos que disponen las empresas. Lleva un control de los vehículos de la flota, para esto prepara una ficha en la cual se especifican todos los datos de cada uno de los vehículos, controlando alarmas de cambios de aceites, revisión técnica y pólizas de seguro. Además, se lleva un control exhaustivo del petróleo, llevando un registro diario de consumo para los vehículos, permitiendo obtener rendimientos de los mismos (KREASOFT, 2010).

- **FlotaWeb**

Es una plataforma de administración de flotas, que permite gestionar todos los procesos relacionados con la flota vehicular. Puede controlar todos los costos, evaluar rendimientos y costos por kilómetros u horas, administrar eficientemente las llantas y el combustible. Dos de sus módulos son los siguientes (FLOTAWEB, 2010):

- **Vehículos:** centraliza la información completa de los vehículos y todos los procesos que tienen que ver con su correcta administración.

- **Combustible:** permite controlar y supervisar los costos de combustible en la flota, analizar rendimientos, y obtener informes estadísticos de los detalles de consumo por proveedor, ciudades, centros de costo y operaciones.

Todos los sistemas internacionales mencionados, se analizaron con el objetivo de conocer cómo gestionan la información relacionada con la asignación, distribución y consumo del combustible. Estos, cuentan con un módulo para la gestión del combustible que permite tener un registro diario del consumo de los vehículos en función de obtener su rendimiento, generar informes y calcular los datos más importantes por cada vehículo. Sin embargo, sus funcionalidades no se pueden reutilizar como parte de la propuesta de solución del Sistema de Gestión de la Información de la asignación, distribución y consumo del combustible en la UCI, debido a que no se crean registros contables de las tres acciones previstas a realizar en el sistema que se propone, esto imposibilita tener un registro exacto de los consumos por vehículos ya sea por asignación mensual o por extra que sea asignado para su consumo.

1.4.2. Nacional

En Cuba a lo largo de los años se han implementado diversos sistemas informáticos para el control estadístico y financiero, particularmente en la administración de finanzas en diversos sectores como el transporte. Algunos de los sistemas son: Apolo, Sistema de Gestión de Mantenimiento Vehicular y SISCOMPA.

- **Apolo**

Es un producto nacional perteneciente al Ministerio de las Comunicaciones (MINCOM) y la Empresa Cubana Nacional de Software (DESOFT), encargada de su desarrollo. Es un sistema basado en el estilo cliente-servidor, donde la base de datos está en un servidor central y los usuarios del sistema, independientemente del lugar donde se encuentren, accederán en tiempo real. Es una herramienta enfocada a gestionar todo tipo de flotas de equipos de transporte y ofrecer soluciones de asistencia a la gestión y toma de decisiones, con el objetivo de optimizar la rentabilidad de la inversión mediante la reducción de los costos fijos y variables. Apolo brinda una potente base informativa, donde la mayoría de la información que se brinda es directamente definible por un usuario de nivel administrativo del sistema aunque es una base informativa de los procesos, no realiza la gestión contable. Para esto se precisan algunos nomencladores como: marca de vehículos, tipos de combustible, usos de cada uno de los tipos de flotas de equipos de transporte, de la asignación de combustibles, lubricantes y líquidos de freno, la cantidad de Km programados para cada marca o línea e información del personal de la entidad (SAAVEDRA, 2013).

- **Sistema de Gestión de Mantenimiento Vehicular**

Es un producto realizado por el Centro de Informatización de la Gestión de Entidades (CEIGE) de la UCI, tiene como propósito gestionar los procesos de mantenimiento que se desarrollan en el área de Transporte de la Policía Nacional Bolivariana. Este sistema fue desarrollado basado en los principios de independencia tecnológica utilizando software libre. Mediante su utilización en la Policía Nacional Bolivariana puede gestionar las asignaciones, recepciones, accidentes, inspecciones técnicas, órdenes de trabajo, informes de baja y de resultado de las unidades policiales de forma centralizada, eficiente y segura. Este sistema de gestión tiene toda la información referente a los procesos de mantenimiento

preventivo planificado y correctivo de una flota de vehículos, sin importar el tamaño de esta y garantizando una correcta planificación, organización y control en la gestión del mantenimiento de los vehículos (SAAVEDRA, 2013). Presenta los siguientes módulos:

- **Configuración:** es el módulo a través del cual se definen los diferentes grupos de unidades de acuerdo a su marca, modelo, régimen de mantenimiento, entre otras características. También se define el umbral de mantenimiento por el cual se van a regir todas las unidades policiales para la realización de los mantenimientos preventivos planificados que le corresponden.
- **Vehículo:** es el módulo desde el cual se va a gestionar toda la información de las unidades policiales (adicionar, modificar, consultar, asignar a dependencias, registrar accidentes, realizar inspecciones técnicas y recepción de unidades) (SAAVEDRA, 2013).

➢ **SISCOMPA**

El sistema fue creado por la Empresa de desarrollo de software para el transporte (TRANSOFT), garantiza el control y la gestión de la flota automotor de transporte de carga, contribuyendo al ahorro de recursos materiales, combustible y tiempo. Está compuesto por módulos técnicos que intercambian información entre ellos, los cuales son: tráfico, técnica, seguridad automotor y portadores energéticos. También posee un módulo de dirección para los directivos y un módulo de administración. Esta herramienta permite potenciar la actividad de ahorros en combustibles, lubricantes, neumáticos y baterías. Ofrece a los directivos la información necesaria para la toma de decisiones, siendo una poderosa herramienta de dirección (TRANSOFT, 2010).

➢ **SIGECOM**

Es un sistema creado por la UCI para el control de la asignación, distribución y consumo de combustible en la UCI. Este sistema, cuenta con varios módulos los cuales permiten tener un registro, el cual posibilita crear un expediente de toda la distribución y consumo del combustible por vehículo. Además posee dentro de sus módulos uno que permite hacer reportes por varios tipos de necesidades las cuales aparecen reflejadas en el sistema (VALDIVIÉ y HERNANDEZ, 2014).

El software creado con tecnología libre, se puso en práctica por la UCI durante 3 meses, permitiendo un mejor control de todo el combustible que le es asignado mensual al centro, pero se decide dejar de

utilizarlo, pues el mismo no posee un módulo que permita tener un registro y control contable de la asignación, distribución y consumo de todo el combustible por vehículo, proceso el cual debe convertirse en una de las tareas principales del producto para culminar todo el ciclo de vida del combustible en el centro.

Los sistemas nacionales anteriormente mencionados se analizaron con el objetivo de conocer cómo gestionan toda la información relacionada con el control del combustible. Los sistemas, Apolo, Sistema de mantenimiento vehicular y SISCOMPA cuentan con un módulo para la gestión del combustible que permiten realizar los registros del consumo de los vehículos en función de obtener su rendimiento y generar informes. Sin embargo, sus funcionalidades no se pueden reutilizar como parte de la propuesta de solución, debido a que el proceso de asignación, distribución y consumo del combustible que realizan los sistemas no están acorde con el proceso que realiza la UCI y por ende no manejan la información necesaria para gestionar la información del combustible en la UCI.

Luego de realizar una investigación sobre los sistemas de gestión del combustible, se arriba a la conclusión de que de todos los sistemas estudiados, solo se asemeja al proceso a desarrollar, el sistema SIGECOM. Creado por el Centro de Ideoinformática (CIDI), al cual se le debe realizar un módulo más para lograr el registro y control contable de la asignación, distribución y consumo de todo el combustible asignado a la UCI de forma mensual. Este módulo a desarrollar debe seguir las de la soberanía tecnológica, haciéndose necesario el desarrollo del sistema sobre herramientas y tecnologías libres basadas en la web.

1.5. Metodologías de desarrollo de software

Las metodologías para el desarrollo del software imponen un proceso disciplinado con el fin de lograr eficiencia en la realización del producto. Este tipo de metodología tiene como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. Se encarga de estructurar, planificar y controlar el proceso de desarrollo del software, a través de un conjunto de procedimientos, técnicas, herramientas y soporte documental que ayuda a los desarrolladores a realizar un nuevo software (GRAÑA, 2011). A continuación se mencionan metodologías que fueron estudiadas para el desarrollo de la presente investigación.

➢ Metodologías ágiles

Las metodologías ágiles, se fundamentan en la entrega temprana del software con el uso de métodos no formales, ya que enfocan su mayor esfuerzo en la elaboración y entrega del producto. Se nombran ágiles por la capacidad de responder rápida y efectivamente ante los cambios, apoyándose en las habilidades y experiencias personales del equipo. Estas se basan en el "Manifiesto Ágil" el cual plantea los principales aspectos que se valoran en el desarrollo ágil, es decir, se valora (LETELIER, 2012):

- Al individuo y las interacciones del equipo de desarrollo sobre el proceso y las herramientas.
- Desarrollar software que funcione, más que conseguir una buena documentación.
- La colaboración con el cliente, más que la negociación de un contrato.
- Responder a los cambios, más que seguir estrictamente un plan.

Al ser el equipo de trabajo pequeño, no se cuenta con una definición detallada y exacta de todos los requisitos que debe cumplir el sistema, implicando una alta probabilidad de ocurrencia de cambios de forma gradual, a medida que se vayan obteniendo las versiones del sistema. Por ello, es conveniente el uso de una de las metodologías ágiles, diseñadas para pequeños equipos de desarrollo y preparadas para enfrentar cambios durante la creación del software.

A continuación se describe algunas de las metodologías ágiles existentes:

Programación Extrema

Programación Extrema (*eXtreme Programming*, en inglés abreviado XP) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores y propiciando un buen clima de trabajo. XP se basa en retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y cambiantes, y donde existe un alto riesgo técnico, como metodología de desarrollo posee etapas en su desarrolla de software las cuales son planeación, diseño, codificación y pruebas. Es recomendable emplearlo solo en

proyectos a corto plazo y en caso de fallar existen altas comisiones por el desarrollo de software (PRESSMAN, 2005).

Scrum

Scrum es una metodología ágil dirigida a la gestión de proyectos. Precisa un proceso empírico e incremental del desarrollo de software. Sus iteraciones son de un proceso corto de 3 o 4 semanas y cada iteración va enfocada a la terminación parcial del software ejecutable que incorpora nuevas funcionalidades. Surge como modelo para el desarrollo de productos tecnológicos y actualmente se emplea en los entornos que trabajan con requisitos inconsistentes y requieren rapidez y flexibilidad. Busca entregar un software que realmente resuelva las necesidades del cliente, aumentando su satisfacción.

Esta tiene un simple y pequeño conjunto de reglas y está basado en los principios de inspección continua, adaptación, autogestión e innovación (FIGUEROA, SOLÍS y CABRERA, 2010).

Proceso Unificado Abierto

Proceso Unificado Abierto (en inglés abreviado OpenUp) Es un modelo de desarrollo de software, desarrollado por la fundación Eclipse. Esta preserva las mejores prácticas de RUP (según sus siglas en inglés *Rational Unified Process*, Proceso Unificado de *Rational*), por lo que entre sus principales características se mantiene un desarrollo iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura para reducir al mínimo los riesgos y organizar el desarrollo. Esta metodología está diseñada para equipos pequeños ya que se espera obtener resultados en un corto período de tiempo y utilizar los procesos, productos, roles, y tareas que sean indispensables para el mismo. Por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas. El ciclo de vida de OpenUp consiste de cuatro fases: Concepción, Elaboración, Construcción y Transición (TORRES, 2008).

Consideraciones sobre las metodologías ágiles

Después del análisis de las metodologías de desarrollo de software antes expuestas, y en concordancia con las particularidades de la solución a desarrollar en la presente investigación, se elige la metodología XP para realizar el módulo de contabilidad para el sistema de gestión de información de la asignación, distribución y consumo de combustible en la UCI. Esta metodología es ajustada a la propuesta de solución

gracias a la retroalimentación continua entre el cliente y el equipo de desarrollo, garantizando así un fluido intercambio de ambos participantes, lo que agiliza toma de decisiones. En sus principios básicos XP plantea la programación en equipos pequeños con pocos roles, pudiendo intercambiar responsabilidades en un momento determinado. Es además una metodología abierta a los cambios y genera poca documentación lo que hace la entrega del software menos complicada y satisfactoria.

1.6. Herramientas y tecnologías

En la actualidad existe una gran revolución de los medios informáticos, donde los avances en las nuevas tecnologías tienen el propósito de ofrecer soluciones adaptables a las exigencias del usuario final. La integración de las tecnologías de desarrollo con las herramientas, ha surgido como una alternativa que permite elaborar aplicaciones web para la gestión de la información. Además permiten resolver problemas actuales de una organización, facilitando así, la interacción de los usuarios con los sectores involucrados en la problemática.

El módulo a desarrollar se integrará al sistema SIGECOM, esto condiciona la reutilización de las herramientas empleadas en el desarrollo de dicho sistema.

1.6.1. Tecnologías de programación del lado del servidor

PHP[4]. En su versión 5.4.9 es una tecnología del lado del servidor, de código abierto y diseñado en sus inicios para el desarrollo de páginas web dinámicas. Una de sus principales características es que se integra con varios sistemas de bases de datos. También permite la integración con bibliotecas externas permitiéndole al programador analizar código XML y generar documentos en diferentes formatos. Este lenguaje es rápido, libre y multiplataforma, pues permite ser utilizado sobre diferentes sistemas operativos como: Linux y Windows. Posee además una librería de funciones y cuenta con una extensa documentación, la cual permite un rápido aprendizaje (DOYLE, 2012).

Ventajas que proporciona:

[4] Lenguaje de programación

- ✓ Permite la integración con bases de datos entre las que se encuentran: MySQL, PostgreSQL y Oracle.
- ✓ Permite emplear técnicas de programación orientada a objetos.
- ✓ Es libre y multiplataforma.

Desventajas:

- ✓ Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser ineficiente a medida que las solicitudes aumenten de número.

