UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 1

Módulo de gestión de información para el control de pasajes por voucher Viazul.

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor: Liosbel Margolles Clausell

Tutores:

Ing. Yuneldis Reyes Velázquez

Ing. Evelyn Labrada Oduardo

La Habana, junio de 2017

"Año 58 de la Revolución"

Seamos realistas, hagamos lo imposible.

Ché

DECLARACIÓN DE AUTORÍA

Declaro por este medio que yo Liosbel Margolles Clausell, con carné de identidad 93011300307 soy el autor principal del trabajo titulado “Módulo de gestión de información para el control de pasajes por voucher Viazul” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

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

_____

Liosbel Margolles Clausell

Autor

_____	_____
Ing. Evelyn Labrada Oduardo	Ing. Yuneldis Reyes Velázquez
Tutor	Tutor

AGRADECIMIENTOS

Le agradezco a mis hermanas Leydis y Oleydis, por estar siempre conmigo, apoyarme y creer en mi cuando a veces no lo hago.

Le agradezco a mi tío por quererme como un hijo y por darme una prima tan bella.

A mis amigos de estos cinco años que llegué a querer y los voy a extrañar.

A Dayron por ser mi hermano del alma para toda la vida.

Le agradezco a mi abuela por ser mi segunda madre y quererme con locura.

En especial le agradezco a mi madre que sin ella este día no hubiera sido posible, por criarnos sola y ser madre y padre al mismo tiempo, te amo.

RESUMEN

El presente trabajo describe el sistema de gestión de información para el control del pago mediante *voucher,* a la empresa Víazul. Este sistema permite gestionar y controlar la información, incorporar medidas de seguridad mediante la creación de usuarios los cuales acceden a la información en dependencia del rol que desempeñan, entre otras funcionalidades. Se puntualiza el conjunto de herramientas, lenguajes de programación y tecnologías utilizadas donde se especifican las buenas prácticas para el desarrollo y se definen un conjunto de patrones y arquitectura a utilizar. La investigación realizada tiene como finalidad desarrollar una aplicación *web* que permita el manejo y control de la información asociada a los *voucher*. Para ello se lleva a cabo un estudio y valoración de los sistemas económicos de gestión de pagos más utilizados internacionalmente y en el país, a fin de encontrar opciones y elementos a tomar en cuenta para obtener la solución deseada. Debido a que los sistemas contables existentes a nivel nacional e internacional no cuentan con funciones que permitan gestionar la información referente a los *voucher*; se hizo necesaria la informatización de dicho proceso el cual permite registrar la utilización de un *voucher*, archivar el consumo del mismo asociado al área de la institución que lo consume. Con la implantación del sistema el proceso se agiliza y se garantiza la seguridad de la información.

Palabras clave: control de pagos, gestión de facturas de trasporte, gestión de información, *voucher*.

ÍNDICE

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Las nuevas tecnologías empleadas en la computación y el desarrollo de Internet, no sólo como instrumentos de difusión y entretenimiento sino como herramientas de negocio, han cambiado la fisonomía de la informática en la última década. Con el auge en paralelo de los entornos cliente/servidor y de las comunicaciones, se ha aproximado la tecnología telemática a los usuarios finales, de tal modo que casi no se puede imaginar un ambiente de negocio que no considere la informática como una herramienta básica para su desarrollo. Sin embargo, tras una fase de investigación de las funcionalidades y del potencial de desarrollo de estos sistemas, se ha llegado a la conclusión por parámetros cuantitativos y no funcionales que ya se planteaban en arquitecturas centralizadas. Así, por ejemplo, la seguridad, la disponibilidad, la eficiencia o el rendimiento son algunas de las características que normalmente se evalúan a la hora de explotar un sistema informático (Xavier, 2015).

Desde hace algunos años el desarrollo de la informática y las nuevas tecnologías está en constante crecimiento, la utilización de las mismas proporciona facilidades en el control y desarrollo de sistemas, por lo cual en la actualidad muchas empresas e instituciones se han motivado en el uso de estas en aras de optimizar los procesos que realizan en tiempo y costo de ejecución (DESOFT, 2015). Dentro de estos se encuentran los diferentes procesos de pago, los que se llevan a cabo tradicionalmente de manera manual pues de alguna forma siempre se han visto dificultados debido a la situación de bloqueo que tiene nuestro país, lo cual le imposibilita adquirir algunas herramientas para realizar eficientemente este proceso.

El proceso de pago se realiza cuando se adquiere un servicio o se participa en algún servicio en el cual se vea implicado el sistema financiero de cualquier persona o entidad que participe. Uno de los factores que influirá es la forma de pago en que se vaya a liquidar el servicio que se ha brindado. El pago puede ser realizado al contado, término que no se realiza en la Universidad de las Ciencias Informáticas (UCI). La realización del pago al contado puede hacerse de muy diversas formas; entre las más habituales se encuentra: la entrega de dinero en efectivo, en cuyo caso se desarrolla en el Caja de la UCI en los pagos de dietas o las transferencias bancarias que es desarrollada al igual que el cheque a pagos a empresas.

En las instituciones de Educación Superior de nuestro país existen diferentes procesos de pago que se realizan de forma manual. Los principales son el pago de dietas a estudiantes, profesores y trabajadores externos al centro que realizan algunas tareas en colaboración con el mismo, y el pago

del 50% del pasaje a estudiantes como concepto de dieta por liquidación de la bonificación aprobado luego del VII Congreso de la Federación Estudiantil Universitaria (FEU). Otro proceso de pago importante lo constituye el pago de factura a organismos externos a las universidades, que prestan algún servicio, un ejemplo de ellos son los pagos que se realizan en divisa mediante documento de pago por anticipo que se emite a la Empresa de Transporte Viazul.

La Universidad de las Ciencias Informáticas posee una Vicerrectoría Económica que se subdivide en diferentes departamentos de trabajo que de una forma u otra laboran de forma conjunta, compuesta por: Dirección de Contabilidad y Finanzas, Dirección de Planificación y Estadística, Dirección de Compras y Almacenes y el Grupo de Gestión Energética. De todas estas direcciones es la de: Contabilidad y Finanzas, la encargada de registrar todos los hechos económicos que se llevan a cabo en el centro y ejecuta y controla el pago en divisa de los pasajes que se autorizan por parte de la Rectora mediante *voucher*[1] a la Empresa Viazul (Pérez, 2015).

Actualmente estos procesos de pago se realizan de forma no automatizada en la Dirección de Contabilidad y Finanzas, ya que esta solo registra el gasto asociado al área de la UCI en que fue solicitado, pero no crea un expediente con nombre, destino del viaje, así como las fechas de ida y regreso del viajero. Además, lo que implica entre otras dificultades, que no se tiene acceso a la cantidad de veces que una persona puede hacer uso de un *voucher* en divisa, tampoco la cantidad de veces que un área del centro ha ejecutado una solicitud, así como el estado en que está el *voucher* que fue entregado. Tampoco se tiene control del gasto del *voucher* sobre la forma de pago de cheque por el cual se deposita en la Empresa de Viazul del anticipo para ser gastado.

A partir de la problemática antes planteada se define como **problema científico:** ¿Cómo gestionar el control de los pagos de pasajes en divisa por *voucher* a la empresa Viazul para controlar debidamente el proceso?

El **objeto de estudio** lo constituye los sistemas contables para la gestión de la información y el **objetivo general** de esta investigación es desarrollar una aplicación web que gestione el control de los pagos de pasajes en divisa por *voucher* a la empresa Viazul enmarcado en el **campo de acción** los sistemas de gestión de la información de pagos.

[1] Se refiere al comprobante o resguardo de operaciones bancarias o a un vale para adquirir o disfrutar de productos o servicios.

A partir de lo planteado anteriormente se desglosan los siguientes **objetivos específicos:**

1. Diagnosticar el estado actual de la gestión de entrega de *voucher* en divisa por Viazul.
2. Diseñar los elementos que se ajustan a las necesidades del proceso de gestión de entrega de *voucher* en divisa en Viazul.
3. Fundamentar la selección de la metodología, herramientas y tecnologías a utilizar en el desarrollo del sistema de gestión de la información.
4. Realizar el análisis y diseño del sistema para la gestión de la información referente al pago de pasajes en divisa por *voucher* de Viazul.
5. Realizar el análisis de las funcionalidades contables para el pago de pasajes en divisa por *voucher* de Viazul.
6. Implementar el sistema para la gestión de la información referente al pago de pasajes en divisa por *voucher* de Viazul.
7. Realizar pruebas al sistema para la gestión de la información referente al pago de pasajes en divisa por *voucher* de Viazul.

Se define como **Hipótesis:** Si se desarrolla una aplicación web que gestione el control de los pagos de pasajes en divisa por *voucher* a la empresa Viazul será posible agilizar el proceso de pago.

Variable independiente: Aplicación web que gestione el control de los pagos de pasajes en divisa por *voucher* a la empresa Viazul.

Variable dependiente: Agilizar el proceso de pago por *voucher* en divisa a la empresa Viazul.

Métodos de investigación:

Métodos Teóricos:

- ✓ ***Analítico-sintético:*** se utilizó durante el estudio y análisis de las bibliografías consultadas para fundamentar, asumir posiciones y determinar los elementos en relación con el objeto investigado, además de las características generales y las relaciones esenciales inmersa en el proceso que se está investigando (Chagoya, 2012).
- ✓ ***Modelación:*** Con el mismo se lleva a cabo la representación de cómo ocurren los procesos a implementar, además para crear los modelos definidos en la metodología (Chagoya, 2012).

Métodos empíricos:

- ✓ ***Estudio de casos*:** se utilizó con el objetivo de valorar el resultado científico de las investigaciones, validar los beneficios y la aplicabilidad de la propuesta (Chagoya, 2012).
- ✓ ***Pruebas de validación*:** se empleó para validar las variables utilizada en la investigación, así como en el desarrollo de la solución propuesta (Chagoya, 2012).
- ✓ ***Consulta bibliográfica**:* permitió la elaboración del marco teórico de la investigación referente a los sistemas existentes de gestión de la información (Chagoya, 2012).

El presente trabajo se encuentra estructurado en tres capítulos, resumidos de la siguiente forma:

En el **Capítulo 1: Fundamentación Teórica:** se realiza una investigación para esclarecer los conceptos más importantes. Se especifica sobre la utilización de las herramientas, tecnologías y lenguajes de programación que se van a utilizar en el desarrollo del software para el diseño de modelos de negocio. Se plantea la metodología de desarrollo por la cual estará transitado el software en cuestión.

En el **Capítulo 2: Descripción de la solución propuesta:** en este capítulo se presenta una visión del software a desarrollar. Se expondrán los requisitos funcionales y no funcionales que garantizarán el desarrollo de la solución al problema además se determinan los patrones de diseños y arquitectónicos.

En el **Capítulo 3: Implementación y pruebas:** en este capítulo se especifican las tareas realizadas y se plasma el estilo de implementación utilizado, además se realizan las pruebas necesarias para verificar el correcto funcionamiento del software y para conjunto con el cliente probar y aceptar el software.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Introducción

El presente capítulo aborda los principales conceptos asociados al dominio del problema y que están relacionados con los procesos de pago que se realizan en la Dirección de Contabilidad y Finanzas en la UCI mediante *voucher* a la Empresa Viazul. Se realiza un estudio del estado del arte donde queda reflejado un análisis crítico de los sistemas de gestión de la información a nivel nacional e internacional. Además, se describen las características de los lenguajes de programación, herramientas, tecnologías y metodología para el desarrollo de la solución propuesta.

1.1 Marco Teórico. Conceptos y definiciones

Voucher

Es un término de la lengua inglesa que no forma parte del diccionario de la (Real Academia Española, 2016). El concepto puede traducirse como cupón, comprobante o vale. En muchos países, el uso más habitual de un *voucher* es que se refiere al documento que acredita el pago de un producto o de un servicio y que puede intercambiarse, llegado el momento, por aquello que se adquirió (Pérez, 2015).

Un vale es un documento interno que describe y autoriza el pago de una obligación a un proveedor. Es más comúnmente utilizado en un sistema de pago manual (Bragg, 2014).

El diccionario de Cambridge define que un *voucher* es: Un pedazo de papel que se puede utilizar para pagar por bienes o servicios particulares o que le permite pagar menos que el precio habitual para ellos (Press, 2015).

En la Gaceta Oficial No. 040 se designa un *voucher* como el documento que lleva implícito una orden de prestación de uno o varios servicios, siendo a su vez, un comprobante de pago y de reservación de dichos servicios. Para que un *voucher* de servicio tenga validez debe estar debidamente sellado por la oficina que lo emite (Gaceta Oficial, 2011).

Partiendo de los conceptos antes estudiados se puede plantear que un *voucher* no es más un documento que contiene implícitos servicios el cual solo es válido según el sello de la institución que lo emita. Por lo cual para la presente investigación se decide como concepto a seguir el propuesto por la Gaceta Oficial.

Gestión de Pagos

Gestión de pagos son todas aquellas tareas de gestión, control, administración y envío de las transacciones monetarias a los proveedores en una organización (Portas, 2003).

La organización de la gestión de pagos se encuentra dentro de la tesorería en el departamento financiero de una empresa, corresponden al departamento de gestión de pagos la validación de proveedores, comprobación de saldos, envío de dinero y conciliación con los datos del proveedor de manera habitual. Cuando se crea un perfil nuevo de acreedor, e incluso en ocasiones es fundamental realizar auditorías a nivel interno sobre pagos y saldos entre proveedor y cliente (Steven, 2015).

Se difiera de la gestión de cobros en que, mientras que una empresa gestiona los cobros de manera propia, es decir, puede poner sus condiciones, reclamaciones y segmentar según el tipo de cliente, en los pagos la empresa está obligada a realizar los pagos según las condiciones pactadas con el acreedor (Steven, 2015).

1.2 Soluciones existentes

En la actualidad existen un conjunto de sistemas que se emplean en el control y gestión de la economía en las instituciones, los mismos necesarios para el control de los recursos materiales, financieros y humanos, aplicando sistemas de registro. A continuación, se expone una breve descripción de algunos sistemas de gestión económica a nivel internacional y nacional, lo que permitirá realizar un análisis con el objetivo de determinar si los mismos se pueden emplear como posible solución de la investigación.

Internacionales

Existen muchos sistemas encargados de gestionar las actividades económicas de las organizaciones e instituciones. Es tendencia que estos sistemas sean desarrollados con un alto nivel de personalización, con el objetivo de que se ajusten lo más posible a las particularidades de las instituciones que los utilizan, sin embargo, estos sistemas si bien no son desarrollados para controlar los procesos económicos de una determinada institución, permiten realizar la gestión económica de forma muy eficiente, ejemplo de ellos son:

ContaPyme

ContaPyme (Sistema de gestión empresarial y contable para PYMES), es una herramienta que maneja de forma integrada las áreas económicas de gestión de una organización. Él mismo permite llevar la

contabilidad de manera automática, el control total de la información y fácil ingreso de esta, además de la generación de todos los informes financieros y libros legales (ContaPyme, 2014).