Se determinó como tecnología de programación PHP versión 5.4.9, el cual está orientado al desarrollo web, es de gran velocidad por lo que no requiere de amplios recursos de hardware y además se integra perfectamente con varios servidores. Es libre y está disponible bajo la licencia GPL (por sus siglas en inglés *General Public License, Licencia Pública General)*, es multiplataforma por lo que no tendrá ningún inconveniente al usarlo en cualquier computadora de la Universidad. Se caracteriza por la simplicidad de su código y por la abundante documentación que existe. Además se requiere utilizar este lenguaje de programación debido a que es necesario desarrollar una aplicación web para acceder al sistema a través de un servidor web por diversos departamentos de la Universidad y resulta engorroso tener que instalar en todas las estaciones de trabajo la aplicación para que sea accedida por los usuarios.

1.6.2. Lenguaje de modelado

UML[5]. En su versión 2.0, es un lenguaje gráfico utilizado para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. Este lenguaje es semejante al de la vida real, claro y uniforme para el diseño orientado a objetos, ya que permite la fuerte integración entre herramientas, procesos y dominios. UML se especializa en el modelado de elementos conceptuales como son: procesos de negocio y funciones de los sistemas. También este lenguaje permite definir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables (FERREIRA, 2013).

Marco de trabajo (*framework*) de Interfaz **Bootstrap:** En su versión 3.0.1 es un entorno de trabajo definido con artefactos o módulos de software concreto, desarrollado por Twitter. Esta tecnología simplifica el proceso de creación de la interfaz de usuario combinando HTML, JavaScript y CSS (por sus siglas en

[5] Lenguaje unificado de modelado

inglés *Check Cascading Style Sheets,* Hojas de Estilo en Cascada). Además se adapta a los distintos navegadores con numerosos componentes web como: botones, etiquetas, alertas, entre otros definidos en la web. Bootstrap fue programado para dar soporte a CSS 3 y HTML 5. Bootstrap está diseñado para todos los niveles: diseñador, desarrollador y principiante. Este *framework* se utiliza para hacer fácil y rápida su implementación. Bootstrap está definido por módulos que son reutilizables e independientes en la página web (BOOSTRAP, 2013).

Ventajas que proporciona:

- ✓ Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como *tablets* y móviles a distintas escalas y resoluciones.
- ✓ Se integra perfectamente con las principales librerías JavaScript[6], por ejemplo JQuery[7].
- ✓ Ofrece un diseño sólido usando LESS[8] y estándares como CSS3/HTML5.
- ✓ Es un *framework* ligero que se integra de forma limpia en nuestro proyecto actual.
- ✓ Funciona con todos los navegadores, incluido Internet Explorer usando HTML Shim para que reconozca los *tags* HTML5.
- ✓ Dispone de distintos *layout* predefinidos con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos.

Como *framework* de interfaz se determinó hacer uso de Bootstrap versión 3.0.1 por competente en la realización de interfaces web, además de ser recomendable para sitios web, se adapta a los distintos navegadores de la Universidad. Los diseños creados con Bootstrap son simples, limpios e intuitivos, permitiéndole ganar al sistema en agilidad a la hora de cargar y al adaptarse a otros dispositivos. Se requiere utilizar el *framework* debido a que brinda componentes para realizar el diseño de las vistas del sistema de forma organizada.

1.6.3. Framework de desarrollo

Symfony. En su versión 2.1.8 es un *framework* diseñado para optimizar, el desarrollo de las aplicaciones web basado en el patrón de arquitectura de software Modelo Vista Controlador (MVC). Proporciona

[6] Lenguaje de programación interpretado
[7] Biblioteca de JavaScrip
[8] Lenguaje de hojas dinámico de estilo

herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Utiliza *twig* para sus interfaces, un poderoso motor de plantillas. Symfony puede ser completamente personalizado para cumplir con los requisitos de las empresas que disponen de sus propias políticas y reglas para la gestión de proyectos y la programación de aplicaciones.

Ventajas que proporciona:

- ✓ Tiene su propia forma de trabajo, con variantes del MVC clásico como la capa de abstracción de base de datos, el controlador frontal y las acciones.
- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ El completo sistema de registro (*log)* permite a los administradores acceder hasta el último detalle de las actividades que realiza la aplicación
- ✓ Fácil de instalar y configurar en la mayoría de plataformas de programación.
- ✓ Sencillo de usar y flexible.

Desventajas

- ✓ Gran parte de la velocidad de Symfony se debe a un uso extensivo del caché por lo que cuando se está desarrollando tiende a ser algo tedioso tener que estar limpiando el caché de vez en cuando.
- ✓ Se necesita un VPS (por sus siglas en inglés *Virtual Private Server*, Servidor Virtual Privado) para poder publicar aplicaciones en la web porque es necesario poder descargar e instalar en un servidor para que Symfony funcione apropiadamente (POTENCIER y ZANINOTTO, 2010).

Entre los *frameworks* para PHP se escogió Symfony versión 2.1.8 debido a que posee una abundante documentación tanto en español como en inglés, tiene una variedad de *plugins*, que permiten agilizar el trabajo, reduce el tiempo de desarrollo pues proporciona herramientas y clases que pueden ser reutilizadas por los desarrolladores. Ofrece características permitiendo obtener aplicaciones web robustas y seguras.

Netbeans. En su versión 7.3, es un IDE bajo licencia GPL y de código abierto. Esta herramienta tiene la finalidad de permitirle a los desarrolladores crear diferentes sistemas y proyectos orientados sobre todo a la creación de soluciones en lenguaje Java, ya sea que se encuentren en Java SE (Edición Estándar) o

Java EE (Edición Empresarial), además de soportar otros lenguajes tales como PHP y JavaScript (JACOME, 2009). Se requiere utilizar Netbeans debido a que facilita el desarrollo del sistema, al proveer un entorno de desarrollo profesional para desarrollar sistemas web.

1.6.4. Servicios web

Apache. En su versión 5.4.3, es un servidor web de código abierto, altamente configurable y modular. Utiliza Perl, PHP y otros lenguajes *scripts*. Su función principal es analizar cualquier archivo solicitado por un navegador y mostrar resultados correctos de acuerdo con el código del archivo. Permite configurar los informes de errores, presenta visualización de códigos en numerosos niveles y la capacidad de determinar qué nivel del navegador puede aceptar el contenido. Es uno de los primeros servidores en soportar *host* basados en direcciones IP (por sus siglas en inglés *Internet Protocol,* Protocolo de Internet) y *host* virtuales. Tiene un elaborado índice de directorios, un directorio de alias, informe de errores HTTP HTML (por sus siglas en inglés *Hypertext Transfer Protocol*, Protocolo de Transferencia de Hipertexto) configurable, gestión de recursos para procesos hijos, reescritura de las URL (por sus siglas en inglés *Uniform Resource Locator,* Localizador de Recurso Uniforme), comprobación de ortografía de las URL y manuales *online* (NARAMORE, 2005).

Ventajas que proporciona:

- ✓ Esta incluye formatos de configuración no estándar.
- ✓ Es multiplataforma.
- ✓ Permite elegir el servidor web Apache.
- ✓ Soporta múltiples plataformas por lo que genera mayor usabilidad, dando la opción de utilizar diferentes sistemas operativos sin ningún problema.

Desventajas:

- ✓ No dispone de un entorno integrado con una sofisticada interfaz de usuario, asistente y ayuda en línea (ARRENDONDO, 2012).

Se determinó como servidor web Apache versión 2.2.20 porque no requiere la utilización de muchos recursos y permite además que los lenguajes puedan ser utilizados del lado del servidor. Posee además la capacidad de permitir la protección por contraseñas de las páginas de un gran número de usuarios, la visualización del código HTML en varios niveles, registra los errores en varios formatos y tiene soporte para *host* virtuales. Se requiere utilizar el servidor web Apache debido a que la información que maneja el proceso de gestión de la información del combustible es accedida por diversos departamentos de la Universidad y resulta engorroso tener que instalar en todas las estaciones de trabajo la aplicación para que sea accedida por los usuarios. Además, se prevé en el futuro que se incremente el número de usuarios conectados concurrentemente, por lo que se requiere la implantación de una aplicación cliente-servidor.

1.6.5. Lenguajes de modelado. Herramientas CASE

Visual Paradigm. Es una herramienta CASE que utiliza el lenguaje de modelado estándar UML, permite la generación de códigos e ingeniería inversa. Esta herramienta cumple con las políticas de migración a software libre en Cuba, ya que es una herramienta multiplataforma que se puede utilizar tanto en Linux como en Windows. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código para entornos integrados de desarrollo tales como: NetBeans, Eclipse, Oracle JDeveloper y JBuilder (HOWARD, 2012).

Ventajas que proporciona:

- ✓ Tiene una interfaz intuitiva y es de fácil aprendizaje para los desarrolladores.
- ✓ Permite la generación automática de diagramas a partir de descripciones de casos de usos.
- ✓ Permite hacer descripción de los casos de usos dando una gran variedad de plantillas predeterminadas permitiendo personalizarlas.
- ✓ Combina las funcionalidades en una amplia plataforma de modelado visual.

Como herramienta CASE fue seleccionada *Visual Paradigm* versión 8.0, pues propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código para entornos integrados de desarrollo tales como: NetBeans, Eclipse, Oracle JDeveloper y JBuilder. Al ser multiplataforma *Visual Paradigm* cumple con las políticas

actuales de migración a software libre, de tal forma que facilite la modelación del software libre independientemente del sistema operativo que se emplee. Se requiere utilizar la herramienta CASE debido a que la información que maneja el proceso de gestión de la información del combustible es extensa y se hace necesario modelar todos los proceso que se realizan para realizar la distribución y consumo del combustible en la UCI.

1.6.6. Gestores de base de datos

PostgreSQL. En su versión 9.1, es uno de los motores de base de datos relacionales potentes que existen actualmente. Permite ejecutar consultas SQL, las cuales posibilitan actualizar, insertar, eliminar y realizar reportes sobre los datos almacenados en ficheros o bases de datos. Ofrece la posibilidad de ejecutar y trabajar varios procesos al mismo tiempo sobre la misma tabla sin ser dañada, donde cada usuario obtiene una versión de lo último que ha hecho evitando la pérdida de información. Tiene su propio lenguaje PL/PgSQL, pero también se pueden usar lenguajes como C, C++, Gambas, Java PL, Java Web, Perl, Php, Python (NOVELLA, 2012). Este gestor ofrece muchas ventajas respecto a otros sistemas de bases de datos como son:

- ✓ Mejor soporte que los proveedores comerciales: tiene una importante comunidad de profesionales.
- ✓ El código fuente está disponible para todos de manera gratuita.
- ✓ Multiplataforma: PostgreSQL está disponible tanto para Linux como para Windows.

Fue seleccionado como sistema gestor de bases de datos PostgreSQL versión 9.1, ya que es uno de los potentes motores de base de datos relacionales que existen actualmente, además de ser multiplataforma, y a diferencia de MySQL, es libre. Como administrador de bases de datos para este gestor se seleccionó pgAdmin en su versión 1.16.1, pues está diseñado para satisfacer las necesidades de todos los usuarios, teniendo en cuenta la realización de consultas SQL sencillas, hasta el desarrollo de bases de datos de alta complejidad. Se requiere utilizar el motor de base de datos debido a que la información que maneja el proceso de gestión de la información del combustible es creciente en el tiempo y se hace necesario almacenarla y acceder a ella a través del sistema.

1.7. Conclusiones

- El análisis de las características y funcionamiento de los Sistemas de gestión de información de la distribución y consumo del combustible en el ámbito nacional e internacional, permitió comprobar que los sistemas estudiados no poseen las características específicas de la flota de equipos de transporte de la Universidad para realizar el proceso de distribución y consumo del combustible en la UCI, necesitando la creación del módulo para la información contable del sistema existente SIGECOM desarrollado en dicho centro de estudios que permitirá llevar el registro y control contable de la asignación, distribución y consumo de todo el combustible.

- El estudio sobre las herramientas, tecnologías y metodologías de desarrollo de software utilizadas en la implementación de sistemas de gestión de información, permitió la elección de las herramientas, tecnologías y metodología de desarrollo de software adecuada para desarrollar el sistema. Se seleccionó como lenguaje de programación PHP en su versión 5.4.9, siendo Symfony en su versión 2.1.8 el *framework* php y NetBeans el IDE de desarrollo en su versión 7.3. Además como como herramienta de modelado Visual Paradigm en su versión 8.0 la cual usa el lenguaje modelado UML en su versión 2.0, como sistema gestor de bases de datos PostgresSQL en su versión 9.1 y pgAdmin en su versión 1.16.1 para la administración, como servidor web Apache en su versión 5.4.3, con el propósito de recrear una interfaz fácil y sencilla de usar se decidió utilizar Bootstrap en su versión 3.0.1 y como metodología de desarrollo de software XP.

CAPÍTULO II: DISEÑO DEL MÓDULO DE CONTABILIDAD PARA EL SISTEMA DE GESTIÓN DE INFORMACIÓN DE LA ASIGNACIÓN, DISTRIBUCIÓN Y CONSUMO DE COMBUSTIBLE EN LA UCI

2.1. Introducción

En el capítulo se exponen las principales características del sistema a desarrollar, mediante el modelado del negocio, los requisitos funcionales y no funcionales así como las historias de usuario relacionada a los requerimientos. En correspondencia con la etapa de planificación del sistema se puntualizan el plan de entregas y el de iteraciones. Se describe la arquitectura por la cual se regirá la construcción de la solución, los patrones de diseño que serán utilizados para dar solución al problema que dio origen a la presente investigación.

2.2. Características del entorno

Actualmente para la especialista de la caja le es engorroso consultar todas las asignaciones mensuales de combustible por tarjeta y en el submayor así como hacer el proceso de selección de las tarjetas por los tipos de combustible, ralentizando el proceso de distribución de combustible por cantidades que posean cada tarjeta y la documentación que este lleva consigo. No ser registran los datos de las liquidaciones en ningún sistema para tener un control del consumo que se ha llevado a cabo por cada tarjeta asignada a un vehículo, lugar donde se habilitó, la cantidad habilitada y el día que lo realizo, datos serían imprescindibles para posteriores análisis referentes al consumo de combustible de cualquier vehículo en todo momento.

La documentación referente a la carga efectuada según la tabla de carga por tarjeta se realiza de forma lenta e ineficiente, no existe manera de notificar cuando el especialista realice la carga para cada tarjeta magnética hasta que no se elaboré el modelo tabla de cargue para todas las tarjetas, dicho documento tiene que ser aprobado por todos los factores correspondientes generando retrasos en el flujo de información y retrasos en la entrega del combustible en el próximo mes debido a que el departamento de finanzas tiene que recibir de la caja todos los reportes actualizados en cuanto a consumo y habilitación por vehículo, este proceso ralentiza el cargue de combustible debido al lento flujo de la información necesaria que el mismo lleva consigo. Además no se cuenta con un registro para llevar la información contable por

cada vehículo del consumo que realiza de forma semanal y mensual, al no tener conocimiento del consumo de manera inmediata es imposible estimar el monto que alcanza el gasto por vehículo.

2.3. Modelado de proceso de negocio

El modelado de procesos del negocio facilita entender y organizar algunos procesos que están conformados de múltiples actividades, personas y departamentos en una organización. Su uso trae como beneficios conocer de qué trata el proceso de principio a fin, revelar las fallas y problemas, así como generar soluciones a los problemas detectados. Describe de forma clara y concisa lo que el negocio ofrece a sus clientes y sirve de referencia para el desarrollo de las ventajas competitivas de la institución. La fase de modelado de negocio es la primera y más importante del ciclo de vida correspondiente al desarrollo de software (COLECTIVO DE AUTORES, 2009).

2.3.1. Descripción de los Procesos del negocio

Los modelos de procesos son diagramas de flujo detallados con suficiente información, para poder estudiar el proceso y simularlo como se muestra a continuación:

Recogida y entrega de tarjeta magnética

- **Recogida de tarjetas.** La especialista de la caja se encarga de hacer una consulta de la asignación mensual de combustible a partir de los registros de consumo que posee en caja por cada tarjeta y los que posee la Dirección de Transporte a partir de las liquidaciones mensuales de combustible por cada vehículo. Además, realiza una consulta en el submayor de consumo de combustible que poseen en los registros de la caja. Luego de hacer esto, se encargan de hacer la selección de las tarjetas por los tipos de combustible, los cuales pueden ser, diésel, gasolina especial y gasolina regular. Se elabora otro documento por parte de la caja, donde la cajera hace la distribución de combustible por cantidades que posee cada tarjeta, este documento se hace entrega por parte de la cajera al grupo de finanzas, con el fin de revisar que la asignación que hizo la Dirección de Transporte y que fue aprobada por varios factores sea distribuida a partir de la existencia que posee cada tarjeta, en caso de tener existencia, cargar por parte de finanzas entonces la diferencia a cada tarjeta con el objetivo de ponerle lo asignado por la Dirección de

Transporte. Posteriormente finanza revisa la carga realizada por el especialista de este grupo, la cajera se encarga de recoger las tarjetas en soporte físico y de hacer entrega al responsable de cada vehículo del centro que le fue asignado combustible.

- **Entrega de tarjetas magnéticas.** La caja de la UCI a través de un especialista se encarga de hacer entrega a inicios de cada mes de la tarjeta que tiene asignada cada vehículo. Para esto el chofer del vehículo debe presentarse en la caja y firmar varios documentos que la cajera elabora y se procede a entregar la tarjeta con el pin que posee, el pin es el código de acceso al uso de la tarjeta. A continuación se representa el proceso de recogida y entrega de tarjetas magnéticas.

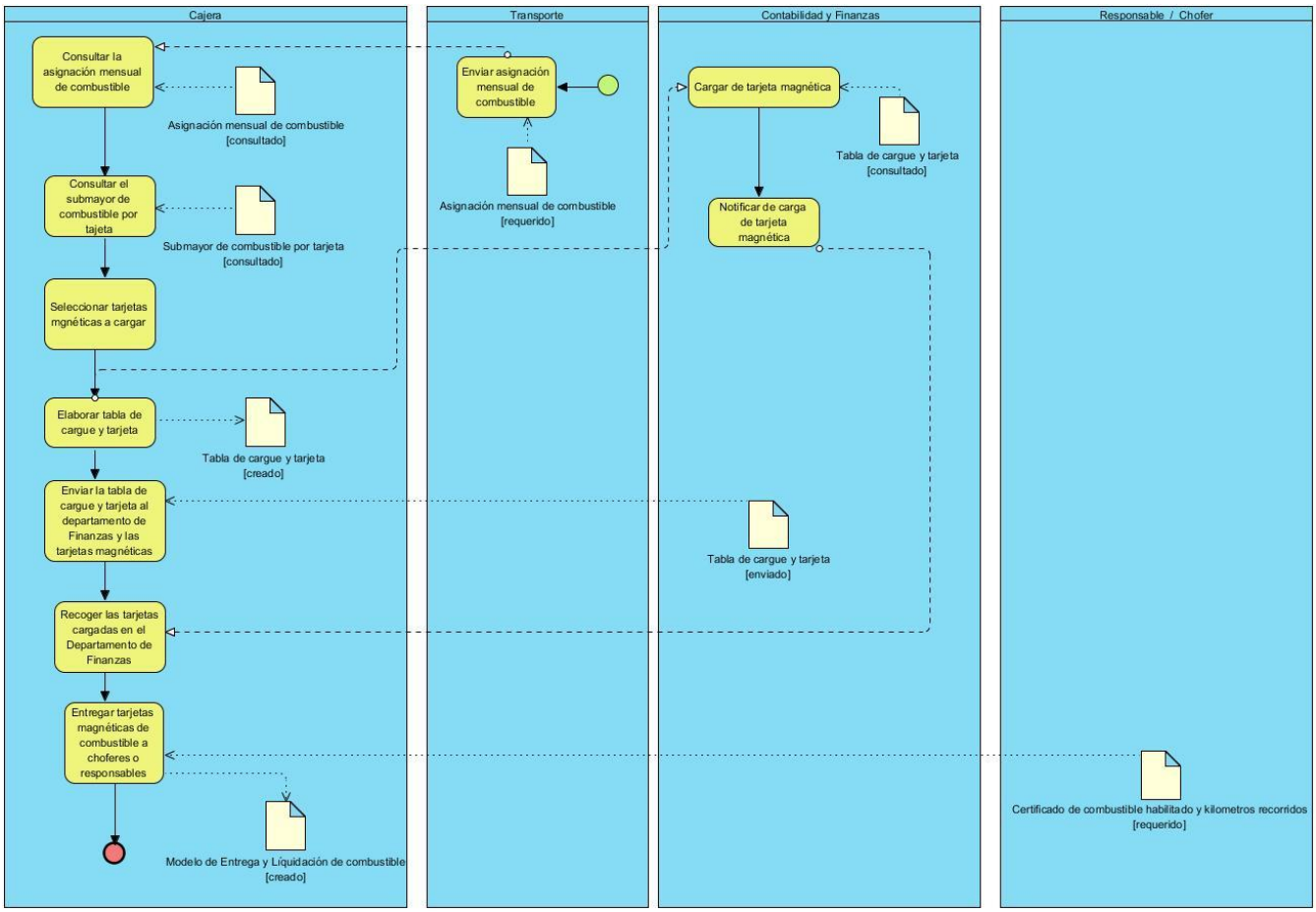


Figura 1: Proceso de recogida y entrega de tarjetas magnéticas.

Proceso de cargue de combustible en tarjetas magnéticas

- Este proceso se lleva a cabo, cuando el grupo de finanzas recibe de la Dirección de Transporte, luego de haberse aprobados por diferentes factores, la tabla de cargue de tarjetas. Esta tabla, posee un desglose por área de cada vehículo que tiene asignado y a su vez la cantidad de combustible que se le debe poner a este vehículo en la tarjeta que tenga asignado para comenzar a ser usada.

  Luego de tenerse la tabla mencionada en el párrafo anterior, un especialista de este grupo se encarga de hacer la carga a cada tarjeta magnética. Después de cargada cada tarjeta, se notifica a la caja de la UCI mediante la tabla de cargue realizada a cada tarjeta magnética, además de hacerle entrega a otro especialista de la caja de las tarjetas magnéticas en soporte duro y los *ticket* de notificación de cargue, los *ticket*, no son más que el reporte que muestra el sistema de recarga de combustible de la empresa Financiera CIMEX S.A, los cuales notifican la carga recibida por cada tarjeta en valor contable y valor en consumo en litros. A continuación se representa el proceso de carga de tarjetas magnéticas.

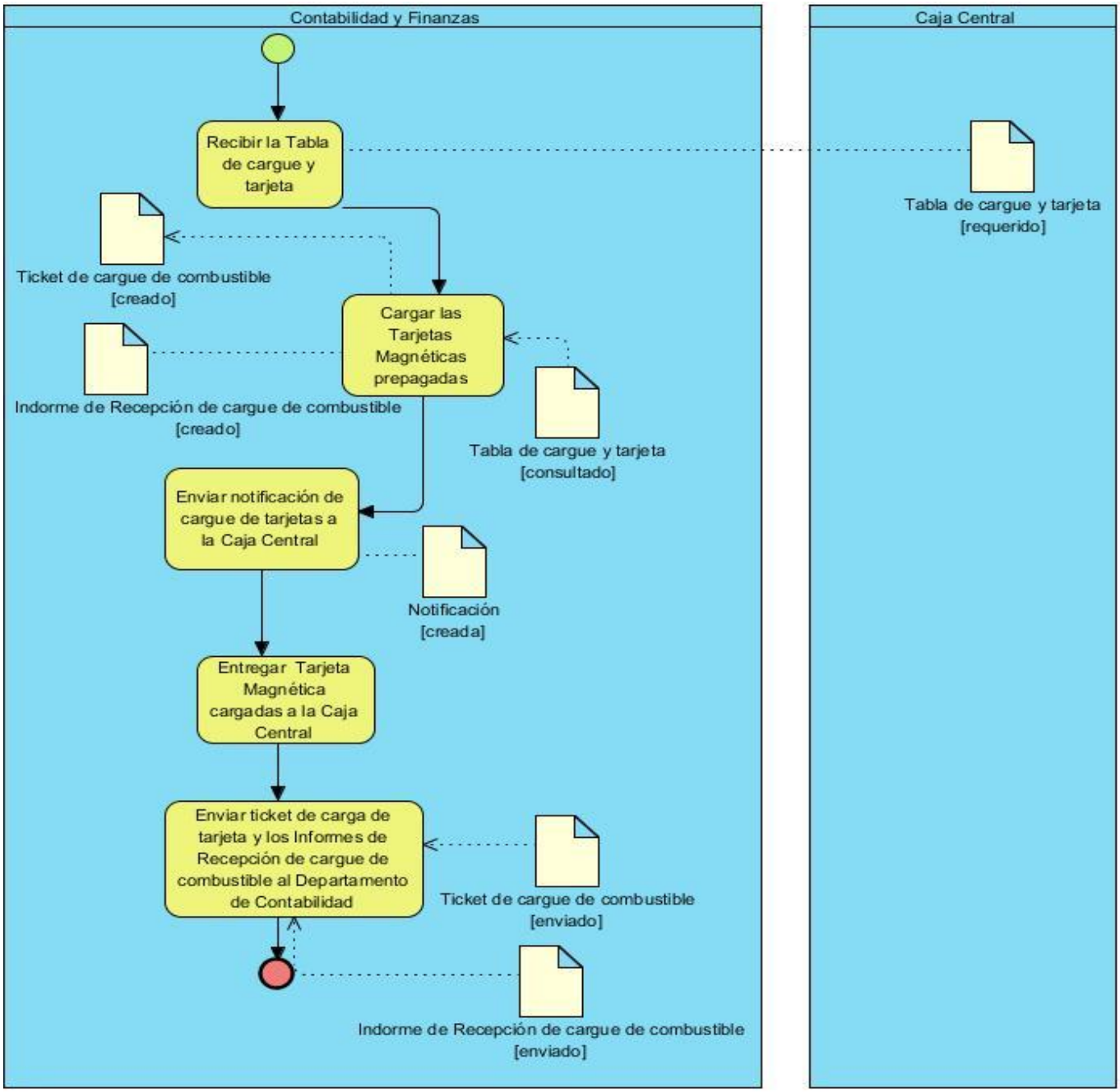


Figura 2: Proceso de carga de tarjetas.

Liquidación de combustible en caja

Este proceso se lleva a cabo en la Caja de la UCI tres veces en el mes. Cada responsable de vehículo debe ir el primer viernes del mes a entregar en la caja los *tickets* de consumo que haya llevado a cabo, luego lo hacer el tercer viernes del mes y luego cualquier día de la última semana del mes, ya esta última vez, entrega la tarjeta para ser resguardada por la caja. Estas entregas de *tickets*, permiten que la cajera pueda tener conocimiento del consumo realizado por cada tarjeta asignada a un vehículo, proceso el cual se debe informatizar, pues se debe tener registrados los consumos que se hacen, donde se habilitó el vehículo, que cantidad se habilitó y que día lo hizo. De este proceso se debe realizar un reporte y enviarlo al departamento de finanzas para ser usado como guía en el cargue de las tarjetas magnéticas del próximo mes. A continuación se representa el proceso de liquidación de combustible en caja.