SAP

SAP es uno de los Sistema de Recursos de Empresas (*Enterprise Resource System*, en inglés abreviado ERP) más utilizado en el mundo. Este sistema tiene muchos módulos integrados que abarcan 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 (Informática Hoy, 2014).

Nacionales

En Cuba se utilizan varios sistemas que posibilitan a empresas, entidades u organismos del estado, llevar un sistema de registro y control de todos los hechos económicos que se evidencian a diario, ejemplo de esto son los siguientes sistemas:

ASSETS

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.

CONDOR

Sistema automatizado de alta complejidad y seguridad que abarca todos los aspectos del proceso contable de una entidad, tales como la dualidad de moneda y el pago por resultados. Está formado por varios módulos como activos fijos, contabilidad general, nóminas, control de inventarios, recursos humanos, entre otros (SÁNCHEZ, 2011).

Versat Sarasola

Es uno de los primeros sistemas de contabilidad utilizado en Cuba. Se conforma por varios módulos que permiten llevar el control y registro contable individual de todos los hechos económicos que se

originan en las entidades, facilitando el análisis y la evaluación de los resultados del negocio o actividad en tiempo real (SÁNCHEZ, 2011).

Tabla 1 Datos generales de los sistemas de gestión de pagos en el ámbito nacional e internacional. (Fuente: Elaboración propia).

Sistema	**Pago x concepto *voucher***	**Plataforma**	**Soporte**	**Certificación**	**Licencia**
ContaPyme	No	Windows 8, Windows Server 2012 o superior.	Si	Certificación de calidad NTC 6001[2]	Licencia Propietaria
SAP	No	Windows 2000 - 2003 Server.	Si	Certificación con las normas internacionales. Desde 1998, SAP posee un certificado ISO 9001. También estan certificados según ISO 27001, ISO 22301 y BS 10012.	Licencias de código abierto y de Software de terceros [3]

[2] La Norma NTC 6001 - Sistema de Gestión para Micro y Pequeñas Empresas MYPES fue emitida por ICONTEC desde el año 2008, como un aporte a la legalización, viabilidad y competitividad para las Micro y Pequeñas Empresas.

[3] En informática, una aplicación o programa de terceros hace referencia a aquellas aplicaciones que son desarrolladas por empresas que no tienen relación con una empresa en particular.

ASSETS	No	Windows 2000 - Windows 2003 Server, o superior	No	Departamento de Informatización de la UCI.	Licencia GNU GPL[4]
CONDOR	No	MSDOS[5]	Sí	Certificado válido otorgado por la Dirección General de Informática.	Licencia Propietaria
Versat Sarasola	No	Windows	Si	Certificado válido otorgado por la Dirección General de Informática.	Licencia Propietaria

Conclusiones del estudio

A partir del estudio realizado a los sistemas anteriores que de una forma u otra se relacionan con el control y registro contable de los hechos económicos que se llevan a cabo en las empresas, se determina no utilizar ninguno de ellos para apoyar la investigación, debido a que en ninguno se implementa la funcionalidad del pago por concepto de vouncher, siendo este el objetivo principal de dicha investigación, además, estos no llevan un expediente donde quedan expuestos los datos del registro y control del pago de vouncher, por tanto, no dan solución al problema planteado en la presente investigación, otra de las deficiencias que presentan los mismos es que son sistemas con licencia privativa excepto Assets pero el mismo solo se puede usar en plataformas privadas por lo cual se descarta su uso teniendo en cuenta que muchas instituciones cubanas, entre ellas la Universidad de las Ciencias Informáticas UCI) han optado por el uso del software libre contribuyendo al desarrollo de sistemas informáticos de gran importancia para el país. Por lo estipulado anteriormente se dispone a

[4] La Licencia Pública General de GNU (o simplemente sus siglas del inglés GNU GPL) es la licencia de derecho de autor más ampliamente usada en el mundo del software libre y código abierto.

[5] Siglas de MicroSoft Disk Operating System, Sistema operativo de disco de Microsoft) es un sistema operativo para computadoras basado en x86.

implementar un SGI que cubra las necesidades detectadas en el pago por *voucher* de Viazul en la UCI y que sea capaz de satisfacer los requerimientos del negocio exigidos por el cliente.

1.3 Metodología, lenguajes y herramientas para el desarrollo

Metodología

Según la Real Academia Española una metodología de desarrollo de software no es más que un conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal (Real Academia Española, 2016) .

Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Hoy en día existen numerosas propuestas metodológicas que inciden en el proceso de desarrollo. Ejemplo de ellas son las propuestas tradicionales, centradas específicamente en el control del proceso. Estas han demostrado ser efectivas y necesarias en un gran número de proyectos, sobre todo aquellos proyectos de gran tamaño (respecto a tiempo y recursos). Sin embargo, la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre.

Existen varias clasificaciones para estas metodologías dentro de las cuales se destacan dos: Metodologías tradicionales y Metodologías ágiles, en estas metodologías se encuentran RUP [6] (Proceso Racional Unificado), MSF[7] (Marco de Solución de Microsoft) y *XP* (*Extreme Programming*), *AUP* (*Agil Unified Process*) y *SCRUM respectivamente.*

Para la presente investigación se asumirá una metodología ágil con el fin de aprovechar las facilidades que brinda en cuanto a la documentación y el entorno cambiante que tiene el proyecto. Luego de haber seleccionado la metodología ágil para el desarrollo del software para el diseño de modelos de negocios, a continuación, se realiza un análisis de varias de ellas:

Extreme Programming

Dentro de los procesos ágiles, este es el que más se destaca. Su mayor diferencia con las metodologías tradicionales es que se enfatiza más en la adaptabilidad que en la previsibilidad, por lo que un cambio

[6]Del común en inglés *Rational Unified Process.*

[7]Del común en inglés *Microsoft Solution Framework.*

de requisitos sobre la marcha es algo natural e inevitable, por lo que el equipo de desarrollo debe ser capaz de adaptarse a los cambios en cualquier punto de la vida del proyecto en vez de intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos mayores en controlarlos a lo largo del ciclo de vida.

Algunas de sus características son:

- ✓ Desarrollo iterativo e incremental: se trata de ir realizando pequeñas mejoras una tras otras.
- ✓ Frecuente interacción del equipo de programación con el cliente o usuario: es recomendado que el cliente o al menos un representante suyo trabaje junto al equipo de desarrollo.
- ✓ Deben corregirse todos los errores antes de añadir nuevas funcionalidades y hacer entregas frecuentes.

Agile Unified Process

Es un acercamiento aerodinámico al desarrollo del software basado en el Proceso Unificado Racional de la IBM[8], basado en disciplinas y entregas incrementales con el tiempo. El ciclo de vida en proyectos grandes es serial mientras que en los pequeños es iterativo. Algunas de las disciplinas de *AUP* según (Salvador, 2015) son:

- ✓ Modelado: entender el negocio de la organización, el problema de dominio que se aborda en el proyecto, y determinar una solución viable para resolver el problema de dominio.
- ✓ Implementación: transformar el modelo en código ejecutable y realizar un nivel básico de pruebas individuales.
- ✓ Prueba: realizar una evaluación objetiva para garantizar la calidad. Incluye la búsqueda de defectos, validar que el sistema funciona tal como está establecido y verificar que se cumplan los requisitos.

Sus principales características son:

- ✓ Iterativo e incremental.
- ✓ Manejo de los casos de uso.
- ✓ Centrado en la arquitectura.
- ✓ Enfocado en los riesgos.

[8] Del común en inglés *International Business Machines Corp.*

SCRUM

Es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo del software. El desarrollo es iterativo e incremental donde cada ciclo culmina con una pieza de software ejecutable que incorpora nueva funcionalidad. Por lo general cada ciclo tiene una duración de entre 2 y 4 semanas. Scrum está mayormente enfocado en priorizar el trabajo en función del valor que tenga para el negocio, diseñado para adaptarse a los cambios en los requerimientos. Su objetivo principal es entregar un software que realmente resuelva las necesidades, aumentando la satisfacción del cliente.

A continuación, se muestran algunas ventajas que posee esta metodología:

- ✓ Se obtiene software lo más rápido posible y este cumple con los requerimientos importantes.
- ✓ El trabajo es en iteraciones cortas, de alto enfoque y total transparencia.
- ✓ Se acepta que el cambio es una constante universal y se adapta el desarrollo para integrar los cambios que son importantes.

Por sí solas las metodologías de software analizadas no son factibles para cumplir con los objetivos de esta investigación. Por lo cual se decide utilizar la metodología XP debido a las ventajas que esta trae consigo entre las misma se menciona que es una metodología fácil de implementar, con una documentación ligera, flexible y eficiente (García, 2014).

1.4 Herramientas de desarrollo

En el mundo existe una amplia gama de herramientas, tecnologías y lenguajes que se utilizan a la hora de desarrollar un software, las mismas cuentan con un conjunto de características que les permiten a los desarrolladores facilidades en su implementación. A continuación, se deja plasmado el conjunto de herramientas y tecnologías que se emplean para desarrollar la solución propuesta. Para seleccionar la mismas se tuvo en cuenta las siguientes características:

- ✓ Conocimiento y experiencia en el uso de las herramientas, tecnologías y lenguajes en la vida académica, lo que permite mayor desenvolvimiento a la hora de su utilización.
- ✓ Licencia, esta otorga a los usuarios libertades de manera adecuada para adaptar el uso de las herramientas, tecnologías y lenguajes a sus necesidades.
- ✓ Multiplataforma, esta posibilita el uso de las herramientas, tecnologías y lenguajes en el sistema operativo que se emplea en la universidad, teniendo en cuenta que el software libre se ha convertido en una premisa de la independencia tecnológica en Cuba (Stallman, 2015).

1.5 Lenguaje de programación

PHP

PHP (acrónimo recursivo de PHP: *Hypertext Preprocessor*) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

Lo que distingue a PHP de algo del lado del cliente como Javascript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente (My PHP, 2016).

Algunas de las características de PHP son:

- ✓ Es un lenguaje multiplataforma.
- ✓ Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
- ✓ El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- ✓ Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
- ✓ Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- ✓ Posee una amplia documentación en su página oficial (Sitio Oficial), entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- ✓ Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- ✓ Permite aplicar técnicas de programación orientada a objetos.
- ✓ Biblioteca nativa de funciones sumamente amplia e incluida.
- ✓ No requiere definición de tipos de variables, aunque sus variables se pueden evaluar también por el tipo que estén manejando en tiempo de ejecución.
- ✓ Tiene manejo de excepciones.

Al utilizar este lenguaje de programación el desarrollador no está obligado a seguir una determinada metodología a la hora de programar (muchos otros lenguajes tampoco lo hacen), aun estando dirigido a alguna en particular, el programador puede aplicar en su trabajo cualquier técnica de programación y/o desarrollo que le permita escribir código ordenado, estructurado y manejable. Un ejemplo de esto son los desarrollos que en PHP se han hecho del Patrón de diseño Modelo Vista Controlador (o MVC),

que permiten separar el tratamiento y acceso a los Datos, la Lógica de control y la Interfaz de usuario en tres componentes independientes (My PHP, 2016).

CSS3

CSS son las siglas del inglés *Cascading Style Sheets* (Hojas de Estilo en Cascada) básicamente consiste en la información que define como va a ser la presentación de una web. Cuando se refiere a presentación, se refiere colores, efectos, tipos de letra que escogemos, de tal manera que se independiza del HTML que es el lenguaje donde se estructura toda la información que se manda al ordenador para que el navegador presente la bonita página web. CSS3 es la versión 3, donde se definen las características de este lenguaje. Ejemplo de esto es la tecnología desarrollada para separar la presentación de la estructura HTML. Esta tecnología aplica reglas de estilo a los elementos HTML, quedando de esa manera separada de la estructura HTML. Poco a poco este lenguaje se ha ido haciendo más importante entre los diseñadores gracias a toda la facilidad de uso, y los resultados que son muy flexibles (Diccionario Informático, 2016).

jQuery

jQuery es un framework para el lenguaje JavaScript, que permiten programar sin preocuparse por los diferentes navegadores, ya que funciona igual en todas las plataformas más habituales. Este framework, ofrece una infraestructura con la que se tendrá mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Por ejemplo, con jQuery se obtiene ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax, etc. Además, todas estas ventajas con jQuery se obtienen de manera gratuita, ya que el framework tiene licencia para uso en cualquier tipo de plataforma, personal o comercial. Para la realización de un proyecto jQuery es rico en sus funciones, ya que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol Document Object Model (DOM) y manejar eventos (Álvarez, 2012).

HTML5

HTML5 es un lenguaje markup (de hecho, las siglas de HTML significan Hyper Text Markup Language) usado para estructurar y presentar el contenido para la web. Es uno de los aspectos fundamentales para el funcionamiento de los sitios, pero no es el primero. Es de hecho la quinta revisión del estándar que fue creado en 1990. A fines del año (2015),

la W3C [9] la recomendó para transformarse en el estándar a ser usado en el desarrollo de proyectos venideros. Por así decirlo, HTML5 está relacionado también con la entrada en decadencia del viejo estándar HTML 4, que se combinaba con otros lenguajes para producir los sitios que se pueden ver hoy en día (Ruiz, 2017).

Con el uso de HTML5, se puede reducir la dependencia de los plug-ins que se deben instalar para poder ver una determinada web. Caso emblemático, es el de Adobe Flash, que se ve claramente perjudicado por la instauración de este estándar, con HTML5 se amplía el horizonte del desarrollo de aplicaciones que pueden ser usadas en una multiplicidad de dispositivos, además los usuarios pueden acceder a sitios web de manera offline, sin estar conectados a internet. Se suma también la funcionalidad de drag and drop[10], y también la edición online de documentos ampliamente popularizada por Google Docs (Ruiz, 2017).

1.6 Marco de trabajo para PHP

Symfony 2.7

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Symfony separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web (Potencier, 2011).

Symfony está desarrollado completamente con PHP y es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. A continuación, se muestran algunas de sus características (Potencier, 2011):

Symfony se diseñó para que se ajustara a los siguientes requisitos:

[9] Consorcio Mundial de la red común del inglés World Wide Web Consortium, es un consorcio internacional que genera recomendaciones y estándares que aseguran el crecimiento de la World Wide Web a largo plazo.

[10] "Desplazar y liberar" o "arrastrar y soltar" (*drag and drop*) es una expresión informática que se refiere a la acción de mover, con el cursor del ratón, los objetos de una ventana a otra o entre partes de una misma ventana.

- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- ✓ Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

1.7 Lenguaje de Modelado UML

Es un lenguaje para la especificación, visualización, construcción y documentación de los productos de trabajo de un proceso de sistema intensivo. Es utilizado para la modelación de sistemas con características orientadas a objetos. Facilita a los integrantes de un equipo multidisciplinario participar e intercomunicarse fácilmente, promueve la reutilización e incrementa la capacidad de lo que se puede hacer con otros métodos de análisis y diseño orientados a objetos (Cornejo, 2011).

Visual Paradigm 8.0

Visual Paradigm es una herramienta UML profesional, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una construcción más rápida de aplicaciones de calidad, mejores y aún menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Cornejo, 2011).