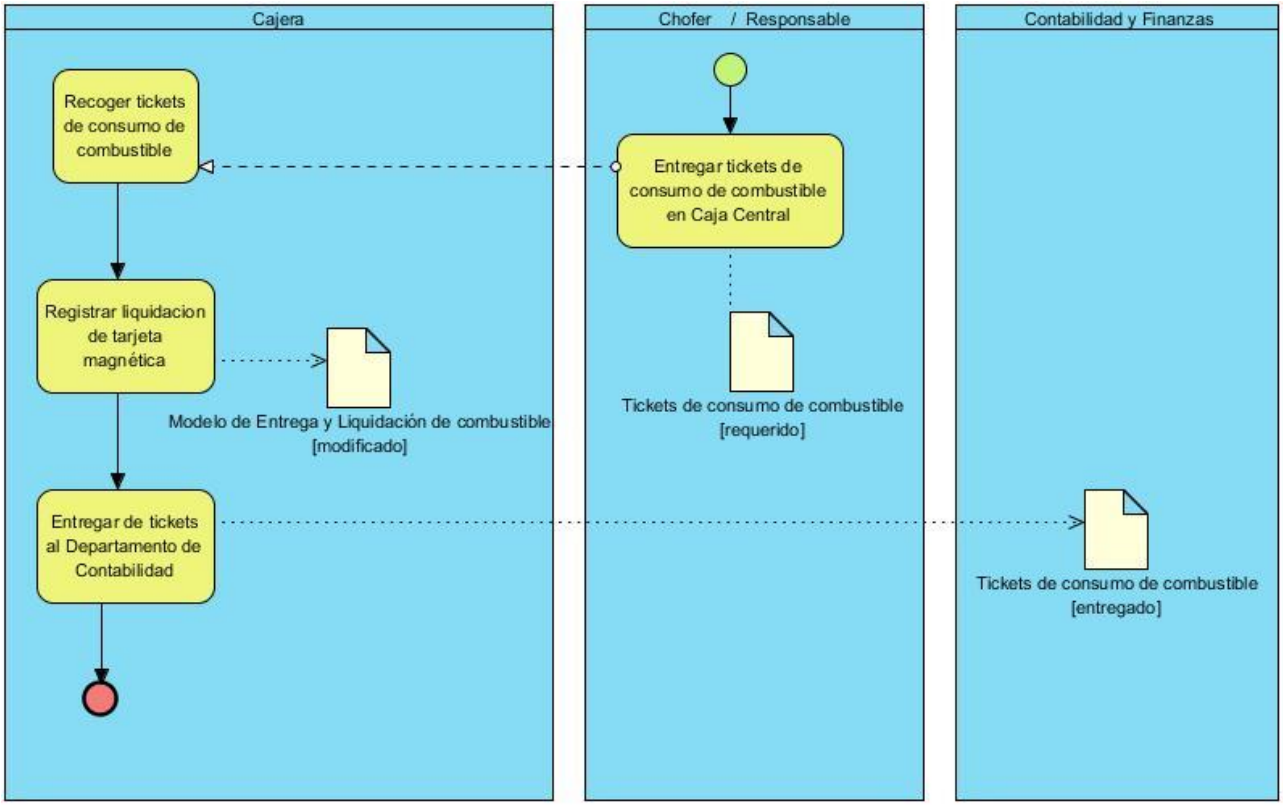


Figura 3: Proceso de liquidación de combustible en caja.

2.4. Requisitos del sistema

Los requerimientos para el módulo contable del sistema de distribución y consumo de combustible de la UCI, son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos se clasifican en requisitos funcionales y no funcionales. Los funcionales declaran los servicios que debe brindar el sistema, como funcionar ante una entrada o situación en particular. Los no funcionales son las restricciones de los servicios o funciones ofrecidas por el sistema.

El flujo de trabajo de requerimientos es de vital importancia para el desarrollo de un software, ya que su propósito fundamental es guiar el desarrollo hacia el sistema correcto. Las técnicas seleccionadas para el levantamiento de los requisitos del sistema fueron entrevistas y tormenta de ideas.

A continuación se mencionan los requisitos identificados:

2.4.1. Listado de requisitos funcionales

Los requisitos funcionales (RF) que describen las funcionalidades que el sistema debe cumplir son:

Tabla 1: Requisitos funcionales.

NO. RF	DESCRIPCIÓN DEL REQUISITO	PRIORIDAD
Grupo de Finanzas		
RF 1	Adicionar carga de combustibles en dinero y en litros	Alta
RF 2	Modificar carga de combustibles en dinero y en litros	Alta
RF 3	Eliminar carga de combustibles en dinero y en litros	Alta
RF 4	Mostrar carga de combustible en dinero y en litros	Alta
RF 5	Adicionar tarjeta	Alta
RF 6	Modificar tarjeta	Alta
RF 7	Eliminar tarjeta	Alta
RF 8	Mostrar tarjeta	Alta
RF 9	Buscar tarjeta	Media
RF 10	Buscar carga por tarjeta	Media

RF 11	Realizar reporte de carga de combustible por mes y por tipo de combustible	Alta
RF 12	Exportar reporte de carga de combustible por mes y por tipo de combustible a pdf	Alta
Caja		
RF 13	Adicionar entrega de combustible por tarjeta	Alta
RF 14	Modificar entrega de combustible por tarjeta	Alta
RF 15	Eliminar entrega de combustible por tarjeta	Alta
RF 16	Mostar entrega de combustible por tarjeta	Alta
RF 17	Adicionar liquidación de combustible por tarjeta	Alta
RF 18	Modificar liquidación de combustible por tarjeta	Alta
RF 19	Eliminar liquidación de combustible por tarjeta	Alta
RF 20	Mostrar liquidación de combustible por tarjeta	Alta
RF 21	Buscar entrega por tarjeta	Media
RF 22	Buscar liquidación por tarjeta	Media
RF 23	Crear modelo de entrega y liquidación	Alta
RF 24	Exportar el modelo de entrega y liquidación a pdf	Alta

2.4.2. Listado de requisitos no funcionales

Luego de analizar las condiciones que resultan apropiadas para el funcionamiento de la propuesta de solución planteada, se identificaron los requerimientos no funcionales (RnF) que no son más que aquellas características que el sistema debe cumplir para garantizar el despliegue y soporte del mismo (Pressman, 2005) .

-Usabilidad:

- **RnF 1.** El módulo debe poseer una distribución de la información simple y que posibilite a los usuarios llegar al contenido que desea en un corto tiempo. Siempre que no fuerce la estructura del sistema, ninguna página debe encontrarse a más de tres clic de la página de inicio.
- **RnF 2.** El módulo debe permitirle a los usuarios con escasos conocimientos de informática poder interactuar con él.

- **RnF 3.** Debe poseer una interfaz intuitiva y fácil de navegar.

-Disponibilidad:

- **RnF 4:** El módulo debe estar disponible las 24 horas del día los 7 días de la semana.

-Confiabilidad:

- **RnF 5:** Los reportes que se obtendrán serán 100% reales y precisos.
- **RnF 6:** La información no podrá ser modificada por ningún usuario no autorizado, protegiendo así la integridad de los datos.

-Fiabilidad:

- **RnF 7:** El módulo debe mostrarle al usuario un mensaje indicándole que ha ocurrido un fallo en la operación que se realice o de lo contrario que se realizó satisfactoriamente.

-Soporte:

- **RnF 8:** El módulo debe dar la posibilidad de ser mejorado, así como de incorporarle nuevas funcionalidades, en caso de ser necesarias.

-Hardware:

Servidores de aplicación y base de datos:

- **RnF 9:** Para garantizar un funcionamiento óptimo del sistema, el servidor donde estará desplegado el mismo deberá ser Corei3, con una velocidad del procesador de 3.00 GHz, 3 GB de RAM y 160 GB de disco duro.

PC cliente:

- **RnF 10:** Para el cliente como requerimientos mínimos: Procesador Pentium III a 2.8 GHz con 512 Mb de memoria RAM y una tarjeta de red o en otro caso un cliente ligero.

-Software:

- **RnF 11:** La aplicación puede ser accedida por parte del cliente desde cualquier navegador web y se requiere un Sistema operativo Nova 3.0 o cualquier otra distribución de Linux y Microsoft

Windows XP, 7,8 o superior. Por parte del servidor necesariamente tiene que ser una distribución de Linux.

Servidores de Aplicación:

- **RnF 12:** El servidor de aplicaciones debe tener el sistema operativo CentOS 7 o superior.
- **RnF 13:** El servidor de aplicaciones debe tener como servidor web Nginx en su versión 1.0.15 o superior.

-Seguridad

- **RnF 14:** Se garantizará la confidencialidad, integridad y disponibilidad de la información mediante mecanismos de control de acceso.
- **RnF 15:** El módulo le permitirá al usuario las funcionales en dependencia del rol que cumpla.
- **RnF 16:** Se utilizará el protocolo HTTPS para la comunicación entre el cliente y el servidor en los procesos de envío de datos entrados por el usuario en la autenticación y en las tareas administrativas y de gestión de contenidos.

2.5. Historias de Usuario (HU)

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software. Estas describen brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. Cada historia de usuario debe ser lo suficientemente comprensible y delimitada para que los programadores puedan implementarla.

Son usadas para estimar el tiempo y plan de lanzamiento, y dirigen la creación de las pruebas de aceptación. Se caracterizan por ser independientes una de otras, negociables, valoradas por los clientes o usuarios, estimables, pequeñas y verificables.

A continuación se definen una de las HU de la solución propuesta.

Tabla 2: Historia de usuario “Gestionar la carga de combustibles en dinero y en litros”.

Historia de Usuario

Número: HU_1	**Nombre Historia de Usuario:** Gestionar carga de combustibles en dinero y en litros
Modificación de Historia de Usuario Número: *1*	
Usuario: Iraida Rondón Sosa	**Iteración Asignada:** 1
Prioridad en Negocio: Alta	**Puntos Estimados:** 1 semana
Riesgo en Desarrollo: Alto	**Puntos Reales:** 1 semana
Descripción: Permite cargar, modificar, eliminar y mostrar el monto de combustible en dinero y la cantidad de litros por cada tarjeta	
Observaciones: 1- Se debe estar autenticado en el sistema 2- Debe estar por niveles de acceso 3- Todos los campos son obligatorios	
Prototipo de interfaz:	

2.6. Planificación

Conformadas las historias de usuario pasamos a realizar un análisis del ámbito de desarrollo para lo cual se tiene en cuenta un plan de entregas y de iteraciones.

2.6.1. Plan de entregas

En esta fase el cliente establece la prioridad de cada historia de usuario, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Se propone el siguiente plan de entregas para la solución propuesta:

Tabla 3: Plan de entrega del módulo.

Entregables	Iteración	Fin de la iteración
Gestionar carga de combustibles en dinero y en litros	1	Febrero 2015
Gestionar tarjeta	1	Febrero 2015
Buscar tarjeta	1	Febrero 2015
Buscar cargar por tarjeta	1	Febrero 2015
Gestionar entrega de combustible por tarjeta	2	Marzo 2015
Gestionar liquidación de combustible por tarjeta	2	Marzo 2015
Buscar entrega por tarjeta	2	Marzo 2015
Buscar liquidación por tarjeta	2	Marzo 2015
Realizar reporte de carga de combustible por mes y por tipo de combustible	3	Abril 2015
Crear modelo de entrega y liquidación	3	Abril 2015
Exportar reporte de carga de combustible por mes y por tipo de combustible a pdf	3	Abril 2015
Exportar el modelo de entrega y liquidación a pdf	3	Abril 2015

2.6.2. Plan de iteraciones

El ciclo de desarrollo de software guiado por XP se caracteriza por ser iterativo e incremental, por lo que se realizan varias iteraciones sobre el sistema antes de su fase de culminación. Las historias de usuarios son desarrolladas y probadas en un ciclo de iteración de acuerdo al orden establecido.

El módulo de contabilidad para el sistema de gestión de información de la asignación, distribución y consumo de combustible en la se realizará en 3 iteraciones.

Tabla 4: Plan de Iteraciones.

Iteración	Historia de Usuario	Semana estimadas
1	Gestionar carga de combustibles en dinero y en litros	4
	Gestionar tarjeta	
	Buscar tarjeta	
	Buscar cargar por tarjeta	
2	Gestionar entrega de combustible por tarjeta	4
	Gestionar liquidación de combustible por tarjeta	
	Buscar entrega por tarjeta	
	Buscar liquidación por tarjeta	
3	Realizar reporte de carga de combustible por mes y por tipo de combustible	4
	Crear modelo de entrega y liquidación	
	Exportar reporte de carga de combustible por mes y por tipo de combustible a pdf	
	Exportar el modelo de entrega y liquidación a pdf	

2.7. Diseño

El papel del diseño en el ciclo de vida del software es adquirir conocimiento de su funcionamiento, constituye el punto de partida para las actividades de implementación, dando soporte a los requisitos funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables y sistemas operativos, que debe poseer la aplicación.

2.7.1. Descripción de la arquitectura

Un patrón de arquitectura de software describe un problema particular y recurrente del diseño, que surge en un contexto específico, y presenta un esquema genérico y probado de su solución. Son soluciones que están apoyadas en la experiencia de las compañías y en general, de los desarrolladores.

Para el desarrollo del módulo del sistema de distribución y consumo de combustible de la UCI, se utilizará el patrón de arquitectura de software Modelo Vista Controlador (en inglés abreviado MVC) usando el *framework* Symfony2. Este patrón separa los datos y la lógica del negocio de una aplicación de la interfaz de usuarios y el módulo encargado de gestionar los eventos y las comunicaciones. Permite además separar cada una de las lógicas del módulo en archivos independientes permitiendo hacerlo flexible y sencillo de mantener. En la siguiente figura se representa el funcionamiento de la arquitectura Modelo Vista Controlador implementada por el *framework* Symfony2.

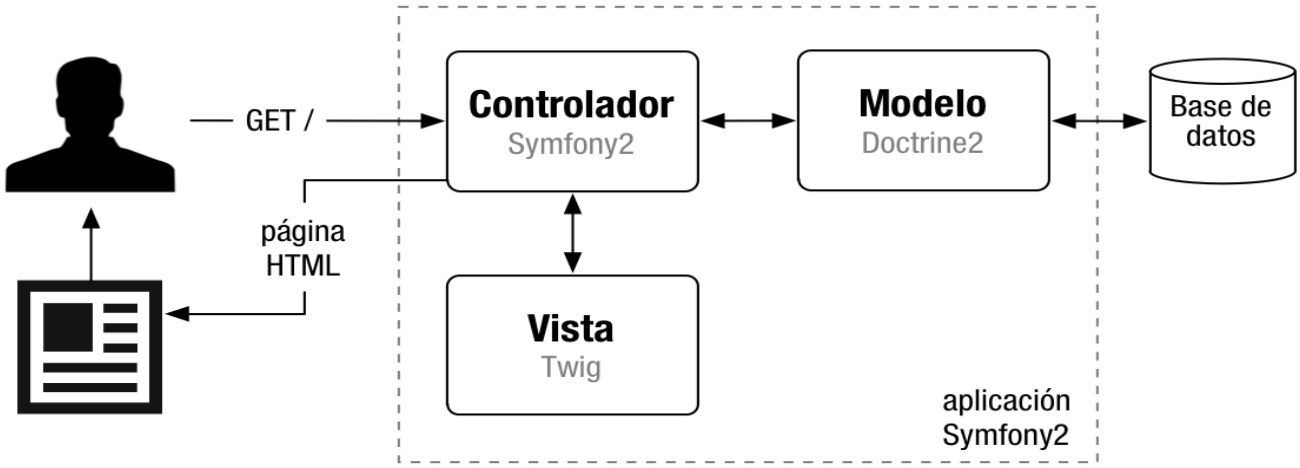


Figura 4: Arquitectura de software MVC implementada por el *framework* Symfony2.