Ventajas (Cornejo, 2011):

- ✓ Tiene disponible distintas versiones: Enterprise, Professional, Standard, Modeler, Personal y Community (que es gratuita).
- ✓ Se pueden dibujar todos los tipos de diagramas de clase, generar el código de diagramas y generar la documentación.
- ✓ Alta interoperabilidad: Los usuarios y proveedores de tecnología pueden integrar con Visual Paradimg modelos en sus soluciones con un mínimo esfuerzo.

1.8 Entorno de desarrollo integrado

Un entorno de desarrollo integrado, llamado también IDE (sigla en inglés de integrated development environment), es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios (Sánchez, 2014).

NetBeans v8.0

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE es un producto libre y gratuito sin restricciones de uso, además es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento, con cerca de 100 socios en todo el mundo. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal del proyecto. Algunas de las características de la aplicación son (NetBeans, 2016):

- ✓ Gestión de la interfaz de usuario (menús y barras de herramientas).
- ✓ Gestión de configuración de usuario.
- ✓ Gestión de almacenamiento (guardar o cargar algún tipo de dato).
- ✓ Gestión de ventana.
- ✓ Librería visual de Netbeans.

1.9 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es una colección de datos relacionados entre sí, estructurados y organizados, un conjunto de programas que acceden y gestionan esos datos. La colección de esos datos se denomina Base de Datos (BD). Los sistemas gestores de base de datos están diseñados para gestionar grandes bloques de información, que implica tanto la definición de estructuras para el almacenamiento como de mecanismos para la gestión de la información. El SGBD es una aplicación que permite a los usuarios definir, crear y mantener la DB y proporciona un acceso controlado a la misma (Alegsa, 2010).

MySQL 5.5

Es un sistema de gestión de bases de datos relacional, fue creada por la empresa sueca MySQL AB, la cual tiene el copyright del código fuente del servidor SQL, así como también de la marca.

MySQL es un software de código abierto, licenciado bajo la GPL de la GNU, aunque MySQL AB distribuye una versión comercial, en lo único que se diferencia de la versión libre, es en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de otra manera, se vulneraría la licencia GPL (Enríquez Toledo Alma, 2014).

En las últimas versiones se pueden destacar las siguientes características principales (Enríquez Toledo Alma, 2014):

- ✓ El principal objetivo de MySQL es velocidad y robustez. Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Gran portabilidad entre sistemas, puede trabajar en distintas plataformas y sistemas operativos.
- ✓ Cada base de datos cuenta con 3 archivos: uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla. Aprovecha la potencia de sistemas multiproceso, gracias a su implementación multihilo.
- ✓ Flexible sistema de contraseñas (passwords) y gestión de usuarios, con un muy buen nivel de seguridad en los datos.
- ✓ El servidor soporta mensajes de error en distintas lenguas

Entre las ventajas se encuentran (Enríquez Toledo Alma, 2014):

- ✓ Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- ✓ Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- ✓ Facilidad de configuración e instalación.
- ✓ Soporta gran variedad de Sistemas Operativos Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- ✓ Conectividad y seguridad.

1.10 Servidor de aplicaciones web

Apache 2.2

Es una tecnología de código de fuente abierta, puede ser usado en varios sistemas operativos, lo que lo hace prácticamente universal. Es un servidor altamente configurable de diseño modular, trabaja con gran cantidad de lenguajes ejemplos de estos: Perl, PHP y otros lenguajes de script. Apache es el servidor web hecho por excelencia, su configurabilidad, robustez y estabilidad hacen que cada vez millones de servidores reiteren su confianza en este programa (Alegsa, 2010).

Características (Alegsa, 2010):

- ✓ Multiplataforma.
- ✓ Apache es una tecnología gratuita de código de fuente abierta.
- ✓ Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor Web Apache. Actualmente existen muchos módulos para Apache que son adaptables a este, y están ahí para que se instalen cuando lo necesiten.
- ✓ Apache trabaja con Java y páginas JSP19. Teniendo todo el soporte que se necesita para tener páginas dinámicas.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- ✓ Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

1.11 Conclusiones parciales

En este capítulo se analizaron conceptos teóricos que permitieron entender el entorno en que se desarrolla la investigación. Luego de realizar un estudio del estado del arte de los Sistemas de Gestión de Información (SGI) a nivel mundial y nacional, se concluye que ninguno cubría todas las funcionalidades que requeriría una solución que resuelva el problema planteado en esta investigación. La selección de las herramientas, lenguajes y la metodología de desarrollo de software permitió contar con una base tecnológica y una guía apropiada para el desarrollo de la solución. Lo anterior tiene mucha importancia tomando en cuenta que el país está inmerso en el proceso de migración a software libre ya que las herramientas informáticas seleccionadas son de código abierto.

CAPÍTULO 2. ANÁLISIS Y DISEÑO

Introducción

En el capítulo se exponen las principales características del sistema a implementar. Se verán diferentes modelos, se expondrán los requisitos funcionales y no funcionales y las historias de usuarios relacionadas con la propuesta de solución. Además, en correspondencia con la etapa de planificación del sistema, se puntualizará el plan de entregas, el de iteraciones y las tareas ingenieriles. También se describe la arquitectura por la cual se regirá la construcción de la solución, los patrones de diseño y los artefactos generados para cumplir con la presente investigación.

2.1 Descripción general de la propuesta de solución

El sistema de gestión a desarrollar informatizará este proceso, permitiendo a los jefes de áreas de la Universidad y específicamente a la Dirección de Contabilidad y Finanzas de la UCI, tener un mejor manejo y por consiguiente mejor control de la información asociada al consumo de pasajes por *voucher* en divisa, este sistema podrá gestionar la información referente a los voucher, cheques, facturas y cuentas en un tiempo más corto y con mayor eficiencia, además el administrador va a tener total control sobre todos los usuarios porque este sistema registrará un historial con todas las acciones que estos realicen.

2.2 Modelo conceptual

Con la construcción del modelo conceptual se logra una mejor comprensión del dominio del problema. Un modelo conceptual no es más que una representación visual de los conceptos u objetos del mundo real que son significativos para el problema o el área que se analiza, representando las clases conceptuales, no los componentes de *software*. Puede verse como un modelo que comunica a los interesados, cuáles son los términos importantes y cómo se relacionan entre sí (Larman Craig, 1999).

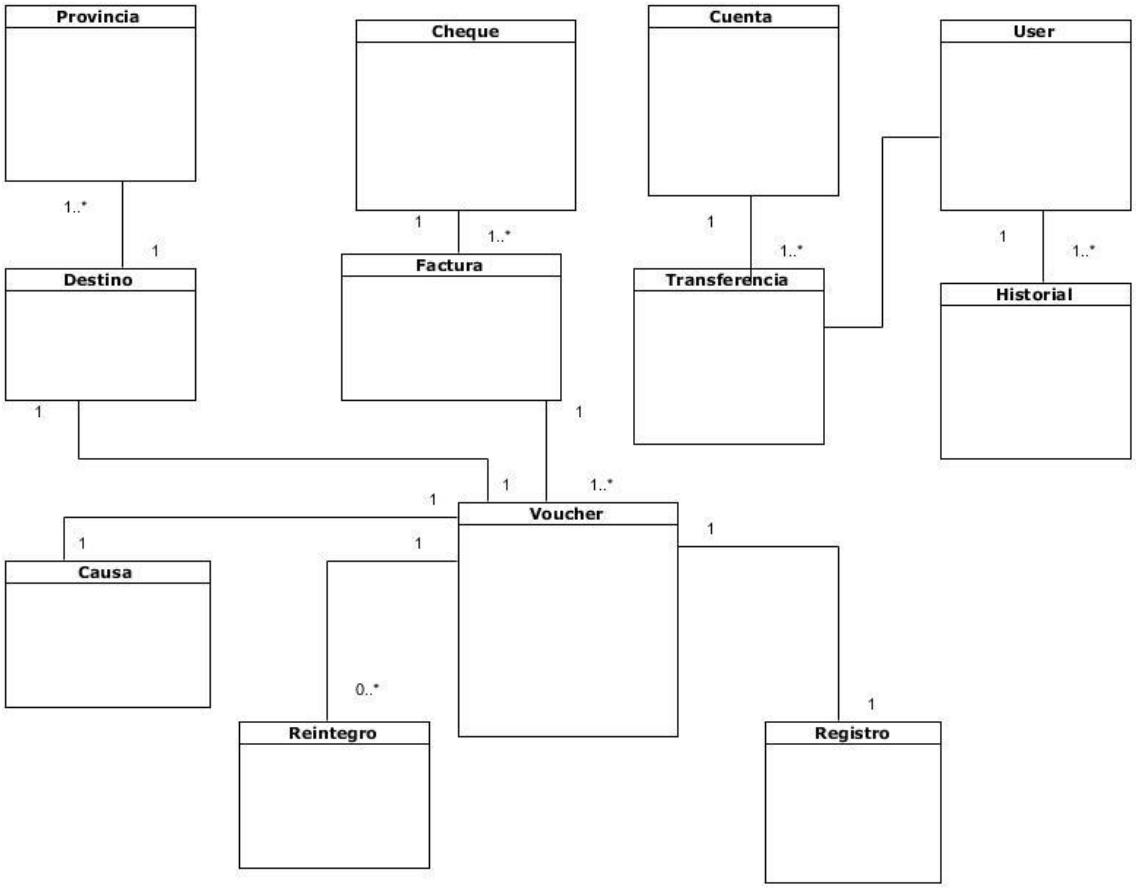


Figura 1: Modelo Conceptual. (Fuente: Elaboración propia)

***voucher*:** contiene los datos correspondientes a un *voucher*.

Causa: contiene los datos correspondientes a la causa por la que puede ser entregado un *voucher*.

Destino: contiene los datos correspondientes al destino para el que se autorizará el *voucher*.

Fondo Viazul: muestra el estado en el que se encuentra el pago por anticipado.

Cuenta: contiene los datos que muestran la cantidad de dinero destinada por la UCI para el consumo de pasajes en divisa por Viazul.

2.3 Requisitos del software

Los requisitos son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos suelen clasificarse en requisitos funcionales o no funcionales. Los funcionales declaran los servicios que debe brindar el sistema, la manera que éste debe reaccionar y funcionar

ante una entrada o situación en particular y los no funcionales son las restricciones de los servicios o funciones ofrecidas por el sistema. Los requisitos importantes para el desarrollo de un *software*, ya que su propósito fundamental es guiar el desarrollo hacia el sistema correcto (Almira Torres, 2012). Para el levantamiento de requisitos se emplean algunas técnicas como entrevistas, cuestionarios y tormentas de ideas. La técnica seleccionada para el levantamiento de los requisitos del sistema a desarrollar fue la entrevista, la cual se le realizó a al Director de Contabilidad y Finanzas de la UCI, así como a la mayoría de las especialistas que laboran en esa área.

Requisitos Funcionales

Los requisitos funcionales (RF) representan las funcionalidades que realizará el sistema y permiten modelar las diferentes operaciones para administrar una entidad de información. Los requisitos definidos son:

Tabla 2 Descripción de los RF. (Fuente: Elaboración propia)

No.	**Nombre del Requisito Funcional**	**Descripción**
Prioridad		**Alta**
RF1	Insertar usuario	Permite insertar usuarios los cuales tendrán acceso a la información del sistema teniendo en cuenta el rol del mismo.
RF2	Eliminar usuario	Permite eliminar usuarios.
RF3	Listar usuarios	Muestra la información general de los usuarios en forma de lista.
RF4	Asignar roles de usuario	Permite asignar un rol a un usuario según el privilegio que tenga el mismo al acceso de la información.
RF5	Modificar roles de usuario	Permite modificar un rol a un usuario según el privilegio que tenga el mismo al acceso de la información.

RF6	Autenticar usuario	Permite autenticar usuario una vez que este exista en el sistema.
RF7	Insertar cuenta	Permite insertar una cuenta al sistema cuando el usuario hace la solicitud.
RF8	Modificar cuenta	Permite modificar una cuenta del sistema si el usuario así lo desea.
RF9	Eliminar cuenta	Permite eliminar una cuenta del sistema si el usuario así lo desea.
RF10	Realizar transferencia entre cuentas	Permite hacer transferencias entre cuentas.
RF11	Insertar *voucher*	Permite insertar un *voucher* al sistema cuando el usuario hace la solicitud.
RF12	Eliminar *voucher*	Permite eliminar *voucher*.
RF13	Buscar *voucher* .	Permite hacer una búsqueda de los *voucher* ya creados.
RF14	Mostrar historial de acciones	Permite mostrar el historial de las acciones en tiempo real.
RF15	Registrar consumo del *voucher*	Permite mostrar la cantidad de veces que un usuario hace uso del *voucher*.
RF16	Listar *voucher* insertado	Muestra la información general de los *voucher* insertados en forma de lista.
RF17	Generar reporte por área	Permite mostrar reporte por área.
RF18	Generar reporte por destino	Permite mostrar reporte por destino.
RF19	Generar reporte por causa	Permite mostrar reporte por causa.
RF20	Generar reporte por fecha	Permite mostrar reporte por fecha.

RF21	Exportar reporte a excel	Permite exportar reporte a Excel.

Requisitos no Funcionales

- ✓ **Usabilidad**:

RNF-1. El sistema será dinámico y de fácil navegación, pues se realizará con campos y botones fáciles y sugerentes para los usuarios.

- ✓ **Confiabilidad**:

RNF-2. El sistema validará la entrada de datos para que no se inserte información errónea. La validación se realizará mediante reglas de validación.

- ✓ **Seguridad:**

RNF-3. Se garantizará la integridad y confidencialidad de la información mediante mecanismos de control de acceso no autorizados utilizando: usuario, contraseña y definiendo niveles de acceso para cada usuario, de manera que estos solo puedan tener disponible solamente las opciones relacionadas con su actividad y tenga datos de acceso propios.

RNF-4. Se podrá acceder a las páginas de administración del sitio *web* y las páginas de usuarios a través del protocolo https.

RNF-5. El sistema permitirá que cuando se borre cualquier información pueda existir una opción de advertencia antes de realizar la acción.

- ✓ **Apariencia o Interfaz externa:**

RNF-6. Se garantizará una correcta organización de la información para permitir una adecuada interpretación.

- ✓ **Interfaces de *Hardware*:**

RNF-7. Los requisitos mínimos para las PC clientes son el uso de una computadora Pentium 4 con acceso a la red UCI.

RFN-8. El servidor de BD debe ser *Core* 2duo, con 4GB de RAM y 500 GB de disco duro.

- ✓ **Interfaces *Software*:**

RNF-9. PC Servidor *web*:

- ✓ Sistema Operativo: CentOS 6.0 o superior,
- ✓ Servidor *web*: Apache v2.2 o superior,
- ✓ Lenguaje de programación: PHP 5.3 o superior.

RNF-10. PC Servidor de BD:

Sistema Operativo: CentOS 6.0 o superior,

Sistema Gestor de Base de Datos PostgreSQL v9.1.