Controlador: El controlador soporta el flujo de información del módulo de control de acceso físico, interactuando con el modelo y la vista. Es el componente que da soporte a las funcionalidades de la capa de negocio y que se encuentra relacionada con la fuente de datos. La principal función de esta capa es realizar una implementación de las funcionalidades definidas en las interfaces de la capa de negocio y al mismo tiempo trabajar directamente con la fuente de datos.

Vista: Es el objeto que maneja la presentación visual de los datos representados por el modelo. Genera una representación visual del modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio modelo.

Modelo: El modelo contiene la información básica del módulo de control de acceso físico. Esto incluye los datos y reglas de validación, así como acceso a datos y lógica de agregación.

En el módulo desarrollado esta arquitectura se evidencia en el siguiente esquema que representa las liquidaciones de tarjetas.

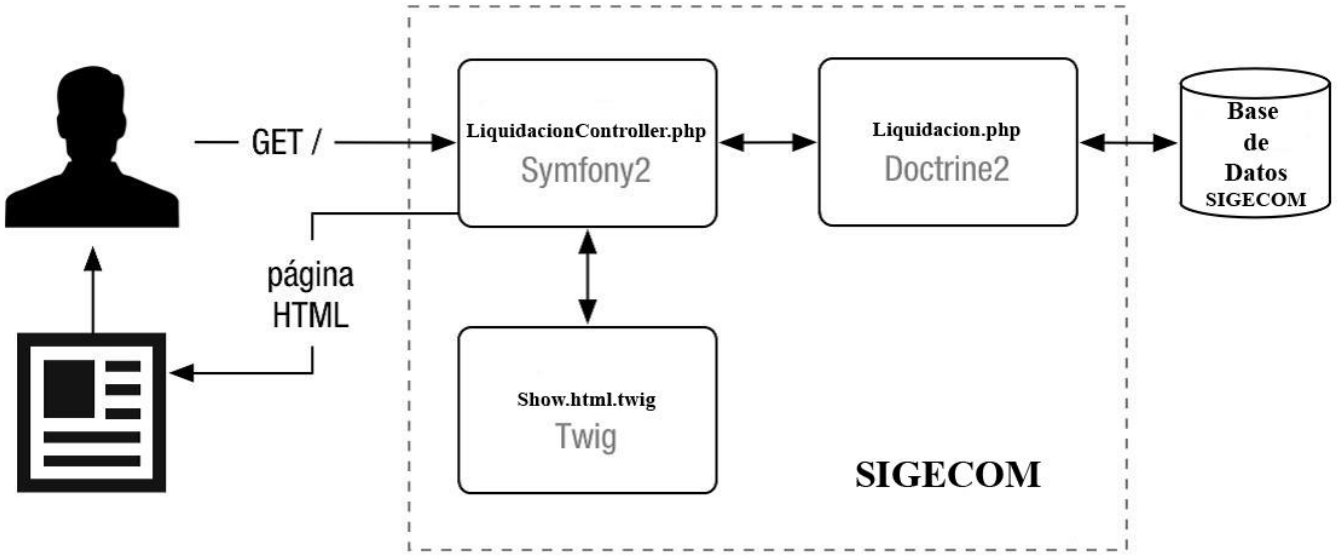


Figura 5: Esquema del funcionamiento de liquidación de tarjetas.

LiquidacionController.php: representa la información con la que trabaja la aplicación y al mismo tiempo trabaja directamente con la fuente de datos Liquidacion.php, encargándose de hacer las peticiones a la base de datos mediante el ORM de Doctrine para enviárselos a Show.html.twig.

Show.html.twig: maneja la presentación visual de los datos del modelo Liquicacion.php generando una representación de los mismos, procesando las interacciones del usuario y ejecutando los cambios en el modelo y la vista.

Liquidacion.php: contiene la información básica del módulo de control de acceso físico referente a las liquidaciones, sus reglas de validación y su lógica. Contiene los elementos para la representación de los datos y la información de los elementos que conformarán los reportes.

2.7.2. Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular, y brindan una solución ya probada y

documentada a problemas de desarrollo de software que están sujetos a contextos similares. Los patrones son una pareja problema/solución con un nombre, que codifican buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades (ALLEN, 1992). Entre los patrones que se utilizarán se encuentran los patrones generales de software para la asignación de responsabilidades (GRASP por sus siglas en inglés *General Responsibility Assignment Software Patterns, Patrones Generales de software para asignación de responsabilidades*) y los patrones GoF (por sus siglas en inglés *Gang Of Four*).

Los Patrones GRASP tienen una importante utilidad en el diseño realizado. Describen los principios fundamentales para asignar responsabilidades a los objetos (POLO, 2011). Los utilizados en el diseño del sistema son los siguientes:

- **Experto**: este patrón se evidencia en la clase LiquidacionController, la cual controla el dominio de la información referente a las liquidaciones de combustible de cada tarjeta.

- **Bajo Acoplamiento**: este patrón se evidencian en las clases Entrega y TransParqueEquipo, las cuales tienen una débil dependencia, y cualquier cambio en una de ellas no afectaría a la otra.

- **Controlador**: este patrón se evidencia en la clase EntregaController, la cual posee responsabilidades específicas (Larmam, 2008).

Patrones de diseño GOF

Los patrones GoF describen soluciones simples y elegantes a problemas específicos en el diseño de software orientado a objetos y se agrupan en tres grandes categorías: creacionales, estructurales y de comportamiento (PRESSMAN, 2010). Los utilizados en el diseño del sistema son los siguientes:

- **Decorador:** Symfony2 presenta el denominado archivo base.html.twig o también conocido como plantilla global, en la que convergen todos los elementos comunes a cada una de las páginas del sistema en construcción. Este fichero se complementa con las plantillas, decorándolas y obteniéndose la interfaz final que será mostrada al usuario.

- ***Command* (Orden):** en el *framework* Symfony se utiliza este patrón, el cual está representado por las acciones, por tanto cada acción, encapsula una petición en un objeto. En el sistema se evidencia el uso de este patrón al realizar la acción de exportar a formato .pdf, a través del cual se construye el documento pdf y se exporta.

- ***Front controller* (Controlador frontal):** es el único punto de entrada a la aplicación. Carga la configuración y determina la acción a ejecutarse. Se encarga de despachar las peticiones, lo que implica algo más que detectar la acción que se ejecuta. De hecho, ejecuta el código común a todas las acciones, incluyendo:

- Carga la clase de configuración del proyecto y las librerías de Symfony.
- Crea la configuración de la aplicación y el contexto de Symfony.
- Carga e inicializa las clases del núcleo del *framework*.
- Carga la configuración.
- Decodifica la URL de la petición para determinar la acción a ejecutar y los parámetros de la petición.
- Si la acción no existe, redireccionará a la acción del error 404.
- Activa los filtros (por ejemplo, si la petición necesita autenticación).
- Ejecuta los filtros, primera pasada.
- Ejecuta la acción y produce la vista.
- Ejecuta los filtros, segunda pasada.
- Muestra la respuesta.

2.7.3. Modelo físico de la base de datos

El diseño de la base de datos representa el tratamiento de la información con carácter persistente dentro del sistema. A continuación se muestra el modelo físico de la base de datos.

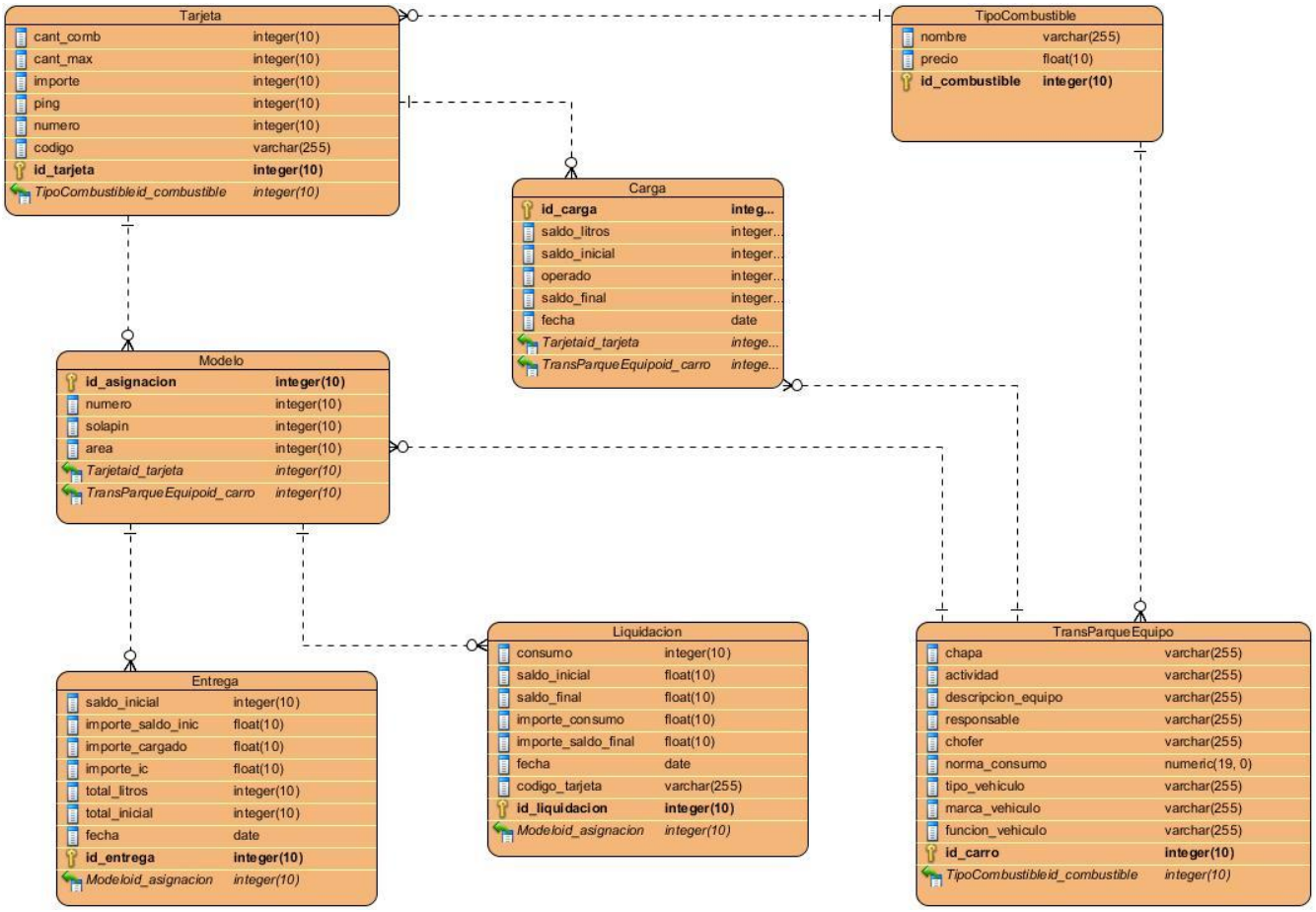


Figura 6: Diagrama del modelo entidad relación de la base de datos.

2.7.4. Tarjetas CRC

Las tarjetas CRC son herramientas usadas por la metodología XP para el diseño de software. A continuación se muestran algunos ejemplos de las empleadas en el desarrollo del módulo.

Tabla 5: Tarjeta CRC de la entidad Tarjeta.

Nombre de la clase: Tarjeta	
Responsabilidades	Colaboradores
1. getID() 2. getCantComb()	1. Modelo 2. TipoCombustible

3. setCantComb($cantComb) 4. getImporte() 5. setImporte($importe) 6. getPing() 7. setPing($ping) 8. getCantMax() 9. setCantMax($cantMax) 10. getNumero() 11. setNumero($numero) 12. addModelo($modelos) 13. removeModelo($modelo) 14. getModelo() 15. getTipoCombustible() 16. setTipoCombustible($tComb)	

Tabla 6: Tarjeta CRC de la entidad Modelo.

Nombre de la clase: Modelo	
Responsabilidades	Colaboradores
1. getID() 2. getNumero() 3. setNumero($numero) 4. getSolapin() 5. setSolapin($solapin) 6. getArea() 7. setArea($area) 8. getResponsable()	1. Tarjeta 2. Liquidacion 3. Entrega 4. TransParqueEquipo

9. setResponsable($responsable) 10. getTarjeta() 11. setTarjeta($tarjeta) 12. getTransParqueEquipo() 13. setTransParqueEquipo($tPE) 14. getFecha() 15. setFecha($fecha) 16. addLiquidaciones($liquidaciones) 17. removeLiquidaciones($liquidaciones) 18. getLiquidaciones() 19. addEntrega($entregas) 20. revomeEntrega($entregas) 21. getEntregas()	

Tabla 7: Tarjeta CRC de la entidad Liquidacion.