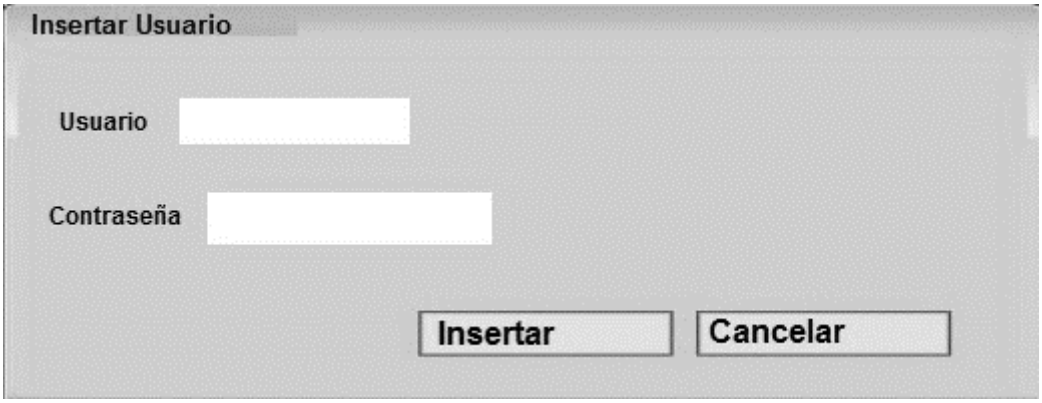
2.4 Historias de usuario

Las historias de usuario (HU) es la técnica utilizada en XP para especificar los requisitos funcionales 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.

Estas se usan 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 (Letelier, 2011). A continuación, se define una HU de la solución propuesta. Otras Historias de Usuario se encuentran en el Anexo.

Tabla 3 HU_1 Insertar Usuarios. (Fuente: Elaboración propia)

Historia de Usuario	
Número: HU_1	**Nombre Historia de Usuario:** Insertar Usuarios
Modificación de Historia de Usuario Número: *1*	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada: 1**
Prioridad en Negocio: Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 5 días.
Descripción: El sistema debe brindar la posibilidad al usuario de insertar usuario. La HU inicia cuando el usuario selecciona en el menú principal la opción “insertar usuario”. El sistema muestra una ventana con los campos usuarios y contraseña.	

La misma contará con los campos **Usuario**: el mismo admitirá letras y números con un tamaño definido entre 5-7 caracteres, **Contraseña:** el mismo admitirá letras y números con un tamaño definido entre 7-12, además esta vista cuenta con dos botones para realizar la acción ya sea de **Insertar** o **Cancelar** la operación que se está realizando. Se muestra un mensaje luego de haber realizado la modificación.
Observaciones: ✓ No puede existir un usuario con el mismo nombre y contraseña. ✓ Si el usuario no tiene acceso en el sistema debe aparecer un mensaje de error: "Usted no tiene permiso de acceso al sistema". ✓ El usuario debe autenticarse con usuario y contraseña de dominio. (LDAP).
Prototipo de interfaz de usuario: 

2.5 Planificación

La planificación en XP está basada en un conjunto de decisiones tomadas por el cliente junto al programador. Los clientes representan las necesidades propias del negocio y el programador define los requisitos técnicos que complementan las necesidades del negocio. Seguidamente se define el plan de entrega y el plan de iteraciones que regirán el desarrollo.

Plan de entrega

En esta fase, el cliente establece la prioridad de cada HU, 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 4 Plan de entrega. (Fuente: Elaboración propia)

Entregables	**Iteración**	**Fin de la iteración**
Gestionar Usuario	1	Marzo 2017
Gestionar Roles a Usuarios	1	Marzo 2017
Autenticar Usuario	1	Marzo 2017
Gestionar Cuenta	1	Marzo 2017
Gestionar *voucher*	1	Marzo 2017
Buscar *voucher*	1	Marzo 2017
Mostrar Historial de Acciones	1	Marzo 2017
Registrar consumo del *voucher*	2	Abril 2017
Mostrar *voucher* insertado	2	Abril 2017
Generar reportes	2	Mayo 2017
Exportar reportes a PDF	2	Mayo 2017
Exportar reportes a Excel	2	Mayo 2017

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 producción (Letelier, 2011). El sistema de gestión de pago de pasaje por *voucher* en divisa se realizará en 2 iteraciones como se demuestra en la siguiente tabla 5.

Tabla 5 Plan de iteraciones. (Fuente: Elaboración propia)

Iteración	**Historia de Usuario**	**Semanas estimadas**
1	Gestionar Usuario	7
	Gestionar Roles a Usuarios	
	Autenticar Usuario	
	Gestionar Cuenta	
	Gestionar *voucher*	
	Buscar *voucher*	
	Mostrar historial de acciones	

2	Registrar consumo del *voucher*	4
	Mostrar *voucher* insertado	
	Generar reportes	
	Exportar reportes a Excel	

2.6 Diseño

El papel del diseño en el ciclo de vida del *software* es adquirir conocimiento de su funcionamiento. Este 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 (Informática, 2013).

Tarjetas de Clase, Responsabilidad y Colaboración (CRC)

Las Tarjetas CRC (Clase, Responsabilidad y Colaboración) constituyen uno de los artefactos de la metodología XP que guía el proceso de desarrollo de la solución propuesta. Sirven para diseñar el sistema entre todo el equipo. Permiten reducir el modo de pensar procedural y apreciar la tecnología de objetos (Letelier, 2011).

Tabla 6 Tarjeta CRC correspondiente a la clase *voucher.* (Fuente: Elaboración propia)

Nombre de la Clase: ***voucher***	
Responsabilidades	**Colaboradores**
1. getCodigo	
2. getImporte	Cuenta
3. getArea	
4. getDirArea	
5. getFechaEmitido	
6. getFechaConsumido	
7. getCausa	Causa
8. getDestino	Destino
9. getEstado	

Tabla 7 Tarjeta CRC correspondiente a la clase Causa. (Fuente: Elaboración propia)

Nombre de la Clase: Causa	
Responsabilidades	**Colaboradores**

1. getId	
2. getNombre	
3. getDescripcion	

Tabla 8 Tarjeta CRC correspondiente a la clase Destino. (Fuente: Elaboración propia)

Nombre de la Clase: Destino	
Responsabilidades	**Colaboradores**
1. getId	
2. getMunicipio	
3. getPrecio	

Tabla 9 Tarjeta CRC correspondiente a la clase Cuenta. (Fuente: Elaboración propia)

Nombre de la Clase: Cuenta	
Responsabilidades	**Colaboradores**
1. getId	
2. getCuenta	
3. getFondo	

Tabla 10 Tarjeta CRC correspondiente a la clase Transferencia. (Fuente: Elaboración propia)

Nombre de la Clase: Transferencia	
Responsabilidades	**Colaboradores**
1. getId	
2. getCuenta	Cuenta
3. getImporte	
4. getFecha	

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**.**

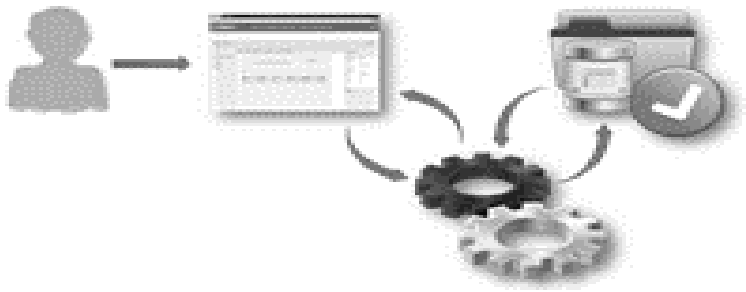


Figura 2: Modelo-Vista-Controlador (MVC) (Fuente: Documentación de Symfony) (Potencier, 2015).

Para el desarrollo del sistema de gestión de la información referente al control de pago de *voucher* de Viazul en divisa en la UCI se utiliza el patrón de arquitectura de *software* Modelo Vista Controlador (en inglés abreviado MVC). 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 mucho más flexible y sencillo de mantener. El marco de trabajo definido, en este caso Symfony, impone el uso del patrón arquitectónico MVC.

Controlador: el controlador soporta el flujo de información del módulo de control de acceso físico, interactuando con el modelo y la vista donde se introducirá el código de barra que identifica el usuario para su posterior acceso a la entidad. 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. La utilización de este patrón se evidencia en las clases *Vouchercontroller*.

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.

Patrones de diseño

Los patrones de diseño describen un problema particular en un contexto específico, para el cual presentan un esquema genérico probado para su solución, además influyen en la creación de un diseño para el sistema, dotado de flexibilidad y reusabilidad (Buchmann, 1996). A continuación, se describen los que fueron utilizados en el desarrollo para la asignación de responsabilidades:

Patrones generales

En el diseño orientado a objetos, es muy importante asignar las responsabilidades de forma correcta. Para ello se utilizan los patrones GRASP, los cuales promueven buenos principios para la realización de dicha tarea. Para el diseño de la propuesta que se desea implementar fueron empleados los siguientes:

Experto: se emplea en todas las clases del sistema debido a que cada una contiene la información necesaria para cumplir con su responsabilidad. El uso de este patrón trae consigo que se conserve el encapsulamiento debido a que las clases se valen de su propia información para cumplir con las tareas asignadas, soportando además un bajo acoplamiento.

Creador: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. Se pone de manifiesto en las clases controladoras del sistema, las cuales tienen dicha responsabilidad a su cuenta, además brinda soporte a un bajo acoplamiento y mejora la reutilización. Este patrón se observa en la clase *voucher*Controller.

Bajo acoplamiento: el diseño del sistema consta de este patrón pues cada clase depende lo menos posible de otra, de manera tal que una de ellas solo recurre a otra en caso de que exista referencia dentro de sus atributos, lo cual permite que el sistema sea mucho más robusto y de fácil mantenimiento. Se evidencia el uso de este patrón en la relación existente entre las clases Cuenta y *voucher*, la entidad Cuenta solo se relaciona con la entidad *voucher*, que a su vez es la que se encarga de ejercer acciones sobre esta.

Alta cohesión: el sistema está diseñado de forma que las clases tienen responsabilidades estrechamente relacionadas y no realizan un trabajo excesivo, lo cual permite simplificar el mantenimiento, generar el bajo acoplamiento y aumentar la capacidad de reutilización de las mismas.

Se evidencia el uso de este patrón en la relación existente entre las clases *voucher*, Causa y Destino, dado que la entidad *voucher* necesita de las entidades Causa y Destino para su funcionamiento.

Los patrones del diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de *software* y otros ámbitos referentes al diseño de interacción o interfaces.

2.7 Modelo de base de datos

El diseño de la base de datos es fundamental para el desarrollo de cualquier aplicación, debido a que describe la representación lógica y física de los datos persistentes del sistema. Seguidamente se muestra el diagrama de base de datos que genera el sistema a implementar.

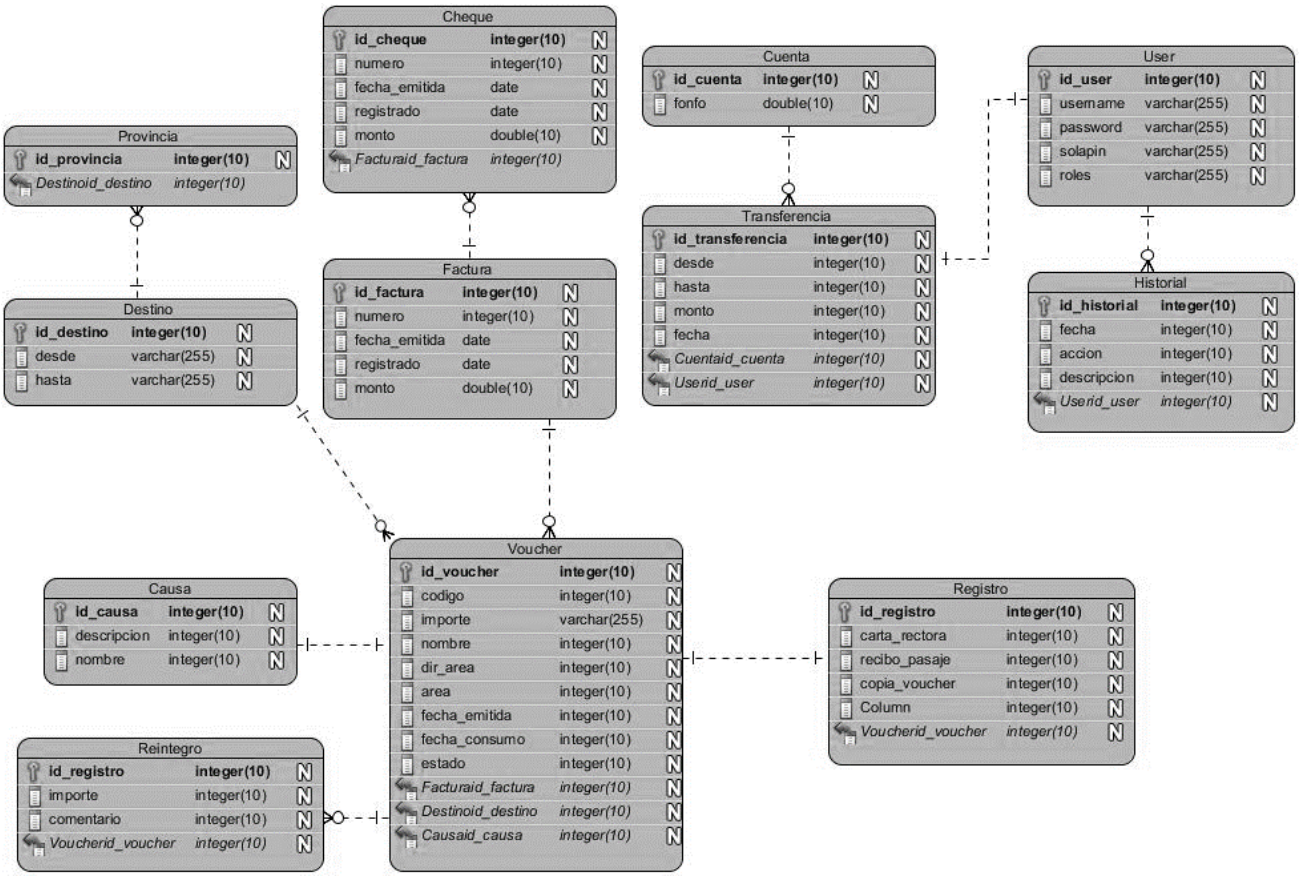


Figura 3 Diagrama de base de datos. (Fuente: Elaboración propia)

2.8 Conclusiones parciales

En el presente capítulo se inició el desarrollo de la propuesta de solución para el problema definido con perspectivas de darle cumplimiento al objetivo general planteado en el marco teórico de la

investigación. Se identificaron los requisitos funcionales y no funcionales con los que deberá contar el sistema. Se realizó la creación de los planes de entregas e iteraciones. Se definió la arquitectura y patrones de diseño a utilizar, además se generaron los artefactos necesarios para guiar el desarrollo del sistema tales como: historias de usuario (HU), modelo conceptual y modelo de los datos.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA

Introducción

En el presente capítulo se describe la implementación del *software*, fase donde se materializa el producto final y se cumple con los requisitos obtenidos al inicio de la investigación. Para ello se definen los estándares de codificación que debe cumplir el desarrollador, se crea el diagrama de componentes y el de despliegue, además de presentarse las principales interfaces de usuario de la solución. A partir del código resultante y su funcionamiento se ejecutan las pruebas de aceptación y de seguridad.

3.1 Plan Releases

Un plan release o plan de proyecto es un conjunto de historias de usuario agrupadas por versiones del producto, también conocido como plan de entregas. Le posibilita al dueño de producto y al equipo decidir cuánto se debe desarrollar y cuánto tiempo se tardará antes de tener un producto entregable. Sirve para realizar un análisis del progreso.

Tabla 11 Plan Releases. (Fuente: Elaboración propia)

Releases	**Descripción de la Iteración**	**Orden de Historia de Usuario a implementar**	**Duración total**
1ra Iteración	Agrupa las UH de prioridad Alta y Muy Alta.	Gestionar Usuario	7 semanas
		Gestionar Roles a Usuarios	
		Autenticar Usuario	
		Gestionar Cuenta	
		Gestionar *voucher*	
		Buscar *voucher*	
		Mostrar historial de acciones	
2da Iteración	Agrupa las HU de prioridad Media y Baja.	Registrar consumo del *voucher*	4 semanas
		Mostrar *voucher* insertado	
		Generar reportes	
		Exportar reportes a PDF	
		Exportar reportes a Excel	

3.2 Estándares de codificación

Es necesario establecer un criterio fijo que proporcione reglas para la creación de nombres para variables y métodos, permitiendo una mejor lectura del *software*, y un mejor entendimiento del código por parte de los desarrolladores. Las pautas fundamentales definidas para la implementación son las siguientes:

Estructura

- ✓ Agrega un único espacio después de cada delimitador coma,
- ✓ No utilizar espacios después de la apertura de un paréntesis y antes del cierre del mismo,
- ✓ Agregar un único espacio alrededor de operadores (== , &&),
- ✓ Agregar un único espacio antes de los paréntesis de apertura de una palabra clave de control (*if, else, for, while,* . .),
- ✓ Agrega una línea en blanco antes de la sentencia *return,*
- ✓ No agregar espacios al final de las líneas,
- ✓ Utiliza llaves para indicar el cuerpo de las estructuras de control sin importar el número de sentencias que éstas contengan,
- ✓ Coloca las llaves en sus propias líneas para clases, métodos y declaración de funciones,
- ✓ Separa las sentencias condicionales y las llaves de apertura con un único espacio sin dejar una línea en blanco,
- ✓ Declara explícitamente la visibilidad de clases, métodos y propiedades,
- ✓ Utiliza constantes de tipo PHP nativas en minúsculas: *false, true* y *null*. Lo mismo aplica para *array(),*
- ✓ Utiliza letras mayúsculas para constantes, con palabras separadas por guiones bajos,
- ✓ Define una clase por archivo,
- ✓ Declara las propiedades de las clases antes de los métodos,
- ✓ Declara los métodos públicos primero, luego los protegidos y finalmente los privados.

Convención de Nombres

- ✓ Utiliza *camelCase* y no guiones bajos, para variables, funciones y nombres de métodos,
- ✓ Utiliza guiones bajos para definir opciones, argumentos y nombres de parámetros,
- ✓ Utiliza los *namespace* para todas las clases,
- ✓ Utiliza *Symfony* como el *namespace* de primer nivel,
- ✓ Añade como sufijo *Interface* a las interfaces,

- ✓ Utiliza caracteres alfanuméricos y guiones bajos para nombres de archivos.

3.3 Diagrama de despliegue

El diagrama de despliegue es utilizado para capturar los elementos de configuración del procesamiento, las conexiones entre esos elementos y visualizar la distribución de los componentes de *software* en los nodos físicos. Entre los nodos existen relaciones que representan los protocolos de comunicación que se utilizan para acceder a cada uno (Informática, 2013).

En la siguiente figura puede visualizarse el diagrama de despliegue definido para la solución propuesta. La PC Cliente representa las estaciones de trabajo de los usuarios que se conectan al sistema, las cuales realizan peticiones al Servidor Web mediante el protocolo HTTPS. Este servidor establecerá una conexión mediante el protocolo TCP\IP al servidor de base de datos.

Figura 4 Diagrama de Despliegue. (Fuente: Elaboración propia)

Descripción del Diagrama de Despliegue.

- ✓ **PC Cliente:** ayuda a los usuarios a consultar y actualizar la información que se encuentra en el Servidor *Web* utilizando protocolos de comunicación HTTPS.
- ✓ **Servidor *Web* (Apache):** se ubican íntegramente las capas de presentación, lógica del negocio y de acceso a datos del sistema, así como los servicios que se brindan.
- ✓ **Servidor de base de datos (PostgreSQL): s**e almacenan todos los datos que son consultados y actualizados por los usuarios del sistema de gestión de la Información para el control de la venta de pasajes en divisa por Viazul y garantizará el acceso a ellos a través del Servidor *Web*.

3.4 Interfaces de usuarios

La interfaz de usuario es el medio con el que un usuario puede comunicarse con una máquina, un equipo o una computadora, y comprende todos los puntos de contacto entre el usuario y el equipo. Normalmente suelen ser fáciles de entender y fáciles de accionar. A continuación, se muestran algunas interfaces del sistema.

A través de la interfaz principal, el usuario podrá acceder a la información y realizar las funcionalidades para las que fue creado el sistema. Cuenta con dos barras superiores que muestra la información más relevante almacenada en el sistema, cantidad de Cheques, *voucher*, Facturas, Cuentas y Usuarios creados hasta el momento y el número de Transferencias que ha hecho el sistema, y en la parte inferior cuenta con tres gráficas que muestran: Las últimas transferencias, Cantidad de cheques por año y el Monto de cheques por año. En el lateral izquierdo cuenta con el Menú Principal para acceder a las funcionalidades del sistema, forman parte de este menú principal el botón "*voucher*" para acceder a la gestión de estos; "Cheques" para la gestión de los cheques; "Facturas" para la gestión de las facturas; "Cuentas" para la gestión de las cuentas; "Historial" para tener control de todas las acciones que realiza el sistema; "Configurar" para añadir usuarios, destinos y causas; "Reportes" para emitir los reportes y exportarlos.

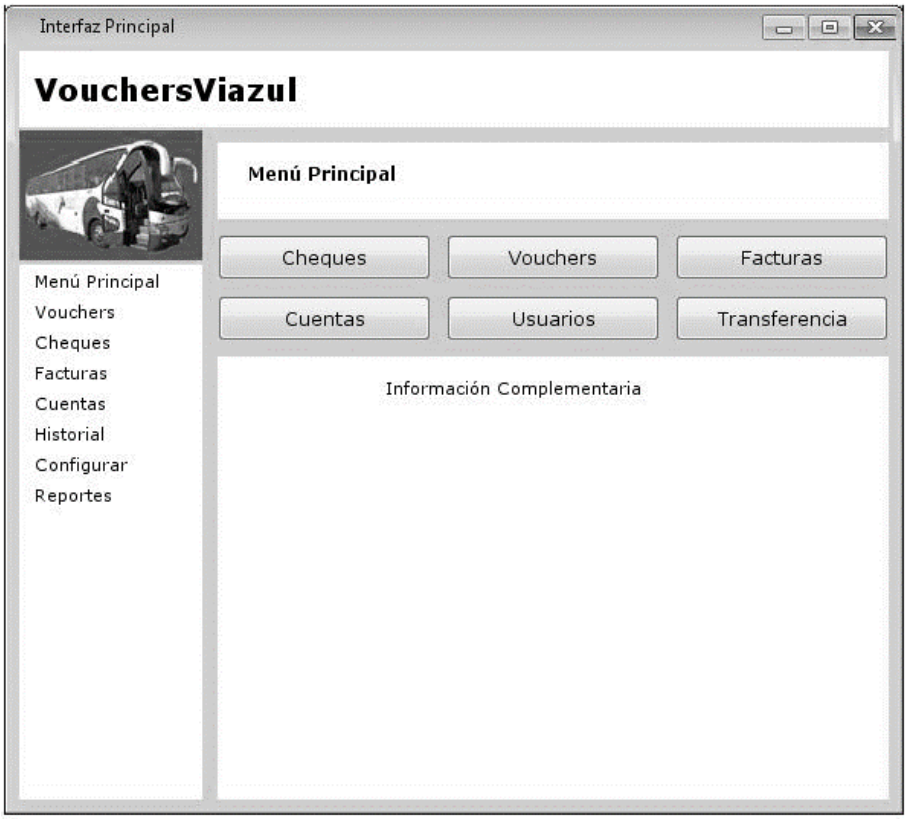


Figura 5 PIU Interfaz principal del sistema. (Fuente: Elaboración propia)

Mediante la interfaz de usuario "Historial" puede tener un control total de las acciones que se realizan en el sistema. En el listado que muestra el historial estarán contenido la fecha, el nombre del usuario, la acción que realizó y además contará con una pequeña descripción de cada una de estas acciones.

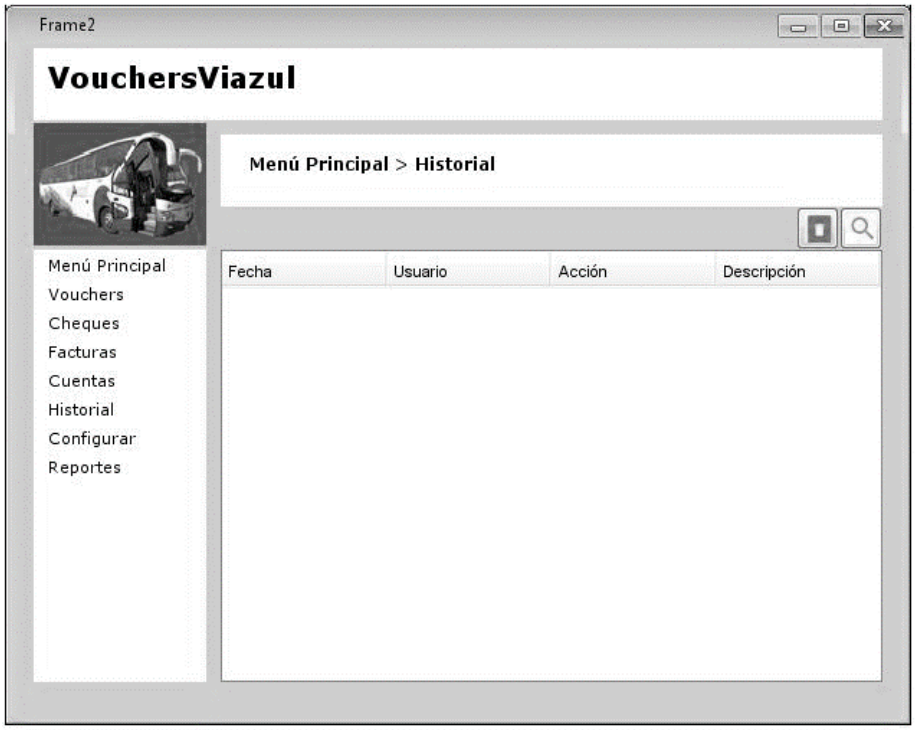


Figura 6 PIU Interfaz Historial. (Fuente: Elaboración propia)

Accediendo al botón “reportes” y a través de filtros, los cuales pueden ser área, destino, causa y un rango de fecha se podrá obtener un reporte de los *voucher* con la información especificada. Además se podrá exportar la información resultante en formato excel.

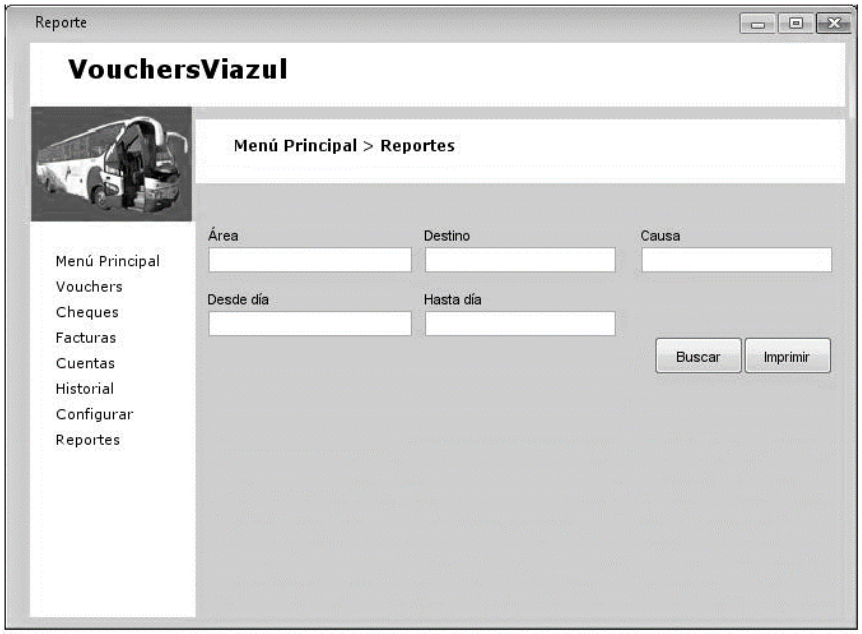


Figura 7 PIU Interfaz Reporte. (Fuente: Elaboración propia)

3.5 Pruebas

Las pruebas son un proceso de ejecución de un programa con la intención de descubrir errores. Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces. Una prueba tiene éxito si descubre un error aún no detectado.

Las pruebas de software son aquellos procedimientos que se realizan para verificar la calidad de un producto de software y pueden ser aplicadas periódicamente. Estas tienen como objetivo fundamental la identificación de posibles errores, además representan una revisión final de las especificaciones del diseño y de la codificación. La metodología XP establece dos tipos de pruebas: pruebas unitarias y pruebas de aceptación o funcionales, además se realizarán pruebas de seguridad (PRESSMAN, 2007).

Estrategia de pruebas funcionales

Roger S. Pressman plantea que una estrategia de prueba del *software* integra los métodos de diseño de caso de pruebas del sistema en una serie bien planeada de pasos que desembocará en la eficaz construcción de *software*. Esta proporciona un mapa que describe los pasos que se darán como parte de la prueba, indica cuándo se planean y cuándo se dan estos pasos, además de cuánto esfuerzo, tiempo y recursos consumirán. Una estrategia de prueba debe ser lo suficientemente flexible como para promover un enfoque personalizado. Al mismo tiempo, debe ser lo adecuadamente rígida como para

promover una planeación razonable y un seguimiento administrativo del avance del producto (PRESSMAN, 2007).

La estrategia de prueba para este sistema se enfoca en demostrar que el *software* es funcional y seguro, tiene como objetivo detectar errores que interfieran en el exitoso funcionamiento del mismo y corregirlos. Para realizar las pruebas se utilizarán el nivel de prueba de aceptación y nivel de prueba de sistema, empleando el tipo de prueba de funcionalidad. El método de pruebas que se empleará es el método basado en caja negra: se refiere a las pruebas que se llevan a cabo sobre la interfaz del *software*. O sea, los casos de prueba pretenden demostrar que las funciones del *software* son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, así como que la integridad de la información se mantiene (PRESSMAN, 2007).Los casos de prueba de caja negra soportado por la técnica de partición equivalente que es la más utilizada para este método pretenden:

- ✓ Demostrar que las funciones del *software* son operativas,
- ✓ Que las entradas se aceptan de la forma adecuada y que se produce el resultado correcto,
- ✓ Así como que la integración de la información externa (por ejemplo, archivos de datos) se mantiene.