Nombre de la clase: Liquidacion	
Responsabilidades	Colaboradores
1. getID() 2. getSaldoInicial() 3. setSaldoInicial($saldoInicial) 4. getConsumo() 5. setConsumo($consumo) 6. getSaldoFinal() 7. setSaldoFinal($saldoFinal) 8. getImporteConsumo() 9. setImporteConsumo($impConsumo) 10. getImporteSaldoFinal() 11. setImporteSaldoFinal($ImpSalFin) 12. getFecha()	1. Modelo

13. setFecha($fecha) 14. getModelo() 15. setModelo($modelo)	

2.7.5. Diagrama de despliegue

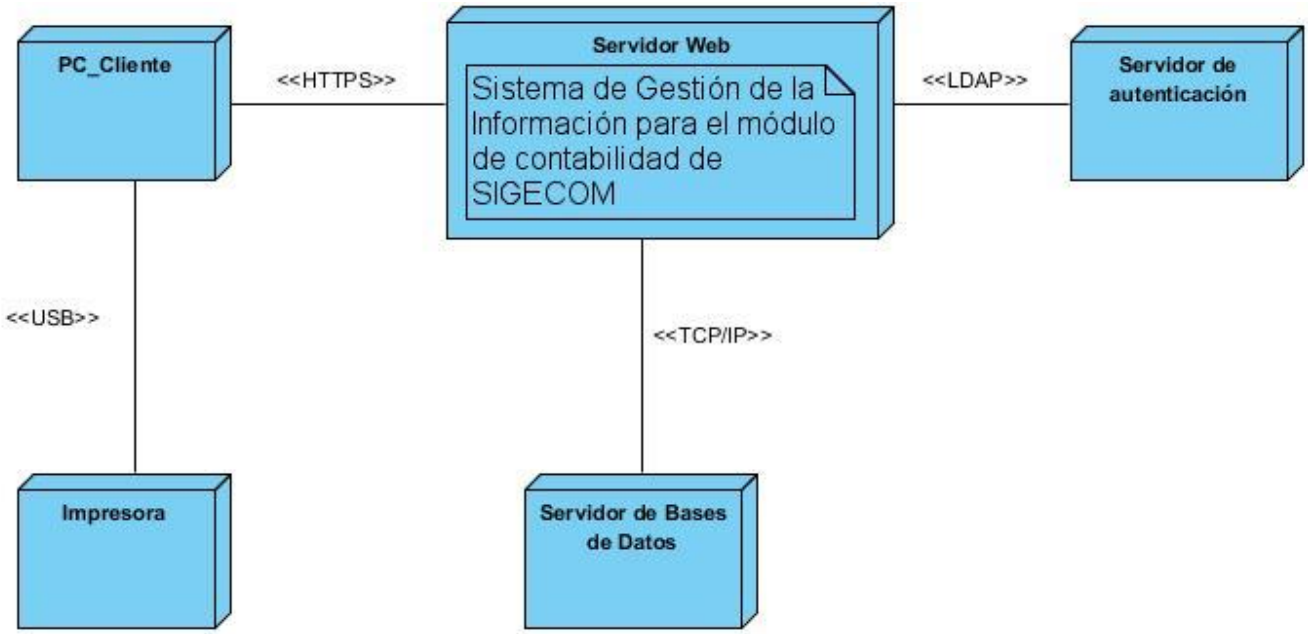


Figura 7: Diagrama de despliegue del módulo.

2.8. Conclusiones

Con el modelado de los procesos del negocio y la explicación detallada de los mismos fue posible identificar con mayor facilidad los requisitos funcionales del sistema, propiciando así la comprensión de su funcionamiento. Las historias de usuario fueron especificadas para cada uno de estos requisitos empleando un lenguaje acorde al cliente sin caer en un lenguaje técnico, detallando entre otros elementos el tiempo que requieren para su desarrollo y su prioridad, delimitando la duración de cada iteración. El uso de patrones y el empleo del marco de trabajo Symfony2, permitió obtener un sistema mantenible, escalable y flexible ante posibles cambios en la implementación de las funcionalidades.

CAPÍTULO III: IMPLEMENTACIÓN Y PRUEBAS DEL MÓDULO DE CONTABILIDAD PARA EL SISTEMA DE GESTIÓN DE INFORMACIÓN DE LA ASIGNACIÓN, DISTRIBUCIÓN Y CONSUMO DE COMBUSTIBLE EN LA UCI

3.1. Introducción

En el capítulo se presentan los resultados de la implementación y validación del módulo empleando diferentes técnicas definidas por la metodología de desarrollo de software utilizada para medir la calidad de los mismos. Se especifica el estándar de codificación empleado, se crea el diagrama de despliegue representando los recursos físicos necesarios para el despliegue de la aplicación, y se presentan las interfaces de pruebas creadas para comprobar el funcionamiento de la solución. A partir del código resultante y su funcionamiento se ejecutan las pruebas unitarias y de aceptación del módulo.

3.2. Estándar de codificación

Los estándares de codificación son reglas de programación enfocadas a su estructura para facilitar la lectura, comprensión y mantenimiento del código. (ARIAS, 2012)

El uso de estos estándares repercute en la calidad del software de manera positiva, facilitando el trabajo del desarrollador y siendo una guía para el futuro mantenimiento de la aplicación.

Estructura

Las propiedades de las clases son declaradas antes que los métodos.

Nomenclatura

Los nombres de las clases y objetos deben comenzar con mayúsculas.

Los nombres de los parámetros, variables y atributos usan camellCasing, a excepción de nombres compuestos.

Los guiones bajos se utilizan para nombres compuestos de parámetros, variables y atributos, empezando siempre con minúscula.

Los nombres de las funciones usan el estándar camellCasing, comenzando siempre por minúscula.

Identación

La identación se realiza utilizando cuatro espacios, la tabulación no está permitida.

Agregar un único espacio después de cada delimitador coma.

Agregar un único espacio alrededor de los operadores.

Agregar un único espacio en blanco antes de los paréntesis de apertura, lo mismo sucede para el caso de las instrucciones if, else, while, for.

Agregar una línea en blanco antes y después de las declaraciones de clases y estructuras.

Dejar dos espacios por bloques de código.

Los paréntesis requieren una línea propia, con excepción de los que están después de las instrucciones if y los ciclos.

Variable y constantes

Los nombres de atributos solo de leerlos dan el conocimiento de a que se refieren.

Se utiliza camelCase en las declaraciones de variables.

3.3. Implementación

Luego de la fase de diseño y con los resultados obtenidos en la misma da comienzo la etapa de implementación. El modelo de implementación contiene el diagrama de componentes. Este describe la dependencia entre nodos físicos en los que funcionará el sistema, los cuales comienzan a desarrollarse en el flujo de trabajo de diseño y que se perfeccionan en la implementación (VÁZQUEZ y RODRÍGUEZ, 2013)

3.3.1. Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones, siendo los componentes piezas reutilizables de alto nivel a partir de las cuales se pueden construir los sistemas. Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean estos componentes de código fuente, binarios o ejecutables. Normalmente contienen componentes, interfaces y relaciones entre ellos, y como todos los diagramas también puede contener paquetes utilizados para agrupar elementos del modelo (Valdés, 2014). La siguiente figura muestra el diagrama de componente del sistema:

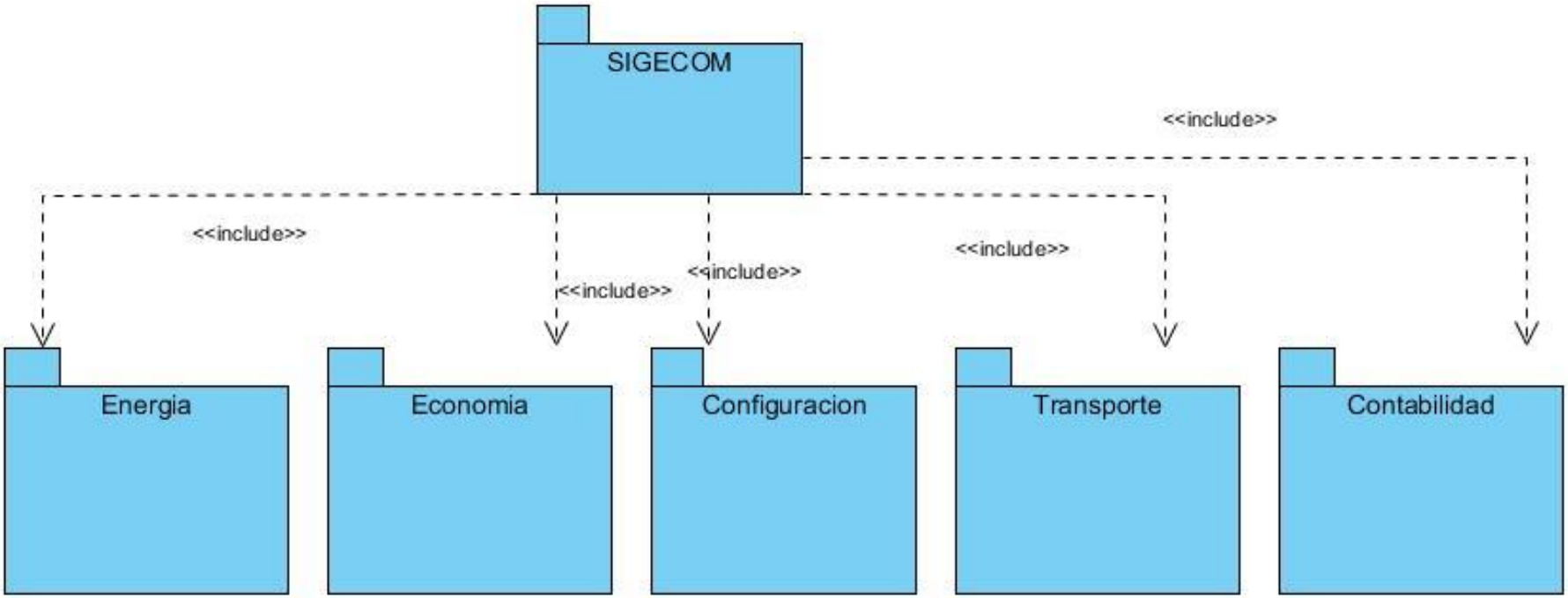


Figura 8: Diagrama de componentes del sistema.

La figura muestra una parte de los componentes del módulo desarrollado.

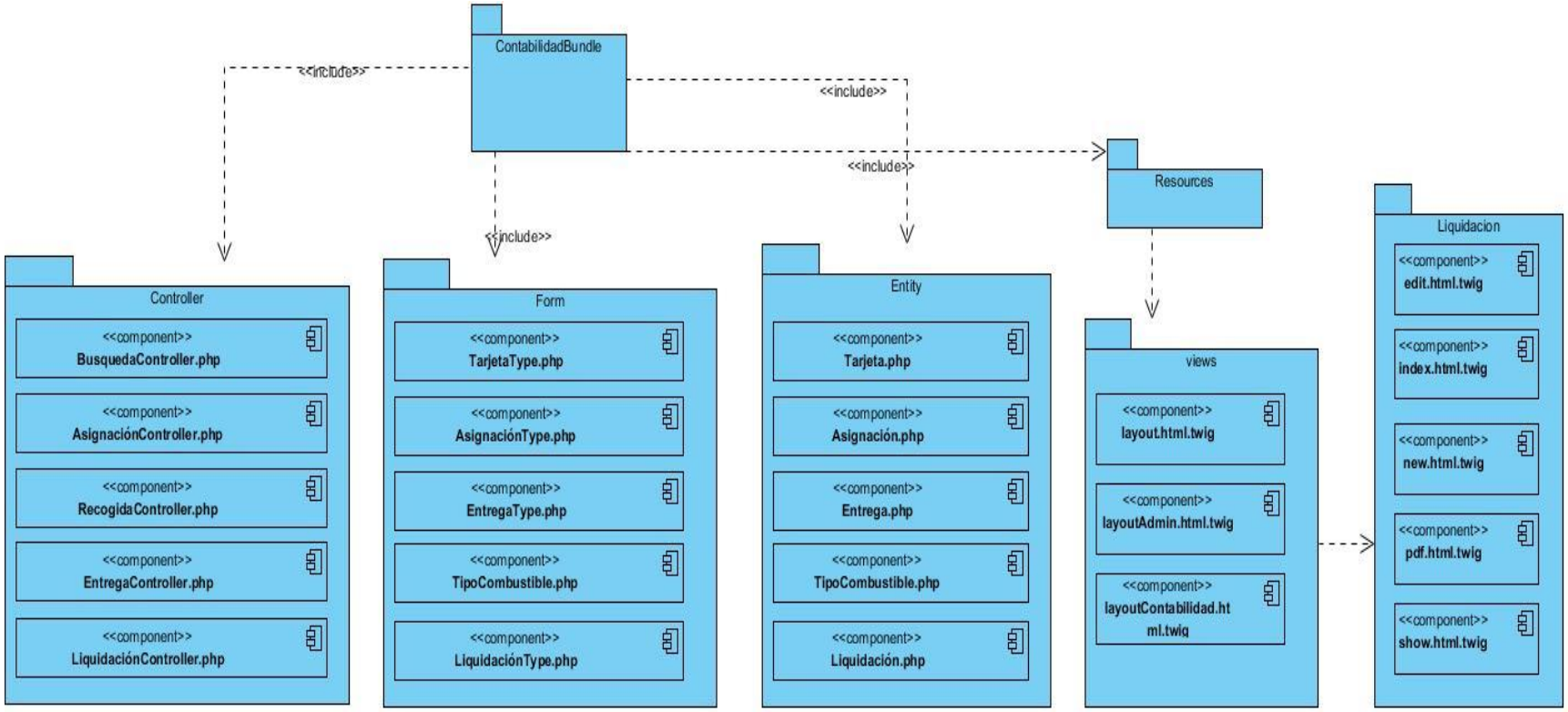


Figura 9: Diagrama de componente del módulo.

3.4. Pantallas principales de la aplicación

A continuación se muestra la pantalla de inicio de sesión de la aplicación donde se realiza la autenticación del usuario.

Figura 10: Pantalla de la vista autenticar usuario.

A continuación se muestra la interfaz principal de la aplicación correspondiente al rol de administrador donde se encuentra el menú principal con todos los módulos del sistema y la información detallada de cada uno.

Figura 11: Pantalla de la vista principal.

A continuación se muestra la interfaz correspondiente al usuario del departamento de contabilidad.

Figura 12: Pantalla del módulo de contabilidad.

A continuación se muestra la interfaz de liquidaciones por tarjetas, proceso realizado en la caja central.