A la aplicación *web* no se le realizaron las pruebas empleando el método de caja blanca debido a que la mayoría de los requisitos funcionales son gestionar y los métodos correspondientes a la implementación de dichos requisitos no tienen un nivel de complejidad alto como para aplicar dicho método de prueba (PRESSMAN, 2007).

Pruebas de funcionalidad

Las pruebas de funcionalidad o de aceptación son un tipo de prueba de caja negra orientadas a evaluar las distintas tareas en las que ha sido dividida una historia de usuario. Para asegurar el funcionamiento final de una determinada historia, las pruebas son creadas y usadas por los clientes para comprobar que el producto sí satisface los requisitos del usuario, tal y como se describe en las especificaciones de los requisitos. Verifica que el producto se comporta como se describe en las especificaciones funcionales del diseño.

A continuación, se describe el caso de prueba para la HU_Adicionar *voucher* compuesto por un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. Siempre es ejecutado como una unidad, desde el comienzo hasta el final.

Tabla 12 Caso de Prueba: Gestionar *Voucher*_Adicionar. (Fuente: Elaboración propia)

Caso de Prueba de Aceptación	
Código Caso de Prueba: SGIVPD-1.1	**Nombre Historia de Usuario:** Gestionar *Voucher*.
Nombre de la persona que realiza la prueba: Liosbel Margolles Clausell	
Descripción de la prueba: Se crea un *voucher* en el cual los campos requeridos son: nombre, área, director del área, causa, destino, fecha de emisión y estado.	
Condiciones de ejecución: Luego de autenticado el usuario, este podrá acceder a una interfaz que le permita crear un *voucher*.	
Entrada/Pasos de ejecución: Una vez dentro del sistema en el menú lateral izquierdo, se escoge la opción *voucher*, luego selecciona la opción de adicionar y seguidamente se llenan todos los campos requeridos del *voucher* a crear, los campos obligatorios son: nombre, área, director del área, causa, destino y fecha de emisión. Se debe asegurar que ningún campo quede en blanco y la fecha no debe exceder el día en que se emite el *voucher*. Luego se presiona el botón aceptar.	
Resultado esperado: El sistema muestra un listado donde se muestra el *voucher* creado.	
Evaluación de la prueba: Satisfactoria.	

Tabla 13 Caso de Prueba: Gestionar *voucher*_Editar. (Fuente: Elaboración propia)

Caso de Prueba de Aceptación	
Código Caso de Prueba: SGIVPD-1.2	**Nombre Historia de Usuario:** Gestionar *voucher*.
Nombre de la persona que realiza la prueba: Liosbel Margolles Clausell	
Descripción de la prueba: Se edita un *voucher* y en el proceso se verifica que todos los campos requeridos se llenen correctamente.	
Condiciones de ejecución: Luego de autenticado el usuario, este podrá acceder a una interfaz que le permita editar un *voucher*.	
Entrada/Pasos de ejecución: En el menú lateral izquierdo, se escoge la opción *voucher* y se selecciona el *voucher* a editar. Seguidamente se modifican los campos deseados y se debe asegurar que el campo fecha no exceda la fecha en que se modifica. Luego se selecciona la opción aceptar.	
Resultado esperado: El sistema muestra el listado de *voucher* donde se incluye el editado.	
Evaluación de la prueba: Satisfactoria.	

Tabla 14 Caso de Prueba: Gestionar *voucher*_Eliminar. (Fuente: Elaboración propia)

Caso de Prueba de Aceptación

Código Caso de Prueba: SGIVPD-1.3	**Nombre Historia de Usuario:** Gestionar *voucher*.
Nombre de la persona que realiza la prueba: Liosbel Margolles Clausell	
Descripción de la prueba: Se elimina un *voucher*.	
Condiciones de ejecución: Luego de autenticado el usuario, este podrá acceder a una interfaz que le permita eliminar un *voucher*.	
Entrada/Pasos de ejecución: En el menú lateral izquierdo, se escoge la opción *voucher* y se selecciona el *voucher* a eliminar, seguidamente el sistema muestra un mensaje para confirmar la eliminación del mismo. Luego se selecciona la opción sí.	
Resultado esperado: El sistema muestra el listado de un *voucher* sin el eliminado.	
Evaluación de la prueba: Satisfactoria.	

Luego de realizados un conjunto de casos de prueba se detectaron 10 no conformidades en la primera iteración que causaban que el sistema no se comportara de la forma esperada, de las cuales 6 eran errores ortográficos y 4 de validación. De estas no conformidades se solucionaron las 10. En una segunda iteración se encontraron 6 no conformidades, resolviéndose las 6. Para una tercera iteración se encontraron 0 no conformidades, obteniéndose un resultado satisfactorio para cada una de las combinaciones de datos por escenario. Los datos conformes a estas pruebas se muestran en la siguiente gráfica:

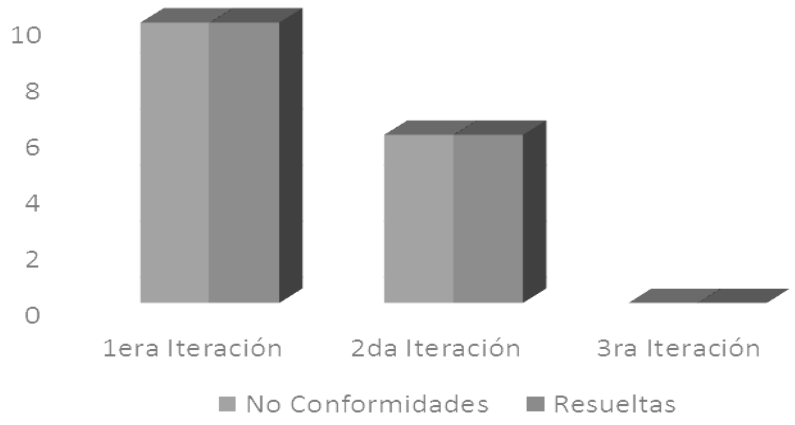


Figura 8 Gráfica de resultado de las pruebas de funcionalidad. (Fuente: Elaboración propia)

Pruebas de Seguridad

Estrategia de Prueba de Seguridad

Estas pruebas se enfocan en garantizar la seguridad (confidencialidad, integridad y disponibilidad) de la información que gestiona el sistema y tiene como objetivo detectar errores que pongan en riesgo la información y el funcionamiento efectivo del sistema, evaluar el riesgo y corregir las deficiencias encontradas. Los principales elementos a tener en cuenta durante la aplicación de estas pruebas son: el control de acceso garantizando que el mismo esté basado en la asignación de los distintos roles definidos para acceder al sistema y a la información, previendo que solo los usuarios que posean rol de administrador pueden acceder a la aplicación y realizar las operaciones que para cada uno de ellos están definidas sobre la información sensible; la protección contra ataques de inyecciones SQL, el desbordamiento de búfer, los ataque de denegación de servicios y la transmisión de datos sensibles sin cifrar.

Para probar la seguridad del sistema se aplicaron en una secuencia de 3 iteraciones una serie de pruebas usando las herramientas Acunetix[11] v8.0 y WebSecurity las cuales realizan un escaneo del sistema para luego generar reportes de las vulnerabilidades de seguridad detectadas. Las vulnerabilidades se acompañan de una descripción detallada de la localización, causa y efecto de las mismas, además de estar clasificadas por el impacto que pudieran tener en cuanto a los niveles de seguridad, la clasificación se realiza en alta, media, baja e informativa en el caso de la herramienta Acunetix, además se realizaron una serie de pruebas las cuales se presentan a continuación:

- Ataques de inyección.
- *Cross-Site Scripting (XSS).*
- Falsificación de petición (CSRF).

En una primera iteración Acunetix identificó 12 vulnerabilidades entre las que se encuentran y destacan: formularios HTML sin la protección CSRF, envío de información sensible en texto plano, la activación del método TRACE, enlaces rotos y el tipo de contraseña de entrada con autocompletado habilitado. De las vulnerabilidades detectadas se encontraron 5 medias, 4 bajas y 3 informativas. Por su parte la herramienta *Web* Securify arrojó 4 vulnerabilidades: divulgación del banner, auto completamiento activado, formularios HTML sin la protección CSRF, envío de información sensible en texto plano.

Luego de corregidos todos los fallos anteriormente detectados se realizó una segunda iteración donde se detectó que aún persistían vulnerabilidades pues Acunetix indicó la existencia de enlaces rotos y el

11 Acunetix v8.0: Herramienta multiplataforma que se utiliza para realizar pruebas de seguridad a aplicaciones web.

tipo de contraseña de entrada con autocompletado habilitado. En cuanto Web informó la existencia de enlaces rotos.

Se aplicó una tercera iteración donde ninguna de las herramientas utilizadas para el escaneo informó la existencia de vulnerabilidades, quedando garantizada la seguridad del sistema.

De manera general los resultados obtenidos se presentan en la tabla 15 y 16.

Vulnerabilidades encontradas

Tabla 15 Vulnerabilidades del entorno (Fuente: Elaboración propia)

Tipo	Cantidad	Descripción	Recomendaciones
Software	2	La versión de PHP instalada contiene vulnerabilidades que han sido solucionadas en versiones más recientes.	Actualizar a la versión 5.6.4 o superior.
Configuración	8	La configuración del servidor permite la captación de información sensible por el atacante.	Esta vulnerabilidad aunque no permite al atacante tomar el control acerca de la información a obtener puede proporcionar al atacante información sensible que puede utilizar para realizar ataques más específicos, pueden solucionarse limitando el acceso a los archivos del servidor y los elementos de la configuración ajustando el "expose_php" en “off” en el archivo php.ini.

Tabla 16 Vulnerabilidades del sistema (Fuente: Elaboración propia)

Tipo	Cantidad	Descripción	Recomendaciones
Falso positivo	2	El sistema permite la inyección de código.	Esta vulnerabilidad no puede ser utilizada por el atacante para alterar el

			sistema ni obtener información por lo que se denomina falso positivo.

Todas las vulnerabilidades fueron resueltas en el servidor y se recomienda que se tenga en cuenta la nueva configuración en el servidor para un futuro despliegue del sistema, llegando a la conclusión que el sistema desarrollado es seguro y está en condiciones de ser usado por el cliente con el propósito de agilizar en tiempo la gestión de la información de voucher como forma de pago.

3.6 Validación de las variables de investigación

La validación de las variables de la investigación va a permitir conocer y valorar la efectividad de la solución desarrollada en cuanto a agilidad y seguridad en la manipulación de la información contable de la venta de pasajes en divisa en la UCI, para realizar dicha validación se empleó el Método Experimental que consistió en realizar una comparación de cómo se realizaba el proceso antes de ser informatizado y como se realiza luego de automatizarlo.

La manipulación de la información antes de informatizarse el proceso se realizaba de forma manual lo que incurría en demoras para manejar la información, específicamente durante la emisión de reportes. Los reportes pueden ser realizados teniendo en cuenta varios elementos haciendo difícil la convergencia del resultado de su análisis. Anteriormente se dependía de las conciliaciones pactadas entre la UCI y Viazul para conocer el estado en que se encontraban los fondos depositados anticipadamente por la UCI en Viazul, que solían ser trimestralmente.

A partir de la informatización del proceso se tiene un mejor y más centralizado manejo de la información, esta es almacenada de forma más entendible, organizada, centralizada e integrada durante su gestión, que permite que la misma sea manejada mediante funcionalidades que simplifican su entendimiento y control, haciendo por tanto más rápido y eficiente la emisión de reportes. Es muy fácil mediante el sistema tener conocimiento del estado en que se encuentran las cuentas referentes al consumo de este servicio, entiéndase por cuentas la suma de dinero destinada y concebida en el presupuesto de la UCI para transportación en divisa y la depositada por anticipo en Viazul. El sistema registra cada transferencia que se realiza entre las cuentas y automáticamente que se consume un *voucher* se realiza el descuento del fondo de Viazul, manteniendo actualizado el estado del mismo.

Con la implantación del sistema para la gestión de la información del control de la venta de pasajes en divisa en la UCI se garantiza que el acceso a la información contable sea controlado, solo las personas

autorizadas podrán acceder a la información y de acuerdo a el rol que desempeñan podrán tener permisos de modificación de los datos.

A continuación, se muestra una tabla comparativa del tiempo que demoraba en realizarse un reporte de la cantidad de *voucher* consumido por un área en un trimestre antes de ser informatizado el proceso y luego de ser informatizado. La información empleada para realizar el experimento se obtuvo a partir de una entrevista realizada a los especialistas en contabilidad y finanzas de la Dirección de Contabilidad y Finanzas.

Tabla 17 Validación de la variable agilizar. (Fuente: Elaboración propia)

Tabla Comparativa para realizar reporte de la cantidad consumo en un trimestre de un área					
Actividades Tiempo	**Seleccionar los *voucher* en estado consumido**	**Seleccionar los *voucher* en el rango de fechas**	**Seleccionar los *voucher* del área especificada**	**Sumar la cantidad de *voucher* consumidos**	**Tiempo total para realizar el reporte**
Antes	1 hora	2 horas	45 min	15 min	4 horas laborables
Después	1 min	1 min	1 min	1 min	4 min

3.7 Conclusiones Parciales

En este capítulo se realizaron las pruebas a la aplicación que tienen como objetivo descubrir errores. Para conseguir este objetivo se planificaron y se ejecutaron una serie de pasos que van revisando todos los elementos del *software*. La etapa de prueba refleja la calidad con que ha sido llevada a cabo la proyección del sistema permitiendo: Encontrar y documentar los defectos que puedan afectar la calidad del *software*. Validar que el *software* trabaje como fue diseñado. Validar y probar en forma de demostración los requisitos que debe cumplir el *software*. Validar que los requisitos fueron implementados correctamente. Se realizaron las pruebas al sistema con resultados satisfactorios que demuestran que la aplicación cuenta con las características y funcionalidades para las que fue concebido.

CONCLUSIONES GENERALES

La investigación desarrollada y los resultados obtenidos permiten al autor plantear las siguientes conclusiones:

- ✓ La búsqueda y análisis de sistemas similares demostró que las herramientas internacionales y nacionales utilizadas para la gestión de la información referente a la gestión económica no poseen mecanismos para controlar la información referente a la utilización de *voucher* como forma de pago. Además, estos sistemas se encuentran desarrollados sobre tecnologías privativas. Lo anterior evidenció que no existe un sistema que permita gestionar la información del control de pagos de *voucher* para la confección de un expediente contable que gestione dicha información basado en tecnologías libres.
- ✓ El estudio de los sistemas y los conceptos teóricos permitió sentar las bases teóricas para el manejo de la documentación generada por la venta de pasajes en divisa y evidenció la necesidad de tener en cuenta varios elementos que permiten mejorar la calidad de los documentos, a través de diseños más adaptables y seguros que incorporan medidas de seguridad.
- ✓ El enfoque ágil propuesto por la metodología XP, el uso de tecnologías y herramientas libres, permitieron realizar el diseño de un sistema que gestiona la información que genera la utilización de *voucher* en concordancia con las especificaciones del cliente.
- ✓ La selección del estándar de codificación permitió la implementación del sistema garantizando el cumplimento de normas y reglas de implementación.
- ✓ La realización de las pruebas aceptación, seguridad y de rendimiento permitió comprobar el correcto funcionamiento del sistema desarrollado, además demostró su validez con resultados satisfactorios.

RECOMENDACIONES

Luego de concluido el desarrollo de la solución se recomienda integrar el sistema desarrollado al sistema Asset, utilizado por la UCI para el control de la contabilidad. Además de incorporar una funcionalidad al sistema propuesto que notifique a las áreas que solicita el servicio de *voucher* la veracidad de sus datos.

REFERENCIAS BIBLIOGRÁFICAS

Alegsa, Leandro. 2010. DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA. . [En línea] 12 de octubre de 2010. [Citado el: 5 de noviembre de 2016.] http://www.alegsa.com.ar/Dic/sgbd.php.

Almira Torres, Liuba. 2012. *Intranet del Ministerio de la Informática y las Comunicaciones.* La Habana : s.n., 2012.

Álvarez, Miguel Angel. 2012. Desarrolloweb. [En línea] 19 de septiembre de 2012. [Citado el: 3 de marzo de 2016.]

Bolivariana, Universidad Unión. 2009. Programación Extrema (XP). 2009.

Bragg, Steven. 2014. AccountingTools. [En línea] 22 de Diciembre de 2014. [Citado el: 10 de Mayo de 2017.] http://www.accountingtools.com/questions-and-answers/what-is-a-voucher.html.

Buchmann. 1996. *A System of Patterns.* 1996.

Buytaert, Dries. 2011. Drupal Groups. [En línea] 25 de mayo de 2011. [Citado el: 27 de noviembre de 2016.] https://groups.drupal.org/node/148379.

Chagoya, Ena Ramos. 2012. Gestiopolis.WebProfit Ltda. [En línea] 2012. [Citado el: 12 de octubre de 2016.] http://www.gestiopolis.com/metodos-y-tecnicas-de-investigacion.

ContaPyme. 2014. InSoft – Ingeniería de Software. [En línea] 2014. [Citado el: 15 de enero de 2017.] http://www.contapyme.com/.

Cornejo, José Enrique González. 2011. docIRS/¿Qué es UML? [En línea] 2011. [Citado el: 16 de noviembre de 2016.] http://www.docirs.cl/uml.htm. .

Definicion, Via. 2015. Via Definicion. [En línea] 2015. [Citado el: 15 de enero de 2017.] http://definicion.mx/gestion/.

DESOFT. 2015. Las comunicaciones al servicio de la sociedad. [En línea] 25 de febrero de 2015. [Citado el: 25 de noviembre de 2016.] http://www.mincom.gob.cu/?q=node/821.

Diccionario Informático. 2016. [En línea] 2016. [Citado el: 25 de octubre de 2016.] http://www.lawebdelprogramador.com/diccionario/buscar.php?opc=1&charSearch=dise%C3%B1o.

Enríquez Toledo Alma, Maldonado Ayala Jesús,Nakamura Ortega Yunko,Nogueron Toledo Goretty. 2014. Gridmorelos. *MySQL.* [En línea] 30 de septiembre de 2014. [Citado el: 27 de noviembre de 2016.] www.gridmorelos.uaem.mx/~mcruz/cursos/miic/MySQL.pdf.

Gaceta Oficial, LA REPÚBLICA DE CUBA MINISTERIO DE JUSTICIA. 2011. Gaceta Oficial DE LA REPÚBLICA DE CUBA MINISTERIO DE JUSTICIA. [En línea] 21 de noviembre de 2011. [Citado el: 10 de mayo de 2017.] http://www.gacetaoficial.cu/.

García, Alién. 2014. Estrategia metodológica para la elaboración y utilización de objetos de aprendizaje interactivos y experimentales en el proceso de enseñanza - aprendizaje de la Matemática Discreta en la UCI. *Tesis presentada en opción al título académico de Máster en Ciencias Matemáticas. Mención: Enseñanza de la Matemática.* 2014.

Graus, Significados. 2013. Significados. [En línea] 2013. [Citado el: 15 de enero de 2017.] https://www.significados.com/gestion/.

Informática Hoy. 2014. Informatica Hoy. [En línea] 2014. [Citado el: 15 de enero de 2017.] http://www.informatica-hoy.com.ar/sap/Que-es-SAP.php.

Informática, Universidad de las Ciencias. 2013. Entorno Virtual de Aprendizaje. [En línea] 2013. Http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_1_Procesos/Metodologias/Metodolgias_agiles_en_el_desarrollo_de_software.pdf.

Ing. Tamara, Sánchez Rodríguez y González Fernández, Ing. Mairelys. 2011. *Cedrux a la vuelta de la esquina.* 2011.

Larman Craig, Uml y Patrones. 1999. *Introducción al análisis y diseño orientado a objetos.* s.l. : s.l. : PRENTICE HAL, 1999. 1999. 970-17-0261-1.

Letelier, Patricio y Penadés, Mª Carmen. 2011. *Métodologías ágiles para el desarrollo de software:eXtreme Programming (XP.* España : Universidad Politécnica de Valencia : s.n., 2011.

Mozilla. 2016. Mozilla. [En línea] 2016. [Citado el: 5 de noviembre de 2016.] https://www.mozilla.org/en-US/foundation/.

My PHP. 2016. The PHP Group. [En línea] 2016. http://php.net/manual/es/intro-whatis.php.

NetBeans. 2016. NetBeans. [En línea] 2016. [Citado el: 5 de noviembre de 2016.] https://netbeans.org/about/history.html.https://netbeans.org/about/history.html.

Pérez, Julián, Merino Porto, Maria. 2015. Definicion de un voucher. [En línea] 2015. [Citado el: 5 de mayo de 2017.] http://definicion.de/voucher/.

Potencier, Fabien. 2015. LibrosWeb. [En línea] 2015. [Citado el: 12 de febrero de 2017.] http://librosweb.es/libro/symfony_1_2/capitulo_1/symfony_en_pocas_palabras.html.

—. 2011. SensioLabs. [En línea] octubre de 2011. http://fabien.potencier.org/what-is-symfony2.html.

Press, Cambridge University. 2015. Cambridge Advanced Learner's Dictionary & Thesaurus . [En línea] 2015. [Citado el: 10 de Mayo de 2017.] http://dictionary.cambridge.org/dictionary/english/voucher.

PRESSMAN. 2007. *Ingeniería del software: un enfoque práctico.* s.l. : [ed.] Mikel Angoar. s.l. : R.S, 2007.

Real Academia Española. 2016. Real Academia Española. [En línea] 2016. [Citado el: 20 de octubre de 2016.] http://dle.rae.es/.

—. 2016. Real Academia Española. *2016.* [En línea] 2016. [Citado el: 20 de octubre de 2016.] http://dle.rae.es/.

Ruiz, Alfredo. 2017. Postgrado en Comunicación y Marketing Digital. [En línea] 2017. [Citado el: 15 de enero de 2017.] http://www.postgradomarketingonline.com/blog/cuales-son-las-caracteristicas-principales-del-html/.

Salvador, Juan. 2015. Metodologías Ágiles AUP. [En línea] 2015. http://www.academia.edu/7894130/METODOLOGIAS_AGILES_AUP.

Sánchez, Alonso. 2014. Prezi. [En línea] 2014. [Citado el: 5 de noviembre de 2016.] https://prezi.com/7wmx8_d6ertl/entorno-desarrollo-integrado-ide/.

SÁNCHEZ, GONZÁLEZ. 2011. *La gestión empresarial de las entidades cubanas.* 2011. ISSN-1029-5186 15:1-11.

Stallman, Richard M. 2015. Free Software, Free Society. [En línea] 5 de junio de 2015. [Citado el: 20 de enero de 2016.] https://www.gnu.org/doc/fsfs-ii-2.pdf.

UNWORK. 2015. UNWORK: Conceptode. [En línea] 2015. [Citado el: 15 de enero de 2017.] http://concepto.de/gestion/.

Xavier, Molero. 2015. Evaluación y Modelado del Rendimiento de los Sistemas Informáticos. [En línea] 1 de octubre de 2015. [Citado el: 25 de noviembre de 2016.] https://www.researchgate.net/profile/Carlos_Juiz/publication/.

ANEXOS

Tabla 18 HU_2 Adicionar roles a usuario (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_2	**Nombre Historia de Usuario:** Adicionar Roles a Usuario
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “usuarios” en el menú principal, “adicionar roles de usuarios”, finalizando así la HU. Para insertar un nuevo rol de usuario al sistema se deben insertar los siguientes datos: ✓ Nombre. ✓ Permisos.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema. ✓ Se debe tener privilegios de Administración.	
Prototipo de interfaz de usuario:	

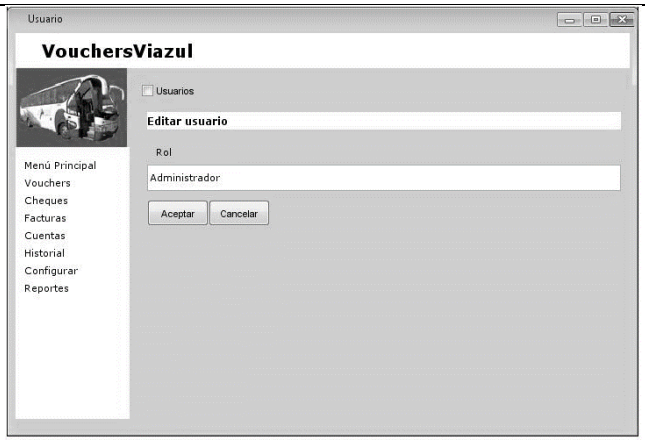


Tabla 19 HU_3 Eliminar usuario (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_3	**Nombre Historia de Usuario:** Eliminar Usuario
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “usuarios” en el menú principal, luego “eliminar usuarios”, finalizando así la HU. Para insertar un nuevo rol de usuario al sistema se deben insertar los siguientes datos: ✓ Nombre. ✓ Permisos.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema. ✓ Se debe tener privilegios de Administración.	

Prototipo de interfaz de usuario:

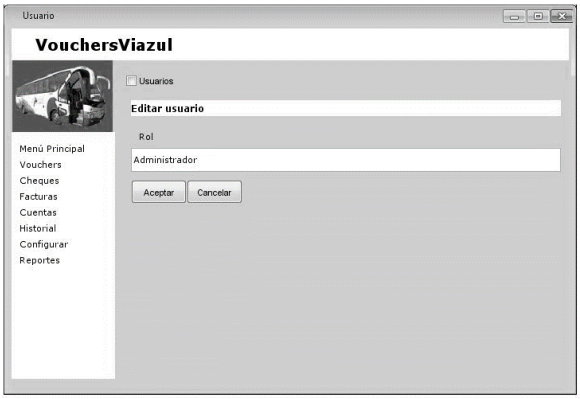


Tabla 20 HU_4 Listar roles a usuario (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_4	**Nombre Historia de Usuario:** Listar Roles a Usuario
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Medio	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “usuarios” en el menú principal y luego aparece la lista de todos los usuarios que han sido creados, finalizando así la HU. Para insertar un nuevo rol de usuario al sistema se deben insertar los siguientes datos: ✓ Nombre. ✓ Permisos.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema.	

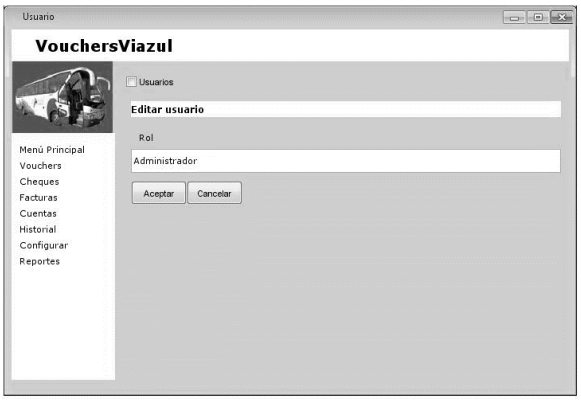
✓ Se debe tener privilegios de Administración.
Prototipo de interfaz de usuario: 

Tabla 21 HU_5 Adicionar Cuentas (Fuente: Elaboración propia)

Historia de Usuario	
Número: HU_5	**Nombre Historia de Usuario:** Adicionar Cuentas
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “cuentas” en el menú principal, luego el sistema permitirá insertar y el usuario debe seleccionar la operación realizar dando clic en el botón correspondiente a la operación, finalizando así la HU.	
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.	
Prototipo de interfaz de usuario:	

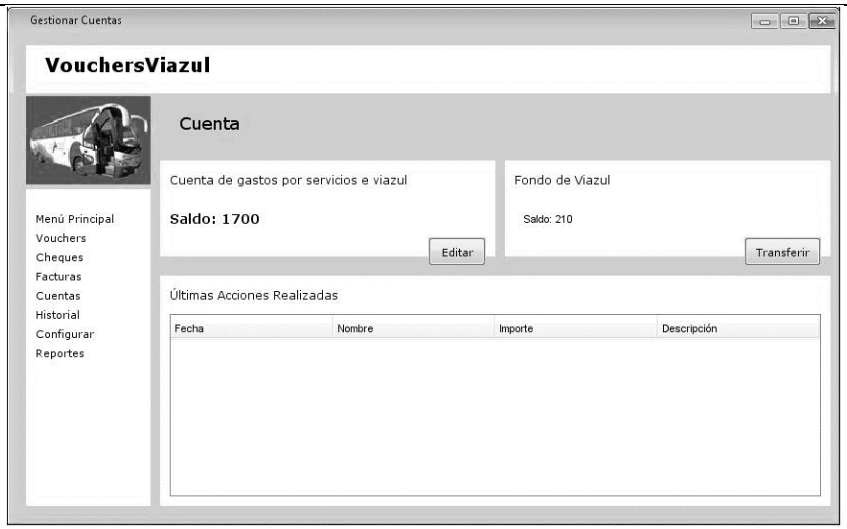


Tabla 22 HU_6 Editar Cuentas (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_6	**Nombre Historia de Usuario:** Editar Cuentas
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “cuentas” en el menú principal, luego el sistema permitirá modificar y el usuario debe seleccionar la operación realizar dando clic en el botón correspondiente a la operación, finalizando así la HU.	
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.	