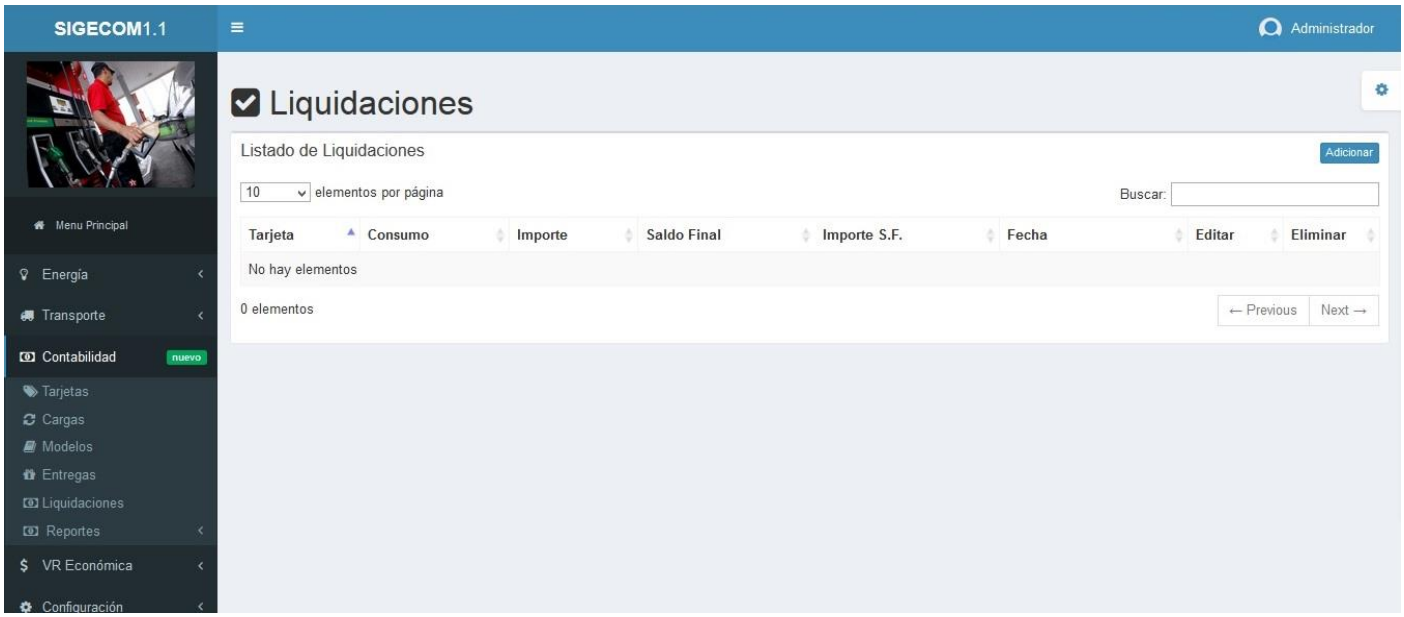


Figura 13: Pantalla de liquidaciones por tarjetas.

3.5. Pruebas

Las pruebas de software son el proceso de ejercitar el software con la intención de encontrar y corregir errores. Estas verifican que el software implemente una función específica de forma correcta y validan que las acciones que realiza respondan a los requisitos del cliente (PRESSMAN, 2005).

La programación extrema define dos tipos de pruebas las cuales son: pruebas unitarias y pruebas de aceptación. Las pruebas unitarias son desarrolladas por los programadores, dichas pruebas son las encargadas de verificar el código de forma automática mientras que las de aceptación son destinadas a evaluar si al final de la iteración se obtuvo la funcionalidad requerida, además de comprobar que dicha funcionalidad sea la esperada por el cliente.

3.5.1. Aceptación

Para la realización de las pruebas de aceptación fueron confeccionados casos de prueba, estos no son más que un conjunto de entradas, ejecuciones y resultados esperados. A continuación se muestran algunos casos de prueba realizados al módulo desarrollado.

Tabla 8: Descripción del caso de prueba número 01 a la HU_1.

Caso de prueba de aceptación	
Código: 01-01	**Historia de usuario:** Gestionar carga de combustibles en dinero y en litros.
Responsable de la prueba: Geinel Dorta Guevara	
Descripción: Se realiza la prueba a la funcionalidad Adicionar carga de combustible en dinero y en litros.	
Condiciones de ejecución: El usuario debe estar autenticado, tener el nivel de acceso y llenar todos los campos.	
Entrada/Pasos de ejecución: 1. Ir al menú Cargas dentro de módulo de Contabilidad. 2. Selecciona la opción Adicionar. 3. Introducir todos los datos necesarios para adicionar la carga y posteriormente guardarla.	
Resultado esperado: Mensaje que indique que se adicionó correctamente la carga.	
Evaluación de la prueba: Prueba satisfactoria.	

Tabla 9: Descripción del caso de prueba número 02 a la HU_1.

Caso de prueba de aceptación	
Código:01-02	**Historia de usuario:** Gestionar carga de combustibles en dinero y en litros.
Responsable de la prueba: Geinel Dorta Guevara	
Descripción: Se realiza la prueba a la funcionalidad Modificar carga de combustible en dinero y en litros.	

Condiciones de ejecución: El usuario debe estar autenticado, tener el nivel de acceso y llenar todos los campos.
Entrada/Pasos de ejecución: 1. Ir al menú Cargas dentro de módulo de Contabilidad. 2. Seleccionar cualquiera de las cargas e ir a la opción Editar. 3. Modificar los valores de la carga seleccionada.
Resultado esperado: Mensaje que indique que se modificó correctamente la carga.
Evaluación de la prueba: Prueba satisfactoria.

Los especialistas de los departamentos de Caja y Finanzas realizaron pruebas a las historias de usuario implementadas a través de los casos de prueba correspondientes. Al concluir cada iteración del desarrollo del módulo con el objetivo de guiar el desarrollo hacía los requisitos definidos según sus necesidades y corregir los errores detectados. Las pruebas efectuadas se realizaron en 2 iteraciones y una revisión final. En la primera iteración de pruebas de 10 requisitos funcionales implementados, se identificaron 8 no conformidades de las cuales fueron solucionadas 5 referentes a las validaciones y las 3 restantes referentes al negocio fueron arrastradas a la siguiente iteración por su relación con las próximas funcionalidades a implementar, siendo corregidas en dicha iteración. En la segunda iteración de 20 requisitos identificados se identificaron 3 no conformidades corregidas en su totalidad y en la revisión final se encontraron 0 no conformidades.

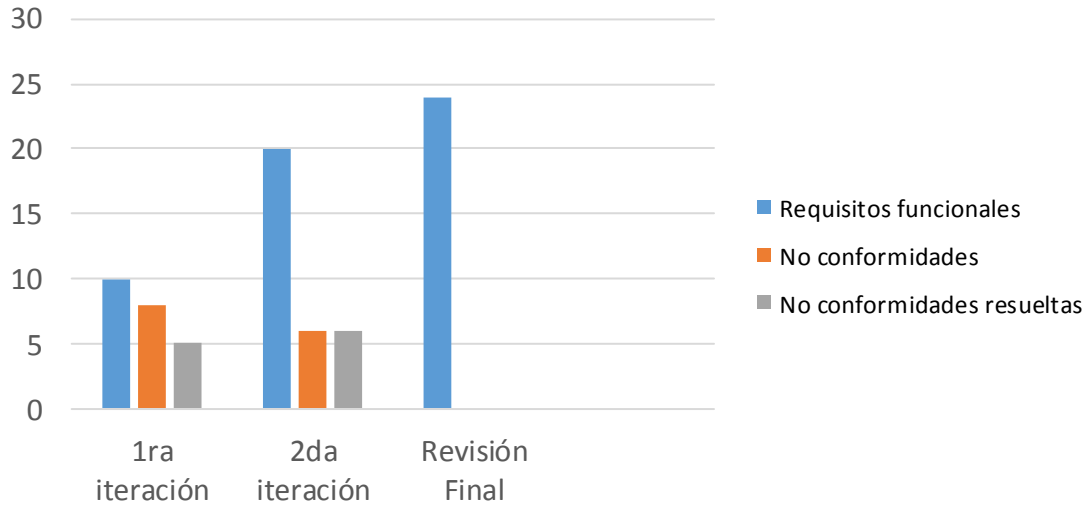


Figura 14: Gráfico de las iteraciones de las pruebas realizadas.

3.5.2. Unitarias

Estas pruebas se realizan para comprobar que el módulo no presenté errores en su codificación y que funcione adecuadamente. Para llevarlas a cabo se utilizó el método de caja blanca, utilizando la técnica del camino básico.

Pruebas de caja blanca

Estas pruebas intentan garantizar que se ejecuten al menos una vez todos los caminos independientes que presente el módulo y que todas las estructuras de datos internas sean usadas.
Se realizaron pruebas a las principales funcionalidades del módulo utilizando la técnica del camino básico, para lo cual es necesario conocer el número de caminos independientes de un determinado algoritmo mediante el cálculo de la complejidad ciclomática.
El cálculo de la complejidad ciclomática se realiza de tres formas diferentes:

- El número de regiones del grafo de flujo coincide con la complejidad ciclomática.
- La complejidad ciclomática, V (G), de un grafo de flujo G se define como: V (G) = A – N + 2, donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.

- La complejidad ciclomática, V (G), de un grafo de flujo G también se define como: V (G) = P + 1, donde P es el número de nodos predicado (son los nodos de los cuales parten dos o más aristas) que tiene contenido el grafo de flujo G.

Ejemplo de código de la funcionalidad createAction:

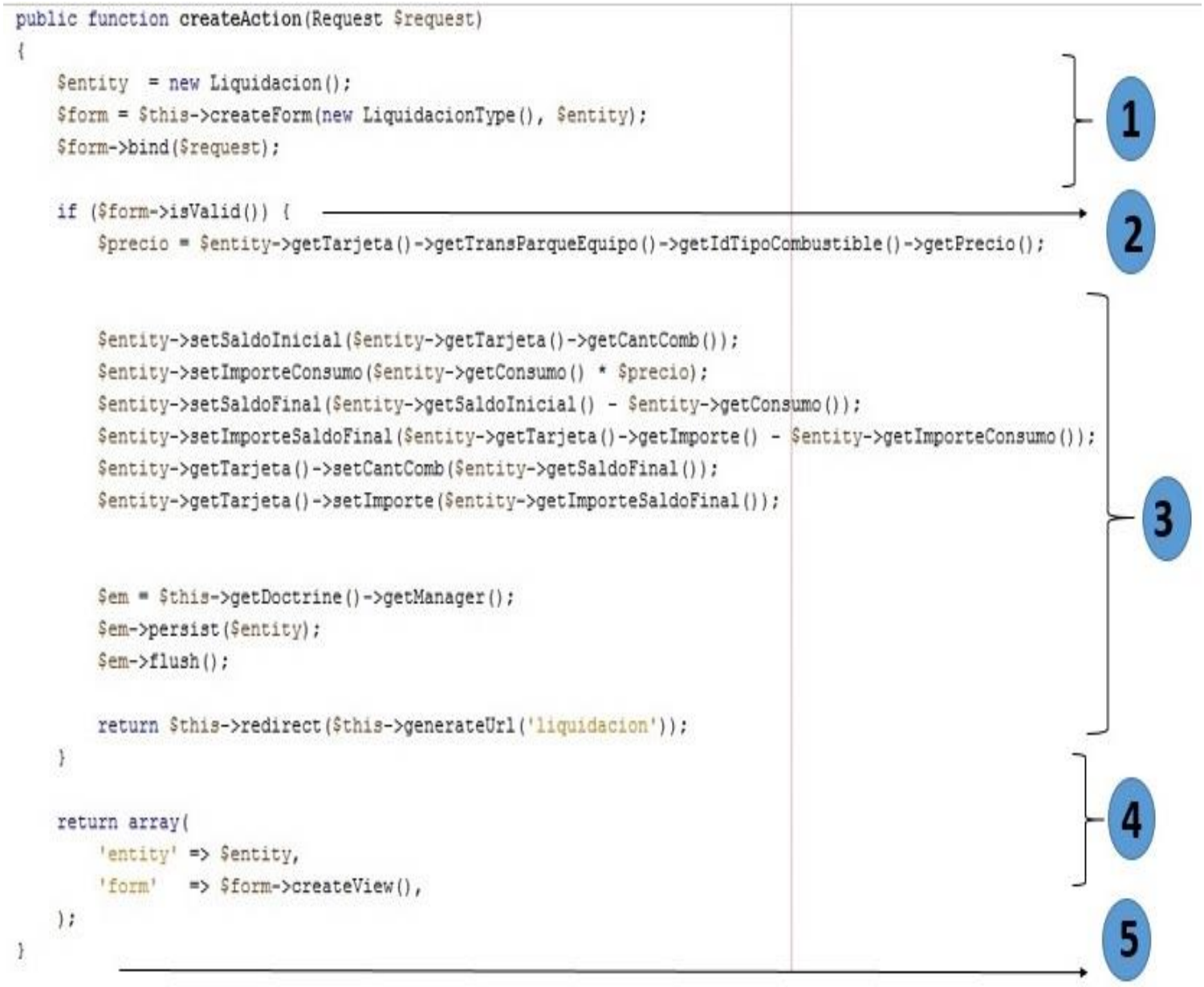


Figura 15: Código de la funcionalidad createAction**.**

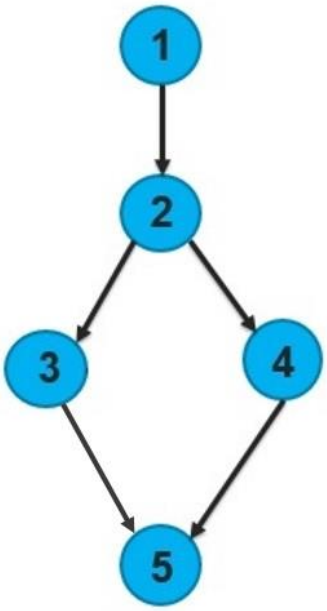


Figura 16: Grafo del flujo asociado a la funcionalidad createAction.

Resultado del cálculo de la complejidad ciclomática:

- El grafo de flujo tiene 2 regiones.
- **V(G)** = A – N + 2 = 5 – 5 + 2 = 2
- **V(G)** = P + 1 = 1 + 1 = 2

A partir del resultado obtenido, se determina que la funcionalidad presenta una complejidad ciclomática de 2, lo que deriva que existen a lo sumo 2 caminos lógicos por donde ejecutarse dicha funcionalidad. En la siguiente tabla se presentan los caminos básicos:

Tabla 10: Caminos lógicos.

Nro.	Camino básico
1	1, 2, 4, 5
2	1, 2, 3, 5

Se procede a realizar los casos de prueba para la funcionalidad en cuestión. Se debe elaborar al menos un caso de prueba por cada camino básico.

Casos de pruebas

Tabla 11: Caso de prueba camino #1.

Caso de prueba	
Camino	1, 2, 4, 5
Descripción	Crear una nueva liquidación de combustible.
Condición de ejecución	Tiene que estar autenticado. Tiene que tener permisos. Debe haberse efectuado la carga y entrega de la tarjeta.
Entrada	Se introduce un número positivo y se escoge un valor del campo modelo.
Resultados esperados	Se ha creado la nueva liquidación.

Tabla 12: Caso de prueba camino #2.

Caso de prueba	
Camino	1, 2, 3, 5
Descripción	Crear una nueva liquidación de combustible.
Condición de ejecución	Debe estar cargada la tarjeta de combustible.
Entrada	Se introduce un valor negativo y se escoge un valor del campo modelo.
Resultados esperados	Se obtiene un mensaje de mensaje de error.

3.5.3. Integración

Después de validadas las pruebas unitarias se pasa a realizar a las pruebas de integración. Aun cuando los módulos de un programa funcionen bien por separado es necesario probarlos conjuntamente ya que pueden traer efectos adversos sobre otros módulos. A partir del esquema del diseño, los módulos probados se vuelven a probar combinados para probar sus interfaces y verifica que todos los componentes funcionan juntos.

Tabla 13: Caso de prueba de integración.

Caso de prueba
Componentes que la integran: Trasporte, Contabilidad y Configuración.
Descripción: El módulo de Contabilidad utiliza los nomencladores Tipo de Combustible, Dependencia y Estructura del módulo Configuración; y la Asignación Mensual y el Parque de Equipos del módulo Trasporte.
Condiciones de ejecución: 1. Debe existir un usuario registrado en el sistema. 2. Deben estar definidos los niveles de acceso. 3. Es necesario llenar todos los campos estén llenos. 4. Tiene que estar registrado como mínimo los nomencladores, las estructuras y dependencias, un vehículo y una tarjeta para realizar la asignación.
Entrada/Pasos de ejecución: 1. Autentificación 2. Crear estructura 3. Crear dependencia 4. Crear nomencladores 5. Crear Parque de Equipos 6. Realizar las operaciones del módulo contable (asignaciones, cargas, liquidaciones, reportes y estadísticas.)

Resultado esperado: El sistema funciona en su conjunto con todos los módulos integrados.
Evaluación de la prueba: Prueba satisfactoria

3.5.4. Resultados de las pruebas

Luego de las pruebas aplicadas al módulo desarrollado se encontraron y corrigieron todos los errores presentados proporcionando un producto de calidad para el cliente. Se efectuaron 3 iteraciones, en una primera iteración fueron detectadas 6 no conformidades en las pruebas unitarias, 8 no conformidades en las pruebas de aceptación y 10 no conformidades en las pruebas de integración. Se corrigieron la mayoría de los errores detectados, los demás pasaron a la próxima iteración y luego de la segunda iteración se encontraron 4 no conformidades en las pruebas unitarias, 5 no conformidades en las pruebas de aceptación y 5 no conformidades en las pruebas de integración. Al concluir la segunda iteración y ser resueltas todas las no conformidades encontradas en ella se pasa a la última iteración Luego de ser resueltas las no conformidades detectadas en la segunda iteración y concluida la tercera iteración sin encontrar no conformidades el módulo desarrollado queda listo para ser puesto en uso para la satisfacción de las necesidades del cliente.

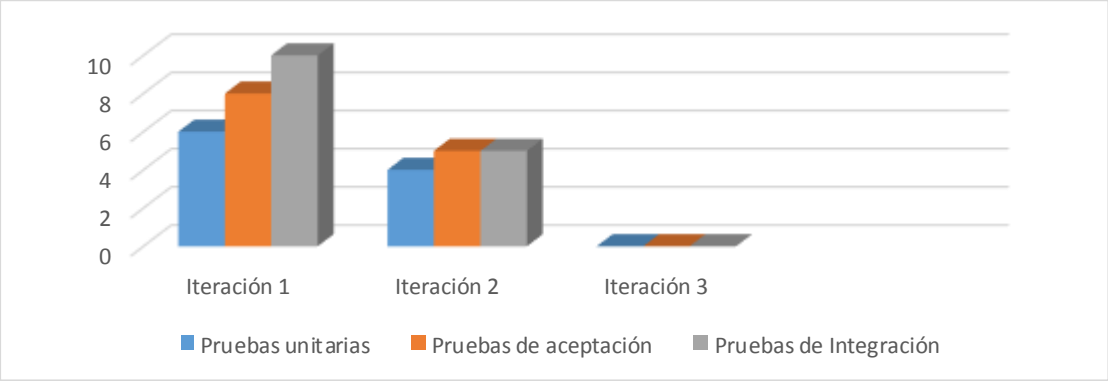


Figura 17: Resultados de las pruebas.

3.6. Conclusiones

En el desarrollo de este capítulo se definió el estándar de codificación utilizado en la implementación del módulo permitiendo una uniformidad, claridad y fácil entendimiento del código. Los diagramas de componentes y despliegue presentados permitieron una mayor apreciación de la lógica y la organización

de los componentes existentes ayudando a su implementación. El desarrollo de las pruebas detectó no conformidades, las cuales fueron corregidas en cada iteración, asegurando una solución aceptada en todo momento de la implementación.

Conclusiones generales

Durante el desarrollo de la presente investigación se le da solución al problema de investigación que le dio inicio, a través del cumplimiento de sus objetivos específicos y tareas de investigación, arrojando las siguientes conclusiones:

- El estudio de los sistemas homólogos de gestión de la información del combustible a nivel nacional e internacional permitió una mayor comprensión del objeto de estudio y posterior desarrollo de la propuesta de solución acorde a las necesidades existentes.
- El enfoque ágil propuesto por la metodología de desarrollo de software XP, el uso de las tecnologías y herramientas seleccionadas y el modelado de los procesos de negocio permitió un correcto análisis y descripción de los procesos a ejecutar así como facilidades para su desarrollo.
- El estándar de codificación, los diagramas de componente y despliegue permitieron un mayor entendimiento del producto y la lógica dentro de sus componentes facilitando su implementación y futuro mantenimiento.
- Luego de la realización de las pruebas de aceptación, unitarias y las de integración se permitió comprobar el buen funcionamiento del módulo con resultados satisfactorios.

Recomendaciones

Capacitar el personal encargado de interactuar con la aplicación.

Mantener actualizada las informaciones en el sistema.

Migrar el sistema hacia la versión 2.7 o superior del *framework* Symfony2.

Por la utilidad práctica del sistema, se considera que puede ser usado por otras Universidades o entidades del país.

REFERENCIAS

ALLEN, R. 1992. *A formal approach to software architectures.* 1992.

ARIAS CALLEJA, MANUEL. 2012. *Estándares de codificación.* 2012.

ARRENDONDO, K. 2012. Servicio Web 2.0. Apache vx IIS. 2012.

BOOSTRAP. 2013. [En línea] 2013. [Citado el: 20 de Febrero de 2015.] http://sistemasweb2013.webcindario.com/bootstrap/caracteristicas.php..

COLECTIVO de AUTORES. 2009. *´´SGUIA´´.* 2009.

COMBUSTIBLES, DEFINICION. 2004. Definición de combustible. [En línea] 2004. http://www.definicionabc.com/general/combustible.php..

CONTAPYME. 2014. ContaPyme. Casa desarrolladora de AgroWin y ContaPyme. 2014.

DELGADO,C. POYO, A. 2010. COMPARATIVA FRAMEWORKS. Desarrollo de Aplicaciones web II. 2010.

DOYLE, M. 2012. Beginning PHP 5.3. 2012.

E-DUCATIVA. 2014. http://educativa.catedu.es. [En línea] 2014. http://e-ducativa.catedu.es/44700165/aula/archivos/repositorio//3500/3669/html/3_recursos_energticos.html.

EUGCOM. 2009. Administración y Control de Flotas. Chile: s.n., 2009.

FERREIRA, J. 2013. Creando Diagramas UML con StarUML. 2013.

FIGUEROA, ROBERTH G; SOLÍS, CAMILO J; CABRERA, ARMANDO A;. 2010. *Metodologías Tradicionales vs Metodologías Ágiles.* Loja : s.n., 2010.

FLOTAWEB. 2010. Software Incremental. 2010.

Fundamentos de Ingeniería. Inspección, Validación, Completitud, Detección de Conflictos e Inconsistencias de Requerimientos. **2012.** 2012.

GAVILAN, F. 2011. Internet Information Services (IIS). 2011.

GOMEZ, J. 2014. Motores MySQL. 2014.

GONZALEZ, E. 2013. Proyecto complejo deportivo. 2013.

Gracia, L. 2013. *´´Técnica para la captura de Requisitos´´.* 2013.

GRAÑA,R. 2011. Metodología de desarrollo de proyectos informáticos en entornos web. 2011.

HOWARD, Y. 2012. Use Case Modelling Lab using Visual Paradigm. 2012.

IDEA. 2009. IDEA. [En línea] 2009. [Citado el: 10 de Febrero de 2015.] http://www.idae.es/uploads/documentos/documentos_10232_Guia_gestion_combustible_flotas_carretera_06_32bad0b7.pdf.

INFORMATICA HOY. 2007. [En línea] 2007. www.informatica-hoy.com.ar.

INGENIERÍA De SOFTWARE. PROGRAMACIÓN EXTERNA XP. [En línea] http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html..

JACOME, S., TITUAÑA, W. TORRES, E. 2009. Elaboración de un manual de la plataforma Netbeans Ide para la Disicom. 2009.

JETBRAINS. 2013. jetbrains. [En línea] 2013. [Citado el: 6 de Febrero de 2015.] http://www.jetbrains.com/phpstorm/documentation/phpstorm_web.pdf.

KREASOFT. 2010. Software y Tecnología. 2010.

LARMAM, C;. 2008. *UML y Patrones Capítulo 9 MODELO DE CASOS DE USO: REPRESENTACIÓN DE LOS DIAGRAMAS DE SECUENCIA DEL SISTEMA.* 2008.

LETELIER, PATRICIO. 2012. *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* Valencia : Universidad Politécnica de Valencia, 2012.

LETELIER, PATRICIO y PENADÉS, MA CARMEN. 2012. *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* España : Universidad Politécnica de Valencia, 2012.

MARCISZACK, M. 2011. Modelos de especificación de requerimientos para la obtención de esquemas conceptuales en un dominio restringido: comparación de metodologías. 2011.

MARTINEZ, R. 2013. Getting Started with JSP. 2013.

NARAMORE, ELIZABETH. 2005. Beginning PHP5, Apache and MySQL Web Development. Canadá : s.n., 2005.

NOVELLA, J. 2012. Sistema de Gestión de Base de Datos PostgreSQL. 2012.

ORJUELA, A. ROJAS, M. 2008. The methodologies of Agil Development like and Opportunity for the Inginneering of Educative Software. 2008.

PACHECO, YISEL. 2010. Diseño de un sistema de control de pago de dietas y facturas para el Departamento de Transportaciones Nacionales de la UCI. Cuba, La Habana : s.n., 2010.

PEREZ, O. GONZALEZ, S.MARTINEZ, Y. 2013. *La Gestión Energética en el contexto empresarial cubano.* Cuba : s.n., 2013.

POLO,M. 2011. Introducción a patrones. [En línea] 2011. [Citado el: 4 de febrero de 2015.] http://www.inf-cr.uclm.es/www/mpolo/asig/0304/0102/patronesgrasp.ppt.

POTENCIER, F. ZANINOTTO, F. 2010. Symfony en pocas palabras. 2010.

PRADO, F. 2010. Gestión de Flotas. 2010.

PRESSMAN, R. 2010. *Software engineering practitioner's approach.* New York : s.n., 2010.

PRESSMAN, Roger. 2005. *´´Ingeniería del Software Un enfoque práctico´´.* 2005.

RIVAS, J. 2010. *´´´Análisis y diseño de sistemas. Casos de Uso´.* 2010.

RODRIGUEZ, ORQUIDIA FERRER, DIAZ, RAUNER BESTARD. 2013. Sistema de Gestión de la Información para las Ópticas de Cuba. Cuba, La Habana : s.n., 2013.

SAAVEDRA, A. RODRIGUEZ R. 2013. desarrollo del módulo Control de Combustible del Sistema de Control de Flota y Mantenimiento . 2013.

SANCHEZ, A. GARCIA, WILSON, J. WILSON, A. 2011. *Gestión de Portadores Energéticos en la UCI.* Cuba : Serie Científica de la Universidad de las Ciencias Informáticas, 2011. Vol. 4.

SANCHEZ, ING. TAMARA RODRIGUEZ, GONZALEZ, ING. MAIRELYS FERNANDEZ Y CASA, ING. ELIECER CABRERA. 2011. La Gestión Empresarial de las Entidades Cubanas. Cuba, la Habana : s.n., 2011, Vol. 15.

Sistema de gestión de información de las redes sociales Facebook y Twitter. **Vázquez Ventura, Merlyn; RODRÍGUEZ Quintana, Daniel Raúl. 2013.** 2013.

TORRES, C. 2008. Establecimiento de una Metodología de Desarrollo de Software para la Universidad de Navojoa Usando OpenUp. 2008.

TRANSOFT. 2010. Nuestros Productos. [En línea] 2010. http://www.transoft.transnet.cu/.

VALDÉS, J. 2014. *Diagramas de componentes.* 2014.

VALDIVIÉ FERNANDEZ, A. MANUEL Y HERNANDEZ REYES, M. JOSE. 2014. SISTEMA DE GESTIÓN DE LA INFORMACIÓN DEL COMBUSTIBLE DE LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS. Cuba, La Habana : s.n., 2014.

Vázquez Ventura, Merlyn; Rodríguez Quintana, Daniel Raúl. 2013. *Sistema de gestión de información de las redes sociales Facebook y Twitter.* 2013.

VELÁZQUEZ LEYVA, REYNERIO ; DARROMÁN SAVIGNE, CARIDAD;. 2011. *"El proceso de gestión y la gestión económica en las empresas".* 2011.