Prototipo de interfaz de usuario:

Tabla 23 HU_7 Realizar transferencias entre cuentas (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_7	**Nombre Historia de Usuario:** Realizar transferencias entre Cuentas
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción “cuentas” en el menú principal, luego el sistema permitirá realizar transferencias entre una cuenta y otra, finalizando así la HU.	
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.	
Prototipo de interfaz de usuario:	

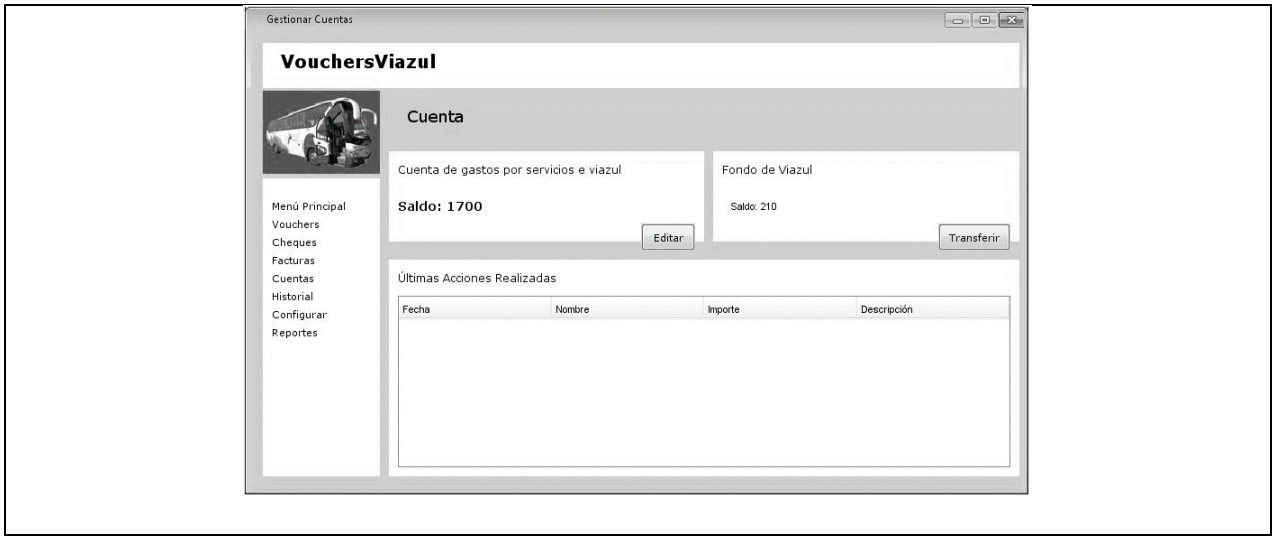


Tabla 24 HU_8 Adicionar Voucher (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_8	**Nombre Historia de Usuario:** Adicionar *Voucher*
Modificación de Historia de Usuario Número: *1*	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 1
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción "*voucher*" en el menú principal, luego el sistema permitirá insertar, dando clic en el botón correspondiente a la operación, después aparecerá unos campos que serán obligatorios a llenar, finalizando así la HU.	
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorio.	
Prototipo de interfaz de usuario:	

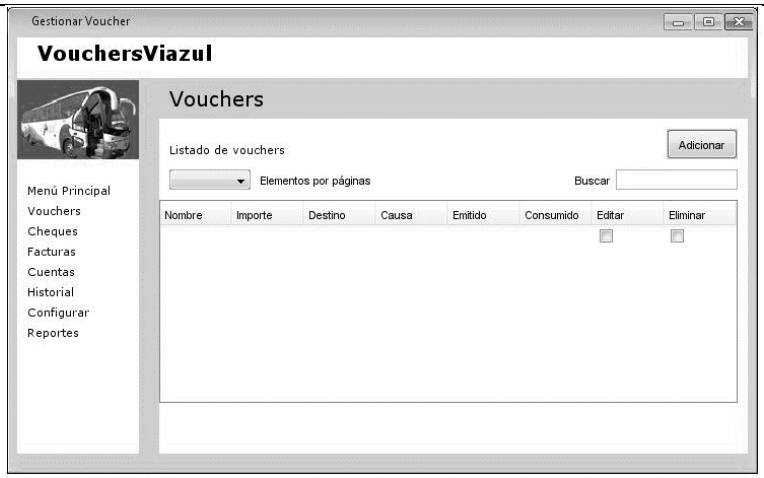


Tabla 25 HU_9 Buscar Voucher (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_9	**Nombre Historia de Usuario:** Buscar *Voucher*
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción "*voucher*" en el menú principal, luego aparecerá un campo para introducir el nombre de la persona que consumió el *voucher* o cualquier otro elemento como el área, destino, entre otros, posteriormente el sistema mostrará todos los datos del *voucher* correspondiente, finalizando así la HU.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema.	
Prototipo de interfaz de usuario:	

Tabla 26 HU_10 Eliminar Voucher (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_10	**Nombre Historia de Usuario:** Eliminar *Voucher*
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción "*voucher*" en el menú principal, luego el sistema permitirá eliminar, dando clic en el botón correspondiente a la operación, finalizando así la HU.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema.	
Prototipo de interfaz de usuario:	

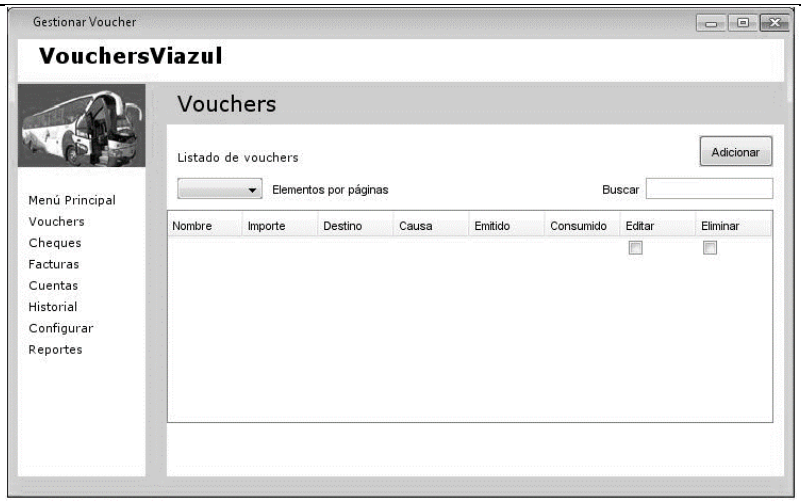


Tabla 27 HU_11 Listar Voucher (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_11	**Nombre Historia de Usuario:** Listar *Voucher*
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Muy Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Media	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario selecciona la opción "*voucher*" en el menú principal, luego el sistema mostrará una lista con todos los *voucher*s ya creados, finalizando así la HU.	
Observaciones: ✓ Todos los campos son obligatorios. ✓ Se debe estar autenticado en el sistema.	
Prototipo de interfaz de usuario:	

Tabla 28 HU_12 Mostrar Historial de acciones (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_12	**Nombre Historia de Usuario:** Mostrar Historial de acciones.
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Alta	**Puntos Estimados:** 5 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 5 días.
Descripción: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: “historial de acciones”. El sistema muestra los datos correspondientes a todas las acciones realizadas que se almacenan en el sistema, finalizando así la HU.	
Observaciones: ✓ Se debe estar autenticado en el sistema.	
Prototipo de interfaz de usuario:	

Tabla 29 HU_13 Registrar consumo del Voucher (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_13	**Nombre Historia de Usuario:** Registrar consumo del *Voucher*
Modificación de Historia de Usuario Número: *1*	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 4 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 4 días.
Descripción: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: "*voucher*". El sistema muestra los datos correspondientes a los *voucher* almacenados en el sistema, un campo para que el usuario pueda introducir el nombre de la persona que consumió el *voucher*, una vez seleccionado el *voucher* se selecciona la opción "editar" que permitirá al usuario cambiar el estado del *voucher* ha consumido. El sistema seguidamente debe mostrar los datos del *voucher* modificado, finalizando así la HU.	
Observaciones:	

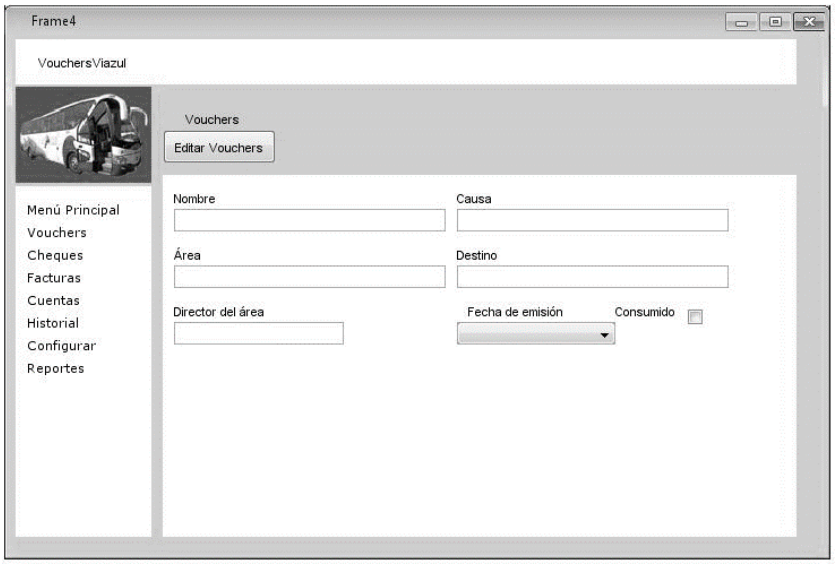
✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.
Prototipo de interfaz de usuario: 

Tabla 30 HU_14 Mostrar Voucher Insertado (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_14	**Nombre Historia de Usuario:** Mostrar *Voucher* insertado
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 4 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 4 días.
Descripción: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: "*voucher*". El sistema muestra todos los datos referentes a los *voucher*, el usuario selecciona la opción de insertar un nuevo *voucher* al sistema e introduce los nuevos datos. El sistema seguidamente debe mostrar los datos del *voucher* insertado como prueba de que se registró correctamente, finalizando así la HU.	

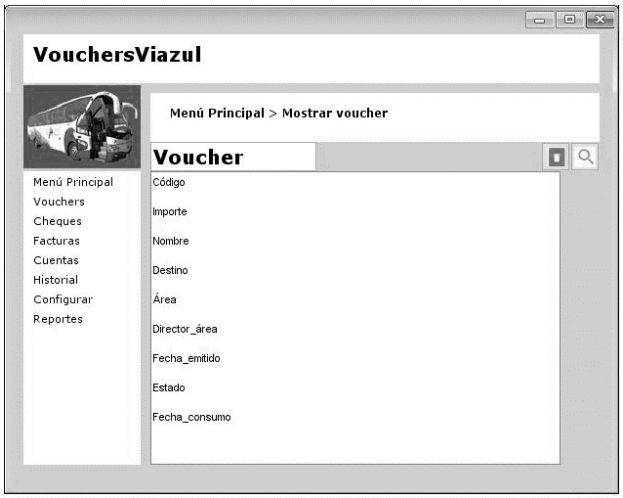
Observaciones:
✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.
Prototipo de interfaz de usuario: 

Tabla 31 HU_15 Generar Reportes (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_15	**Nombre Historia de Usuario:** Generar Reportes.
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 4 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 4 días.
Descripción: **Descripción**: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: “generar reportes”. El sistema muestra filtros por los cuales se pueden realizar los reportes para que el usuario seleccione los elementos que contendrá el reporte, los cuales pueden ser área, destino, causa y fecha. El sistema seguidamente debe mostrar los datos almacenados	

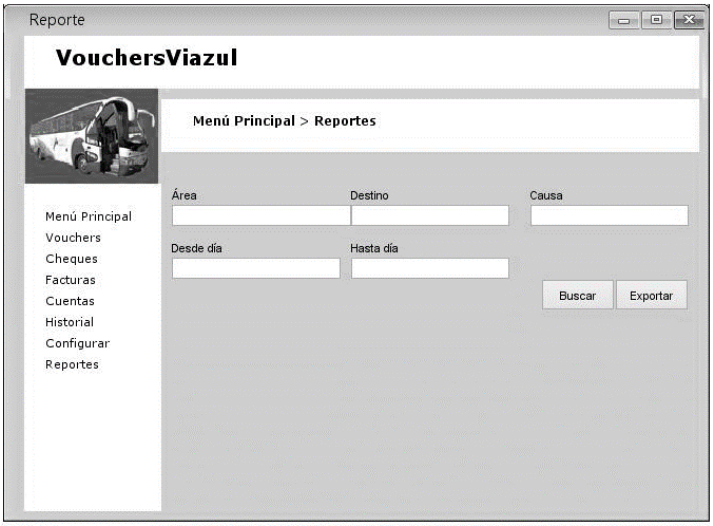
correspondientes a la elección hecha por el usuario, y luego el cliente da un clic en el botón "**g**enerar reportes", finalizando así la HU.
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.
Prototipo de interfaz de usuario: 

Tabla 32 HU_16 Generar reportes por Áreas (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_16	**Nombre Historia de Usuario:** Generar reportes por Áreas.
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 4 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 4 días.
Descripción: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: "generar reportes". Después escribe en la opción área dejando los demás campos vacíos y el	

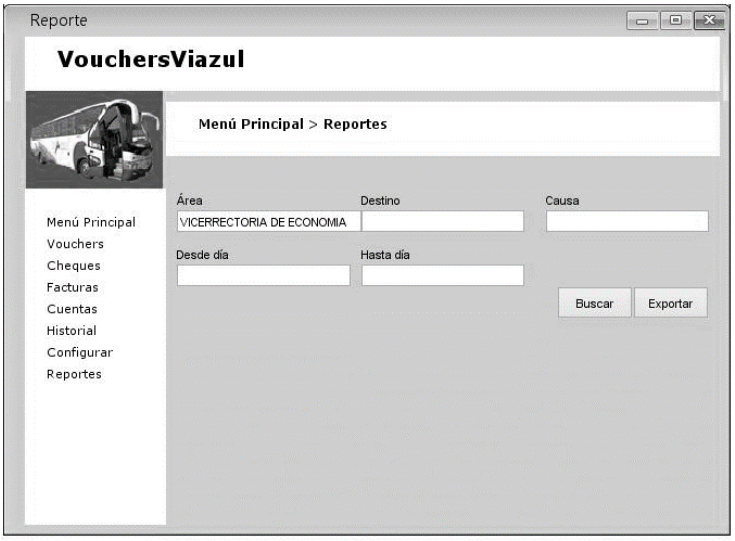
sistema seguidamente debe mostrar los datos almacenados correspondientes a dicha selección, y luego el cliente da un clic en el botón “**g**enerar reportes”, finalizando así la HU.
Observaciones: ✓ Se debe estar autenticado en el sistema. ✓ Todos los campos son obligatorios.
Prototipo de interfaz de usuario: 

Tabla 33 HU_17 Generar reportes por Destino (Fuente: Elaboración propia).

Historia de Usuario	
Número: HU_17	**Nombre Historia de Usuario:** Generar reportes por Destino.
Modificación de Historia de Usuario Número: 1	
Usuario: Liosbel Margolles Clausell	**Iteración Asignada:** 2
Prioridad en Negocio: Media	**Puntos Estimados:** 4 días.
Riesgo en Desarrollo: Baja	**Puntos Reales:** 4 días.

Descripción: La HU inicia cuando el usuario accede a la página principal del sistema, y selecciona la opción: “generar reportes”. Después escribe en la opción destino dejando los demás campos vacíos y el sistema seguidamente debe mostrar los datos almacenados correspondientes a dicha selección, y luego el cliente da un clic en el botón “**g**enerar reportes”, finalizando así la HU.

Observaciones:

- ✓ Se debe estar autenticado en el sistema.
- ✓ Todos los campos son obligatorios.

Prototipo de interfaz de usuario: