UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad 6

Subsistema de Administración para el Sistema de Visualización de Información al Viajero

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autores: Yunior Camilo Cabrera Domínguez

Zaida González Cisneros

Tutores: MSc. Alexis René Rodríguez León

Ing. Yordanys Piñeiro Gómez

La Habana, Junio del 2014

"Año 56 de la Revolución"

Declaración de autoría

Declaramos que somos los únicos autores de este trabajo titulado: “Subsistema de Administración para el Sistema de Visualización de Información al Viajero” y autorizamos al proyecto PRIMICIA, de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los 19 días del mes de junio del año 2014.

_____

Autor: Zaida González Cisneros

_____

Autor: Yunior Camilo Cabrera Domínguez

_____

Tutor: Ing. Yordanys Piñeiro Gómez

_____

Tutor: MSc. Alexis René Rodríguez León

Datos de Contacto

Autor: Yunior Camilo Cabrera Domínguez
Dirección: Edificio 2 apto 5, Micro x, Alamar, Habana del Este, Ciudad de la Habana.
Teléfono (Casa): +58-01-16-09.
Correo electrónico: yccabrera@estudiantes.uci.cu

Autor: Zaida González Cisneros
Dirección: Km ½ Ctra. Briones Montoto, Coop: Gilberto Hernández, P. del Rio, Pinar del Rio.
Teléfono (Casa): +52-29-17-82.
Correo electrónico: zgonzalez@estudiantes.uci.cu

Tutor: Ing. Yordanys Piñeiro Gómez
Graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas en Julio del 2007.
Categoría docente: Asistente
Profesor de las asignaturas Ingeniería de Software I y II, Gestión de Software y Preparación para la Prueba de Nivel de IGSW.
Cargo: Asesora del grupo de estudios de Tecnología Educativa. Centro de Innovación y Calidad de la Educación (CICE). Vicerrectoría de Formación.
Dirección: Carretera a San Antonio, Km 2 ½, Reparto Torrens, La Lisa, La Habana.
Teléfono Oficina: +53-7-837–2522.
Correo electrónico: ypineirog@uci.cu

Agradecimientos

En el trascurso de estos 5 años hubo un momento de duda, de desconfianza, de debilidad, pensé que no estaba capacitada para llegar al final de la meta que muchos de nosotros tenemos en la vida.

Hoy le doy gracias a Dios por esa promesa que me dio en Is 41:10 "*No temas, porque yo estoy contigo; no desmayes, porque yo soy tu Dios que te esfuerzo; siempre te ayudaré, siempre te sustentaré con la diestra de mi justicia.*"

Gracias:

Mi Jesús porque nunca me abandonaste, siempre estuviste para mí, gracias por tenerme como tu hija y serme heredera de tu amor, gracia y misericordia.

A mi familia por su apoyo incondicional, gracias a mi amiga Kathy por ser ese brazo que te sostiene cuando no tienes fuerzas, gracias por hacerme reír cuando estaba triste.

A Ronny por amarme y cuidarme. Gracias a mis hermanos en Cristo por ser parte de una gran etapa de mi vida, gracias por sus oraciones y por su amor.

A todos mis compañeros de clases, amistades de la UCI siempre los recordaré.

Gracias a mis tutores y a mi compañero de tesis por ayudarme a cumplir este sueño.

Y muchísimas gracias a mi esposo, gracias mi amor, por todo tu apoyo gracias, te amo.

Zaida González Cisneros

Agradecimientos

A Jehová Dios por darme la posibilidad de vivir y poder lograr este sueño, por guiarme y darme las fuerzas necesarias para enfrentar todos los obstáculos que se presentaron a lo largo del desafío de convertirme en un profesional.

A mi mamá por todo lo que haces por mí. Por enseñarme a luchar para hacer realidad mis sueños. Por demostrarme que sí se puede. Por todo el amor y cariño infinito que siempre me has dado, por creer en mí, te amo.

A mi papá por toda la preocupación y atención que me has dado durante los años de la carrera.

A mis tíos Osmany y Magie siempre pendientes de mí en todo lo que he necesitado, son una columna para poder sostenerme y no caerme, los quiero mucho.

A mi primo hermano Paddy por todo el cariño y amistad que me brindaste, sé que te hubiera gustado estar aquí conmigo todos estos años.

A mis hermanos Yadiel, Yariel, Camila, Claudia me siento orgulloso de ustedes siempre pendientes de mí los quiero mucho.

A mi abuelita Teresa por su amor y compresión.

A Heidy que ha sido extraordinaria, por su amor, su paciencia, por su dedicación, porque mis sueños también son los suyos, mis preocupaciones y mis problemas también son sus problemas, por ser la mujer más especial de este mundo.

A mis suegros que me apoyaron mucho y gran parte de este trabajo fue más liviano gracias a ustedes.

Agradecimientos

A mis tutores Carlos, Alexis y Yordanys por todo el tiempo que dedicaron a apoyarnos en la realización de este trabajo. A Yordanys por su preocupación y por su apoyo incondicional. A Pedro Palau por su apoyo y las tantas horas de desvelo.

A nuestro tribunal Lisbeth, Félix, Dorgis y Eddy por el tiempo que incondicionalmente dedicaron en la realización de este trabajo. Gracias a todas las críticas constructivas que recibimos de ustedes logramos un mejor trabajo de diploma.

A mi compañera de tesis, somos el dúo de la historia, fuiste mi brazo fuerte en todo momento de esta travesía.

A mis amigos Rainer y Oscar por aguantar todas mis pesadeces y estar presentes a mi lado en los momentos más difíciles.

Estoy y estaré agradecido infinitamente de todos los que de una forma u otra contribuyeron y ayudaron en la realización de este trabajo. A todos muchísimas gracias.

Yunior Camilo Cabrera Domínguez

Dedicatoria

Dedico este trabajo a mi Señor y Creador Jesús Cristo, solo por Él he logrado vencer una de las barreras en mi vida y poder darle un motivo más de alegría a mi familia. Hoy te doy gracias por ser mi consuelo, mi sustento, mi ayudador y principalmente por darme un día más de vida para poder alabarte y darte la gloria que mereces.

Dedico también este trabajo a mi mamá y mi papá por siempre estar a mi lado en los buenos y malos momentos de mi vida, por cuidarme, aconsejarme, por hacer de mí la persona que soy hoy, por todo lo que me demuestran a diario y por ser los mejores padres del mundo.

A mi hermana Zoila por sus consejos a pesar de ser menor que yo, por tener un corazón tan grande y demostrarme a diario cuanto me quiere.

Y por último pero no más importante a mi esposo Yasel, gracias mi amor porque junto con Dios estuviste secando esas lágrimas de angustias y desespero, gracias porque tu amor me ayudó a salir adelante, gracias por tu paciencia, gracias por ser tú.

Zaida González Cisneros

Dedicatoria

Dedico el resultado de esta investigación, a las personas más importantes de mi vida:

A mi mamá por ser la mejor madre del mundo y convertirme en el hombre que soy hoy, tú has hecho realidad este sueño.

A mi papá en estos años fuiste de gran apoyo para mí.

A la memoria de mi abuelo Osmany que le extraño mucho.

A mis tíos Osmany y Magie que en estos años se han convertido en otros padres para mí.

A mi primo hermano Paddy desde pequeño siempre te vi como un hermano.

A mis dos hermanitas Camila y Claudia las quiero mucho, ustedes están en lo más profundo de mi corazón.

A mis hermanos Yadiel y Yariel aunque estén lejos son importantes en mi vida.

A mi abuela Teresa, eres el centro mi familia.

A mi novia Heidy, nadie más que tú sabes lo que enfrenté todos estos años para construir este sueño.

A mis dos perlitas Alejandrito y Osmany Cesar, son lo más lindo que existe.

Son infinitas las palabras de amor y los años que cada uno de ustedes me han dedicado como para pretender que quepan en un papel. Espero que se sientan orgullosos de mí como lo estoy yo de ustedes. He recibido de la vida el regalo más bello, mi familia.

Yunior Camilo Cabrera Domínguez

Resumen

En las terminales de transportación es fundamental que el viajero esté informado sobre el medio de transporte que va a abordar. Esta información se le hace llegar por diferentes vías: de forma personal, a través del altavoz, o por medio de pizarras y pancartas. En varias ocasiones el pasajero no recibe la información en su totalidad por algunas de estas vías, ya que no se encuentra en el momento que se está brindando o recientemente llega a la terminal. Para visualizar la información se realiza primero la administración de la misma. En algunas de estas terminales la administración de información se efectúa de manera estática y manual, lo que trae como consecuencia que no siempre se cuente con la inmediatez que se necesita. La propuesta de solución consiste en el desarrollo del Subsistema de Administración para el Sistema de Visualización de Información. El mismo tiene como objetivo principal gestionar la información que se le brinda al cliente, para así mantenerlo actualizado sobre todo el proceso de transportación. Para desarrollar este sistema informático se eligió *Symfony2* como marco de trabajo, *Doctrine2* como capa de acceso a datos para el mapeo Objeto-Relacional y el Proceso Unificado de Desarrollo como metodología de desarrollo de software.

Palabras claves: información, administración, subsistema.

Abstract

In bus, train station or just airports is essential that the passenger has enough information about the means of transportation that will take. This information could be given the traveler in different ways: personally, through the speaker, or through boards and banners. However, the passenger not always receive the whole information in this way because it is not there in the right moment the information is being shown or has recently arrived to the bus or train station or to the airport. In some of these stations, the information management is done statically or manually before visualizing and that´s the reason why the information is not always given to the users on time. The solution in this investigation consists of in the development of a Management Subsystem for Information and Visualization System. The objective is to manage the information given to the user to keep them updated, about the whole process of transportation. To develop this computer system Symfony2 as a *framework*, Doctrine 2 as a data access layer for mapping Object-Relational and Rational Unified Process as a software development methodology were used.

Keywords: information, management, subsystem.

Índice

Índice de Tablas

Índice de Figuras

Introducción

La información ha alimentado los medios de comunicación desde sus orígenes hasta la actualidad, debido al desarrollo alcanzado por las nuevas Tecnologías de la Información y las Comunicaciones (TIC). Dentro de los medios de comunicación, la televisión ha jugado un papel fundamental ya que a través de la misma la mayoría de la población recibe formación y entretenimiento. Su poder radica en muchos factores, pero sobre todo en el hecho de que permite que los telespectadores puedan aprovecharla a distintos niveles, independientemente de sus recursos, formación, expectativas y necesidades.

En el mundo, las terminales de transportación emplean la televisión como medio principal de visualización de información referente a todos los procesos relacionados con la transportación. Esto brinda un mejor servicio para el ciudadano ya que posibilita que se muestren datos actualizados sobre el transporte a abordar. La mayoría de estas terminales cuentan con un departamento de información encargado de gestionar toda la información referente a las salidas y retorno del medio de transporte a la estación. En las mismas se emplean sistemas automatizados para gestionar y controlar la información a visualizar mostrando datos como: hora de llegada y salida del medio de transporte, estado en que se encuentra, precio del pasaje, etc.

En Cuba, las terminales de transportación también cuentan con un departamento de información que permite que los clientes se mantengan actualizados sobre la transportación, además de administrar los datos que se brindarán. La información se le hace llegar al viajero por diferentes vías: de forma personal, a través del altavoz, por medio de pizarras, pancartas y a través de sistemas de visualización de información.

Cuando la información se brinda de forma personal, puede ser que el viajero no se encuentre en el momento que se está dando, o se reciba con baja calidad, lo cual implique una ausencia o nulo entendimiento de la misma. Esto trae como consecuencia que el pasajero tenga que dirigirse al departamento de información, esperar a ser atendido y así aclarar sus dudas. Cuando la información se brinda por medio del altavoz, muchos de los viajeros no prestan atención en el momento que se está transmitiendo o no escuchan con claridad lo que se quiere informar lo que trae consigo que la información recibida sea incorrecta, de tal forma que podrían perder el medio de transporte.

En caso de que al viajero se le informe por medio de pizarras y pancartas, los datos tendrían que actualizarse constantemente de forma manual, siendo así más complejo para el personal encargado de esta actividad por el tiempo que se emplea. Si el cliente quiere saber alguna información específica sobre la llegada o salida de algún medio de transporte tendría que esperar a que se diga por el altavoz o dirigirse al departamento de información ya que por medio de pizarras y pancartas no obtiene los datos que necesita en un momento determinado.

Para la visualización de esta información se realiza la administración de la misma. En algunas de estas terminales la administración de información se efectúa de manera estática y manual. Un ejemplo de esto sucede cuando una sola persona ocupa la responsabilidad de varias operaciones como responder las inquietudes de los visitantes en la oficina de información, vender pasajes y actualizar la información en la pizarra. Lo antes planteado influye en que proceso de gestión de la información no se realice con la rapidez necesaria logrando la insatisfacción de los visitantes a la terminal.

Por todo lo anteriormente descrito se plantea como **problema a resolver**: ¿Cómo contribuir a la simplificación de los procesos de administración de la información que se le brinda al viajero en las terminales de transporte en Cuba? El problema descrito centra el **objeto de estudio** en los sistemas de visualización de información al viajero en terminales de transporte y como **campo de acción** la administración de los sistemas de visualización de información al viajero para las terminales de transporte en Cuba.

Con la finalidad de darle cumplimiento al problema se precisa el siguiente **objetivo general**: desarrollar un Subsistema de Administración para el Sistema de Visualización de Información al Viajero que simplifique los procesos de la administración de la información que se le brinda al viajero en las terminales de transporte en Cuba.

Para guiar la investigación se definen las siguientes **preguntas de investigación**:

1. ¿Cuáles son los fundamentos teóricos que sustentan los procesos de administración en las terminales de transporte para mejorar los servicios de información al viajero?

2. ¿Qué características debe tener el Subsistema de Administración para el Sistema de Visualización de Información al Viajero para mejorar los servicios de información en las terminales de transporte en Cuba?
3. ¿Cómo organizar el proceso de desarrollo del Subsistema de Administración para el Sistema de Visualización de Información al Viajero para mejorar los servicios de información en las terminales de transporte en Cuba?
4. ¿El Subsistema de Administración desarrollado permite simplificar los procesos de administración de la información que se le brinda al viajero en las terminales de transporte en Cuba?

Las **tareas de investigación** propuestas para dar cumplimiento a los objetivos trazados son:

1. Análisis de los principales conceptos de gestión de archivo y administración informática, además de especificar los conceptos que guiarán el desarrollo de la investigación.
2. Caracterización de los procesos relacionados con la administración de información al viajero.
3. Selección de las herramientas, tecnologías y metodología para el desarrollo del subsistema.
4. Realización del levantamiento de los requisitos funcionales y no funcionales con los que debe contar el Subsistema de Administración.
5. Implementación del subsistema propuesto.
6. Desarrollo de los casos de prueba que certifiquen el correcto funcionamiento de los algoritmos implementados.

Para apoyar el desarrollo de la investigación se emplearon los siguientes métodos científicos:

Métodos Teóricos

Los métodos teóricos permiten ascender del acondicionamiento de información empírica a describir, explicar, determinar las causas y formular la hipótesis investigativa (Hernández León, y otros, 2011).

Analítico-Sintético: Este método permite realizar una definición de los conceptos principales de la investigación a partir de distintas fuentes bibliográficas relacionadas con la gestión de información en las terminales de transportación. También extrae los elementos más importantes que se relacionan con el

objeto de estudio obteniendo como resultado la confección del diseño teórico y metodológico de la investigación.

Histórico-Lógico: Este método facilita el estudio de las etapas por las que ha transcurrido la administración de la información en las estaciones o terminales de transportación, obteniendo así como resultado las características y elementos fundamentales que intervienen en el proceso en cuestión.

Métodos Empíricos

La investigación empírica permite al investigador hacer una serie de investigación referente a su problemática, retomando experiencia de otros autores, para de ahí partir con su exploración, también conlleva efectuar el análisis preliminar de la información, así como verificar y comprobar las concepciones teóricas (Álvarez de Zayas, 2007).

Observación: Este método permite de una manera u otra identificar las principales actividades que se deben desarrollar para la administración de la información que se le brinda al viajero.

Análisis documental: Este método permite realizar el estudio de varios documentos referentes a los sistemas de visualización de información al viajero con el objetivo de obtener la experiencia y sugerencias que pudieran ser incorporadas a través de esta investigación.

El documento está estructurado de la siguiente manera: cuatro capítulos que incluye todo lo relacionado a la investigación, así como la propuesta final que ha dado como resultado. A continuación se brinda una panorámica del contenido de cada uno de estos capítulos.

Capítulo I: Fundamentación Teórica. En este capítulo se profundizan los conceptos fundamentales del negocio asociados al tema de investigación, así como la caracterización de su objeto de estudio. También se aborda con mayor profundidad la situación problemática y se proponen las tecnologías, metodología de desarrollo y herramientas que se van a utilizar en el diseño e implementación del sistema.

Capítulo II Propuesta de solución para el Subsistema de Administración del Sistema de Visualización de Información al Viajero. Mediante el desarrollo de este capítulo se realiza el modelo de dominio concerniente al problema planteado. Se definen las características que debe tener el subsistema mediante

la especificación de los requisitos funcionales y no funcionales. Además de describir la propuesta de solución a través de los actores, casos de uso y la descripción extendida de cada uno de ellos.

Capítulo III: Diseño del Subsistema de Administración del Sistema de Visualización de Información al Viajero. En este capítulo se realiza la descripción de la arquitectura, definiendo los patrones de arquitectura y de diseño que se utilizan, permitiendo definir la estructura general del software, así como el diseño del mismo. Se presentan los Diagramas de Clases del Diseño, además del Diagrama de Despliegue.

Capítulo IV: Implementación y Pruebas del Subsistema de Administración para el Sistema de Visualización de Información al Viajero. En este capítulo se exponen algunos de los principales artefactos generados en los flujos de trabajo, implementación y prueba necesarios en el desarrollo del software, que dará solución al problema a resolver de la presente investigación. Se realizan pruebas al software con el objetivo de comprobar el correcto cumplimiento de los requisitos funcionales propuestos para el componente.

Para finalizar se presentan las conclusiones generales, recomendaciones, bibliografías, referencias bibliográficas y un conjunto de anexos para un mejor entendimiento de lo expuesto en la investigación.

Capítulo I: Fundamentación teórica.

Introducción.

En este capítulo se abordan los aspectos fundamentales que servirán de soporte teórico para el desarrollo de toda la investigación. Se realiza un estudio de algunas soluciones existentes a nivel nacional e internacional relacionadas con la investigación. Se analiza con mayor profundidad la situación problemática que dio surgimiento a la presente investigación. También se explican las razones del uso de la metodología de desarrollo escogida, así como los lenguajes y herramientas empleadas para guiar y dar soporte a la solución propuesta.

1.1 Principales conceptos asociados al tema de investigación.

Para lograr una mejor comprensión del contexto en el que se desarrolla la investigación, se definen a continuación los principales conceptos asociados a la misma.

1.1.1 Subsistema.

Para definir un subsistema se debe saber que es un sistema como tal. Un sistema es un conjunto de partes o acontecimientos que son interdependientes entre sí e interaccionan, por lo que puede considerarse como un todo sencillo, es decir, llamamos sistemas a los conjuntos compuestos de elementos que interactúan (Biología y Geología Interactiva, 2012). Por lo tanto cuando se habla de subsistemas se refieren a las partes o los módulos que forman un sistema. Cada sistema está compuesto de “subsistemas”, los cuales a su vez son parte de otros subsistemas delineados por sus límites (Catédras Mendoza, 2008).

Un subsistema permite dividir el sistema entero en partes más adaptables. Cada subsistema tiene aspectos del sistema con propiedades comunes, siendo un conjunto de elementos ordenados para cumplir con un propósito, estas partes deben cumplir con determinadas condiciones hasta que se complementen formando el sistema (Falgueras Campderrich, 2008).

De manera general se plantea que un subsistema se puede definir como cada uno de los componentes principales de un sistema, teniendo conocimiento de que cada subsistema abarca aspectos del sistema que comparten alguna propiedad común.

1.1.2 Administración.

Según Idalberto Chiavenato (Chiavenato, 2007), la administración es *"el proceso de planear, organizar, dirigir y controlar el uso de los recursos para lograr los objetivos organizacionales".* Para Robbins y Coulter (Robbins , y otros, 2005), la administración es la *"coordinación de las actividades de trabajo de modo que se realicen de manera eficiente y eficaz con otras personas y a través de ellas".* Según Díez de Castro, García del Junco, Martín Jiménez y Periáñez Cristóbal (Castro, y otros, 2009), la administración es *"el conjunto de las funciones o procesos básicos (planificar, organizar, dirigir, coordinar y controlar) que, realizados convenientemente, repercuten de forma positiva en la eficacia y eficiencia de la actividad realizada en la organización"*.

Teniendo en cuenta los diferentes conceptos de administración que plantean estos autores se toma para esta investigación que la administración es el proceso de planificar, organizar, dirigir y controlar toda la información, con el propósito de lograr la digitalización de la información de manera eficiente y eficaz.

1.1.3 Itinerario.

Itinerario permite hacer referencia al rumbo, orientación y descripción de un determinado trayecto, recorrido o camino, el cual contempla la inclusión de citas a los sitios, descansos y accidentes que pueden llegar a aparecer durante la travesía. Asimismo, se conoce como itinerario a la ruta que se elige a fin de arribar a un cierto destino o el listado de datos referentes a un viaje (Montero, 2008).

Según el Diccionario de la Real Academia de la Lengua Española (DRAE) el término itinerario se define como:

"Camino previsto por donde debe discurrir un recorrido o viaje. Ruta" (Española, 2013).

Para la presente investigación se determina itinerario como la descripción de las características principales de un camino o ruta, con sus puntos de paso y paradas.

1.1.4 Gestión de Información.

Según el diccionario de la Real Academia Española (Española, 2013) la *"Información es la comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada".* Por lo tanto cuando se habla de la gestión de información se refiere al proceso que incluye operaciones como extracción, manipulación, tratamiento, depuración, conservación, acceso y/o colaboración de la información adquirida por una organización a través de diferentes fuentes y que gestiona el acceso y los derechos de los usuarios sobre la misma (Dans, 2007).

1.2 Análisis de la Situación Problemática.

En la actualidad, en las terminales de transporte, el proceso de administración de información al viajero se está adentrando cada vez más en el mundo de la digitalización de información. Cada terminal de transporte se enfrasca en encontrar la manera de lograr que cada persona que visite la entidad se sienta informado de una forma amena y que las ofertas y servicios que se brindan, logren satisfacer y atraer a más clientes.

En Cuba, cada terminal tiene una manera específica de lograr la administración de toda la información, algunas emplean métodos automatizados e incluso la mayoría sin el uso de las TIC. Uno de los casos donde se evidencian los problemas anteriormente mencionados es en las terminales de ómnibus de Vía Azul: la información se visualiza en una pantalla estática que incluye tablas con todos los datos referentes a la trasportación, donde el viajero no puede ver con claridad los datos mostrados, por lo cual se evidencia que no existe una correcta administración de la información para poder mostrarla de manera más práctica.

En otras terminales, la información se escribe de forma manual en pizarras haciendo el proceso estático, logrando que sea inevitable la asistencia de los visitantes a las oficinas de información para mantenerse actualizados sobre la transportación. La mayoría de las terminales no cuenta con un sistema visual de alertas, trayendo consigo que la información de los sucesos imprevistos que surgen a diario se lean por un alta voz y estos se repitan varias veces, que conlleva algunas veces, a que el viajero no logra entender lo que se trató de informar.

Otras dificultades existentes en las terminales de viaje es que la información se almacena y se muestra en forma de imágenes, visualizando una secuencia de las mismas con todos los datos. Para poder lograr informar cualquier cambio de última hora se tiene que editar dichas imágenes, además de que no muestran todos los datos que el cliente desea saber. Esto provoca que el proceso sea ineficiente y que no siempre se cuente con la inmediatez que se necesita. La estancia en las terminales de viajes pudiera ser agradable si se mostraran a través de videos las ofertas que brinda la entidad, las características de los medios de transporte, la información de las rutas, videos musicales o documentales que resulten agradables a los viajeros.

Para lograr una mejor calidad del servicio brindado por las terminales de pasajeros se hace necesario contar con una solución capaz de resolver la necesidad de información que presentan las personas que interactúan con ella, utilizando principalmente los medios de televisión. Para la realización de esta tarea se propone un sistema informático que permita la automatización de los procesos de visualización y administración de todas estas informaciones.

1.3 Análisis de las soluciones existentes.

1.3.1 Soluciones existentes a nivel internacional.

A nivel mundial se han desarrollado disímiles software que permiten la administración de información al viajero. Aunque no se puede afirmar que han alcanzado la perfección, realmente existen aplicaciones muy potentes en el mundo de la información al viajero, a continuación se analizan algunas de ellas:

Sistema de gestión automatizada de información al viajero (SIA).

El SIA es una herramienta creada por la empresa INDRA líder en Tecnologías del Transporte en Madrid (España). Esta herramienta esta específicamente desarrollada para la generación y difusión de información al viajero en estaciones, paradas, apeaderos o a bordo de trenes y autobuses. Este sistema se alimenta básicamente de la información sobre el estado real de la circulación facilitada por los diferentes subsistemas externos existentes como los SAE[1] en autobuses o los CTC[2] ferroviarios, o puede

[1] Sistemas de Ayuda a la Explotación e información a viajeros.

[2] Control de Tráfico Centralizado.

incluir sus propios dispositivos embarcados de localización para el conocimiento en tiempo real de la posición y estado de la flota (INDRA, 2010).

Dependiendo de las fuentes de información y gracias a su flexibilidad, el SIA, es capaz de difundir automáticamente contenidos muy variados:

- ✓ Tiempos de llegada.
- ✓ Destino y origen de próximo autobús/tren.
- ✓ Avisos de emergencia.
- ✓ Mensajes libres.
- ✓ Información corporativa de los operadores y autoridades del transporte.
- ✓ Información de interés general y noticias.
- ✓ Entretenimiento.
- ✓ Publicidad.

En la mayoría de los casos, los contenidos se cargan previamente en modo local por lo que el sistema central transmite únicamente la identificación de qué mensaje debe emitirse y cuándo, con el consiguiente ahorro en costes de comunicaciones.

Algunos campos de la aplicación son:

- ✓ Visualización y monitorización sobre GIS[3] del material rodante y de sistemas en estaciones y paradas.
- ✓ Completo sistema de programación en base a múltiples parámetros: horarios, estacionalidades, días de la semana, festivos, líneas, paradas, autobús, tren, etc.
- ✓ Diferentes sistemas de difusión: megafonía, teleindicadores LED[4] o TFT[5], SMS[6], Web, información a invidentes o Bluetooth entre otros.

[3] Sistema de Información Geográfica.

[4] Diodo Emisor de Luz.

[5] Transistor de Película Fina.

[6] Servicios de Mensajes Cortos.

Según el análisis realizado se llegó a la conclusión que el sistema SIA es un sistema potente en cuanto a la difusión de información de viajeros. SIA presenta muchas funcionalidades que se emplearán en el desarrollo del subsistema: mostrar hora de llegada, alertas, videos de publicidad y entretenimiento. Este sistema solventa parcialmente el problema planteado, pero muchos servicios que brinda se recomiendan para versiones futuras, como la difusión de información al viajero no solo en las terminales sino a bordo del medio de transporte. Además de lo anteriormente expuesto otra de las razones es que es privativo y para su uso habría que pagar una licencia y a su vez depender de varias herramientas ajenas que se distribuyen de forma privativas.

IKUSI.

Este sistema está diseñado para los sistemas de trenes, tranvías y metro. Cuenta con un amplio abanico de soluciones que cubren las necesidades de información, entretenimiento, comunicaciones y seguridad de ferrocarriles, tranvías y metro, desde la doble perspectiva de las instalaciones fijas en las estaciones y el interior de los coches. Los aeropuertos de Chile, México y España son los más destacados en la utilización de este sistema (IKUSI, 2013).

IKUSI ofrece soluciones que abarcan el diseño, fabricación, instalación y mantenimiento de los sistemas, incluyendo un servicio de teleasistencia de 24 horas al día, los 365 días al año. Se ha destacado en el mercado de ferrocarril por ser un proveedor global de soluciones, servicios y sistemas integrados. Algunas de los sistemas asociados a las operaciones que se realizan son:

- ✓ Sistema de Videoinformación: La integración de los sistemas de entretenimiento de audio/vídeo con los sistemas de información al viajero del tren permite una forma más avanzada de presentar la información al pasajero por medio de imágenes. La total integración con el sistema de localización de la unidad hacen del sistema de IKUSI una solución fácil de gestionar obteniendo resultados eficaces (IKUSI, 2013).

- ✓ Sistema de Información al Viajero: Basado en la localización en tiempo real de la unidad de tren, realiza la gestión automatizada de la presentación de la información visual y acústica al pasaje. (IKUSI, 2013).

Adicionalmente, estos sistemas permiten un análisis en tiempo diferido del nivel de servicio prestado, así como funcionalidades de planificación de las operaciones de la flota y de los recursos asociados. IKUSI obtiene como resultado sistemas capaces de optimizar la explotación y singularizar el servicio, proporcionando una imagen diferenciada; sistemas que aportan seguridad y confort en los tiempos de espera al pasajero y, en definitiva, sistemas capaces de satisfacer a la clientela más exigente.

Según el análisis de este sistema se puede concluir que IKUSI no cumple con la problemática presente de gestionar la información en diferentes tipos de entidades de transporte, ya que se aplica solamente en terminales de trenes y no es flexible para su aplicación en aeropuertos o terminales de ómnibus. Se tuvo en cuenta de este sistema la integración de los sistemas de entretenimiento de audio/vídeo con los sistemas de información al viajero.

LINARIA.

El sistema de horarios bien establecido LINARIA *Time Table* (TT) proporciona información para pantallas dinámicas en terminales y paradas a fin de informar a los viajeros sobre las horas de llegada y de salida, retrasos y otra información de tráfico. Se utiliza en sistemas que abarcan todo un país, como en el sistema ferroviario de Suecia, en terminales de autobuses y en paradas individuales. Es una potente herramienta y proporciona pantallas de diálogo basadas en Windows de fácil uso y mantenimiento. Además, presenta un nivel de supervisión simplifica para la administración de las pantallas de información (Linaria, 2012).

LINARIA TT fue desarrollado por la compañía SWARCO MIZAR (Linaria, 2012) y a diario ofrece información confiable y actualizada sobre sus rutas y los cambios que pueden ocurrir en los viajes planificados a millones de pasajeros.

A raíz del análisis realizado en el sistema LINARIA se determina que soluciona parcialmente la problemática de la presente investigación, pues a pesar de que es un sistema aplicable a terminales de ómnibus y trenes, no se aplica en aeropuertos. Sería más costoso, en tiempo y esfuerzo, modificarla para lograr su uso en cualquier tipo de entidad de transporte, además de que es un sistema privativo trayendo consigo el pago de una licencia y a su vez la dependencia de varias herramientas ajenas que se distribuyen de forma privativas.

1.3.2 Soluciones existentes a nivel nacional.

Estación Ómnibus Nacional (EON)

El Grupo Empresarial de Transporte por Ómnibus EON cuenta con una aplicación que su objetivo es recopilar la información planificada en el transcurso de cada mes. Dicha planificación recoge: Provincia de origen/destino, hora de salida/llegada de los ómnibus, días de la semana que tienen efecto los viajes, estado de la salida y el número del ómnibus. Sin embargo este sistema tiene algunas dificultades:

- ✓ El sistema de información tiene cuatro pantallas (Próximas Salidas, Próximos Arribos, Salidas Atrasadas, Arribos Atrasados):
    - El tiempo de cambio entre pantallas es muy extenso.
    - No se establece un sistema de prioridades para las pantallas según la necesidad de mostrar en tiempo la información que se corresponde con la acción inmediata.
- ✓ La aplicación presenta deficiencias en la generación de la secuencia de los días del mes, según la metodología de días pares e impares (día 31 impar la aplicación procede día 1ro par).
- ✓ Además, el software es una aplicación cerrada, que no cuenta con los instaladores correspondientes, lo que impide que se extienda su uso en las restantes estaciones de la empresa en el país.

Según el análisis de este sistema se puede concluir que aunque las informaciones que brinda son de vital importancia para el pasajero, al ser un software que no se cuente con los instaladores correspondientes no se podría modificar para adaptarlo a las necesidades que se plantean en la presente investigación.

Estación Central de Ferrocarriles.

Este sistema se creó en el año 1999, fue implementado por el SIFER[7] con el objetivo de brindar información, principalmente, sobre la transportación ferroviaria del día, referente a las salidas y llegadas de los trenes a la estación. Para su implementación se utilizó como lenguaje de programación Visual Basic con una programación Orientada a Eventos, la cual es poco viable para un software que se caracteriza por constantes ajustes dado los cambios y funcionamiento interno de la estación. Como interfaz gráfica de

[7] Servicio Informático de Ferrocarriles.

salida que se presenta por los televisores, la aplicación utiliza un show[8] de Power Point, los cuales se ajustan según la cantidad de información que se desee mostrar a los clientes.

A raíz del análisis realizado al sistema empleado en la Estación Central de Ferrocarril se llega a la conclusión de que este sistema soluciona parcialmente la problemática de la presente investigación, sin embargo al ser programado en Visual Basic no cumple objetivo modificarlo para lograr su uso en cualquier tipo de entidad de transporte ya que la tecnología que se utiliza está obsoleta.

Terminal Internacional José Martí.

La Empresa Cubana de Aeropuertos y Servicios Aeronáuticos (ECASA) tiene implementado un sistema para brindar información a sus usuarios. Este sistema informativo a clientes se encuentra en diferentes terminales (aeropuertos) principalmente en la capital. En la Terminal 3 (Terminal Internacional José Martí), es donde está el servidor central de vuelos tanto nacionales como internacionales, este servidor está desarrollado en SQL Server 2000.

La información que finalmente sale por pantalla es independiente para cada una de las terminales, en dependencia de la programación del día. El sistema está implementado en Visual Basic, y constituye el software que finalmente da salida a la información por pantalla. Las pantallas brindan a los clientes la relación de los vuelos que partirán o arribarán a la Terminal. Además para cada salida o arribo se expone: el número del vuelo, hora de salida, hora de llegada, observación (En Hora, Arribando y A Chequeo.). Las diferentes pantallas de salidas o arribos, se muestran por área, es decir, hay monitores para presentar solamente las salidas y otros para las llegadas, en dependencia del área de la Terminal en el que se encuentre.

Este sistema de una forma u otra cumple con algunas funcionalidades requeridas, para la presente investigación se llega a la conclusión de que sería más costoso en tiempo y esfuerzo modificarlo, para adaptarlo a la información que se necesita registrar. Además de que el mismo no cumple con las exigencias del cliente de ser un sistema que pueda ser aplicado para cualquier tipo de terminal, pues todos los procesos no se manejan de igual forma en todas las terminales de transporte.

[8] Es para guardar la presentación Power Point en .pps y así la presentación se iniciará sin los pasos intermedios de la apertura de PowerPoint.

1.4 Caracterización de las tecnologías, lenguajes y herramientas a emplear.

A continuación se presentan las características de las diferentes tecnologías, herramientas y metodologías a utilizar para dar solución al problema científico planteado.

1.4.1 Metodología de Desarrollo.

Las metodologías de desarrollo de software tienen como objetivo estructurar, planificar y controlar dicho proceso, por lo que indican paso a paso todas las actividades que se deben efectuar para alcanzar el producto deseado. Esto muestra quién debe participar en el desarrollo de cada una de las actividades y qué papel deben tener en las mismas (Fitzgerald, 2008).

Además, detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. Existe una gran variedad de metodologías para la creación del software, las cuales se pueden clasificar en dos grandes grupos: las llamadas metodologías ágiles o ligeras y las metodologías robustas o rígidas (Carrillo, et al., 2008).

Entre las metodologías clasificadas como ágiles, tales como Programación Extrema (XP, por sus siglas en inglés), Scrum, y *Microsoft Solutions Framework* (MSF), ofrecen una buena solución para aquellos proyectos de entorno volátil, donde los requisitos no se conocen con exactitud y pueden sufrir cambios durante el tiempo de vida del proyecto.

Por su parte las metodologías robustas se encuentran, Proceso Unificado de Desarrollo (RUP, por sus siglas en inglés), Métrica y Proceso Unificado Abierto (OpenUP, por sus siglas en inglés) por solo mencionar algunas de las más conocidas. Este tipo metodologías se centran específicamente en el control de los procesos, demostrando ser efectivas y necesarias en proyectos de gran tamaño respecto a tiempo y recursos.

A pesar de que el Subsistema de Administración para el Sistema de Visualización de Información al Viajero no representa un proyecto de gran tamaño en tiempo y recursos que es una de las características de RUP, esta metodología será la que sustentará el desarrollo del subsistema debido a que centra su atención en llevar una documentación exhaustiva y clara de todo el proceso de desarrollo dando

cumplimiento a un plan de proyecto. Además de que genera una gran cantidad de artefactos, lo cual representa una garantía para la continuidad del trabajo del sistema y la futura implementación de funcionalidades. Se cuenta con el apoyo de especialistas con gran experiencia garantizando una mejor calidad del software. También se tuvo en cuenta la metodología que se empleó en los demás subsistemas que conforman el Sistema de Visualización de Información al Viajero para seguir un estándar en todos los artefactos generados garantizando una mejor organización.

1.5.1.1 RUP.

La metodología RUP es una de las más usadas para la modelación visual, análisis de requisitos y diseño de sistemas orientados a objetos. Permite especificar, construir, visualizar y documentar los artefactos de un sistema de software. Los autores de RUP destacan que el proceso de desarrollo de software define tres características fundamentales (Pressman, 2005):

- ✓ **Dirigidos por Casos de Usos:** los casos de uso (CU) son una técnica de captura de requisitos que representan funcionalidades del sistema, las cuales definen lo que el usuario desea obtener y permiten guiar todo el ciclo de vida de la aplicación o proyecto, para crear un resultado que satisfaga las necesidades esperadas. Además integran a todos los flujos de trabajo de RUP, sirviendo de punto de partida y de hilo conductor. En otras palabras, esta característica es la que permite establecer trazabilidad entre los artefactos que son generados en las diferentes actividades del proceso de desarrollo.
- ✓ **Centrado en la arquitectura:** la arquitectura es la organización o estructura de las partes más relevantes de un sistema, ya que brinda una perspectiva clara y una visión común del mismo entre todos sus implicados. En esta se describen los procesos del negocio que son más importantes, teniendo en cuenta los elementos de calidad, rendimiento, reutilización y flexibilidad. En otras palabras, es la que permite entender bien el sistema y la toma de decisiones que indican cómo tiene que ser desarrollada la aplicación y en qué orden.
- ✓ **Iterativo e incremental:** permite dividir el trabajo en partes más pequeñas conocidas como fases de desarrollo, de la cual se obtiene un incremento que produce un crecimiento en el producto. Las cuatro fases son: inicio, elaboración, construcción y transición, donde cada una de ellas se divide en iteraciones. Estas iteraciones son planificadas, a medida que se obtienen sus resultados, se

van integrando. Esta característica permite ir mejorando los resultados, para lograr una optimización y corrección de errores en el momento adecuado, perfeccionando así la aplicación final. En otras palabras, va iterando a medida que se van ejecutando las diversas fases y se vuelven a retocar para ser mejoradas o corregidas, mediante lo cual va incrementando la obtención de resultados y la optimización del sistema en general. Esto acarrea que no se cometan fallos en el desarrollo de la aplicación y que se adquiera calidad y experiencia.

RUP como metodología de desarrollo se integra con el Lenguaje Unificado de Modelado permitiendo realizar el análisis, documentación e implementación de sistemas orientados a objetos.

1.4.2 Lenguaje de Modelado, UML.

UML (acrónico de *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Permite la modelación de sistemas con tecnología orientada a objetos. Es un lenguaje gráfico para visualizar, especificar, construir y documentar los artefactos de un sistema con gran cantidad de software. UML proporciona una forma estándar de escribir los planos de un sistema, cubriendo los aspectos conceptuales, tales como procesos del negocio y funciones del sistema, y los aspectos concretos, tales como las clases escritas en un lenguaje de programación específico, esquemas de bases de datos, componentes y software reutilizable (Jacobson, et al., 2000).

De manera general UML presenta las siguientes características:

- ✓ Tecnología orientada a objetos.
- ✓ Desarrollo iterativo e incremental.
- ✓ Permite documentar disímiles artefactos de un proceso de desarrollo.
- ✓ Permite especificar las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- ✓ Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- ✓ Modela el comportamiento del sistema mediante casos de uso, diagramas de secuencia y de colaboración, que sirven para evaluar el estado de este.

Se seleccionó UML como Lenguaje de Modelado, ya que soporta la metodología de desarrollo RUP, además es orientado a sistemas y no a procesos, está consolidado como lenguaje estándar de análisis y

diseño y es fácil de utilizar. Dentro de sus ventajas figura que la trazabilidad y la documentación del proyecto se realiza de una forma ordenada y guiada por casos de uso.

1.4.3 Herramienta CASE, Visual Paradigm v8.0.

Visual Paradigm es una herramienta profesional para el modelado con UML, es ampliamente utilizada en el mundo del software que permite a los profesionales modelar sus diseños. Es un software privativo, pero brinda una versión libre para uso no comercial. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientado a objetos, construcción, pruebas y despliegues. Proporciona un diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad. Es capaz de generar código e ingeniería inversa para varios lenguajes de programación. Visual Paradigm for UML ha sido desarrollada para todos los sistemas operativos compatibles con Java, incluyendo Windows, Linux y Mac OS X (Visual Paradigm, 2006).

Por todas las ventajas anteriormente expuestas se seleccionó Visual Paradigm como herramienta de modelado teniendo en cuenta además, que el equipo de desarrollo cuenta con experiencia suficiente en el trabajo con la misma.

1.4.4 Lenguajes de Programación.

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que pone a disposición del programador para que éste pueda comunicarse con los dispositivos hardware y software existentes (Definición, 2008).

1.4.4.1 Lenguajes utilizados del lado del cliente.

Un lenguaje del lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio. Son lenguajes que basan su procesamiento en el cliente web, es decir que se ejecutan en el navegador del usuario (Alvares, 2001).

Hojas de estilo en cascada (CSS 3).

Es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML[9] y XHTML[10]. CSS es la mejor forma de separar los contenidos y su presentación, además de ser imprescindible para crear páginas web complejas.

A través del CSS se logra una apariencia agradable de las vistas diseñadas e implementadas para el sistema a construir. Describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos (Pérez Eguíluz, 2009).

JavaScript v1.2.

JavaScript es un lenguaje de programación utilizado para crear pequeños programas encargados de realizar acciones dentro del ámbito de una página web. Posee muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, entre otras. Además, pone a disposición del programador todos los elementos que forman la página web, para que este pueda acceder a ellos y modificarlos dinámicamente (Alvares, et al., 2008).

En la aplicación se utilizará, principalmente, para manejar objetos dentro de las páginas web. Dichos objetos facilitan la programación de páginas interactivas, a la vez que se evita la posibilidad de ejecutar comandos que puedan ser peligrosos para la máquina del usuario, tales como formateo de unidades y modificación de archivos.

[9] Lenguaje de marcas de hipertexto.

[10] Xtensible HyperText Markup Language (por sus siglas en ingles). XHTML es básicamente HTML expresado como XML válido.

1.4.4.2 Lenguajes del lado del servidor, PHP v5.4.

Los lenguajes del lado servidor son aquellos lenguajes que se reconocen, ejecutan e interpretan por el propio servidor y que se envían al cliente en un formato comprensible para él (De la Torre, 2006). El sistema se implementará sobre la base del PHP[11] como lenguaje de programación del lado del servidor.

PHP (Acrónimo de Hipertexto Preprocessor) es un lenguaje "del lado del servidor" (esto significa que PHP funciona en un servidor remoto que procesa la página Web antes de que se abra por el navegador del usuario) especialmente creado para el desarrollo de páginas web dinámicas. Puede incluirse con facilidad dentro del código HTML, es un lenguaje multiplataforma, permite las técnicas de Programación Orientada a Objetos. Puede interactuar con muchos motores de bases de datos tales como MySQL, Oracle[12], PostgreSQL, y otros más (Giráldez Reyes, et al., 2008).

1.4.4.3 Otros lenguajes utilizados, XML v 1.0.

XML son las siglas del Lenguaje de Etiquetado Extensible, es un lenguaje que permite jerarquizar y estructurar la información y describir los contenidos dentro del propio documento, así como la reutilización de partes del mismo. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones (Lamarca Lapuente, 2011). Permite al programador dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corren a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello (Alvares, 2001).

1.4.5 Entorno de Desarrollo Integrado (IDE), NetBeans v7.3.

NetBeans es un entorno de desarrollo integrado, creado principalmente para el lenguaje de programación Java, aunque puede servir para otros lenguajes de programación. Esta herramienta ayuda a los programadores escribir, compilar, depurar y ejecutar programas. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, facilitándole al usuario comenzar a trabajar inmediatamente. Además ofrece servicios comunes permitiéndole al desarrollador enfocarse en

[11] Personal Home Page (por sus siglas en ingles).

[12] Es el acrónimo de Oak Ridge Automated Computer And Logical Engine.

la lógica específica de su aplicación. El mismo es un producto libre y gratuito sin restricciones de uso (Eslava Muñoz, 2012).

1.4.6 Framework de Desarrollo.

En el desarrollo de software, un *framework* o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos de software concretos, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas para así ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Symfony2.

Symfony es un *framework* completo diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. Symfony está desarrollado completamente en PHP 5.3. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y Microsoft SQL Server. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows (Potenciar, et al., 2007).

Características de Symfony

- ✓ Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows, Unix, Linux, etc.)
- ✓ Independiente del sistema gestor de bases de datos.
- ✓ Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.

- ✓ Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- ✓ Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

Symfony está basado en un patrón clásico del diseño web conocido como arquitectura Modelo Vista Controlador (MVC) (Potenciar, et al., 2007).

El modelo representa la información con la que trabaja la aplicación, es decir, su lógica de negocio.
La vista transforma el modelo en una página web que permite al usuario interactuar con ella.
El controlador se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

En el desarrollo del subsistema se trabajará con la versión 2.3.4 de Symfony, que cuenta con las características planteadas anteriormente además de la rapidez y flexibilidad. También cuenta con la nueva estructura de directorios de los proyectos llamada bundles que presta una mejor organización del código y los archivos de configuración que se pueden escribir en PHP, XML o YAML[13], a gusto del programador.

JQuery v1.9.1.

JQuery es una biblioteca de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol Modelo de Objetos del Documento(DOM), manejar eventos, desarrollar animaciones, permite agregar efectos y funcionalidades complejas al sitio web como por ejemplo: galerías de fotos dinámicas y elegantes, validación de formularios, calendarios entre otras opciones (Álvarez, 2010).

Algunas de sus características son:

- ✓ Manipulación de la hoja de estilos CSS.
- ✓ Efectos y animaciones.
- ✓ Animaciones personalizadas.

[13] Ain't Markup Language (por sus siglas en ingles).

- ✓ JavaScript asíncrono y XML (AJAX[14]).
- ✓ Soporta extensiones.
- ✓ Utilidades varias como obtener información del navegador, operar con objetos y vectores, funciones para rutinas comunes, etc.
- ✓ JQuery consiste en un único fichero JavaScript que contiene las funcionalidades comunes de DOM[15], eventos, efectos y AJAX.

En algunos sistemas se ha trabajado con JavaScript y ExtJs pero se seleccionó JQuery como biblioteca de presentación ya que este brinda grandes facilidades, pues tiene una gran cantidad de funcionalidades y plugins como Jqgrid[16] que facilitarán la implementación y el trabajo.

Bootstrap v2.3.2.

Bootstrap es un framework que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. La mayor ventaja es que se puede crear interfaces que se adapten a los distintos navegadores apoyándose en un *framework* potente con numerosos componentes webs ahorrando así, tiempo y esfuerzo (Rodríguez, 2012).

Características principales de Bootstrap según el autor:

- ✓ Bootstrap ofrece una serie de plantillas CSS y ficheros JavaScript que nos permiten integrar el framework de forma sencilla y potente en nuestros proyectos webs.
- ✓ Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como tablets y móviles a distintas escalas y resoluciones.
- ✓ Se integra perfectamente con las principales librerías JavaScript, por ejemplo JQuery.
- ✓ Funciona con todos los navegadores, incluido Internet Explorer.
- ✓ Es un proyecto que combina código abierto.

[14] Siglas de Asynchronous JavaScript and XML, es un término que describe un nuevo acercamiento a usar un conjunto de tecnologías existentes juntas.

[15] Modelo de Objetos del Documento o Modelo en Objetos para la Representación de Documentos.

[16] jqGrid utiliza un Java Script Library jQuery y se escribe como plugins para ese paquete.

1.4.7 Sistema Gestor de Base de Datos, PostgreSQL v9.0.

Para la aplicación se utilizó como *framework* de desarrollo Symfony, este es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft (Lamarca Lapuente, 2011).Para el sistema se recomienda utilizar PostgreSQL ya que el equipo de desarrollo cuenta con vasta experiencia en la utilización de esta herramienta además de otras ventajas que se muestran a continuación.

PostgreSQL es un sistema de gestión de base de datos publicado bajo licencia BSD[17], orientada a objetos y libre. Es multiplataforma, aproxima los datos a un modelo objeto-relacional y es capaz de manejar complejas rutinas y reglas. Además, ofrece varios modos de bloqueo para controlar el acceso concurrente a los datos en tablas (Espinoza, 2005).

PostgreSQL presenta una lista de prestaciones: Su administración se basa en usuarios y privilegios. Es altamente confiable en cuanto a estabilidad se refiere. Cuenta con un conjunto de tipos de datos, permitiendo además su extensión mediante tipos y operadores definidos y programados por el usuario. Soporte para vistas, claves foráneas, integridad referencial, disparadores, procedimientos almacenados y sub-consultas.

1.4.8 Servidor web, Apache v2.4.4.

Apache 2 es un servidor HTTP libre, de código abierto y muy usado a nivel mundial. Es multiplataforma y la documentación referente al mismo es muy abundante. Se caracteriza por su gran escalabilidad, seguridad y rendimiento. Trabaja con varios lenguajes como Perl[18], Python[19] y PHP, lo que permite desarrollar aplicaciones web de buena calidad. Está desarrollado bajo la licencia BSD lo cual permite hacer modificaciones en el código fuente siempre que se reconozca el trabajo de la comunidad. Es un servidor web rápido, flexible y eficiente lo cual lo hace el más utilizado en Internet. Es uno de los

[17] Berkeley Software Distribution (por sus siglas en ingles).

[18] Lenguaje de propósito general originalmente desarrollado para la manipulación de texto.

[19] Lenguaje de programación interpretado, cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible.

servidores web más populares en el mundo de la programación web y algunas de las características que hacen que sea uno de los más usados son las siguientes (Pérez, 2006).

- ✓ Permite la creación de sitios web dinámicos en lenguajes de programación como PHP.
- ✓ Se instala en una multitud de Sistemas Operativos lo que lo hace prácticamente universal.

1.5 Conclusiones parciales.

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- ✓ La investigación realizada sobre las soluciones existentes referente a los sistemas de visualización de información permitió concluir que estos no podían ser utilizados como solución al problema por las desventajas mencionadas durante el análisis de las mismas ya que en su mayoría presentan un elevado costo y están destinados a terminales específicas. También cuentan con características comunes que se pueden tener en cuenta en la implementación del subsistema de administración, enfocándolas en las necesidades actuales.

- ✓ Se caracterizaron las herramientas y tecnologías informáticas a emplear en el desarrollo del sistema, enunciándose de cada una de ellas las principales características y las facilidades que ofrecen al equipo de desarrollo, además de favorecer las posibles integraciones con otros subsistemas y garantizar futuras actualizaciones.

Capítulo II: Propuesta de solución para el Subsistema de Administración del Sistema de Visualización de Información al Viajero.

Introducción.

En este capítulo se realiza el análisis del subsistema que se desea construir. Se hace una conceptualización del entorno a través de un modelo de dominio y se realiza la captura de requisitos para obtener la especificación de los requisitos funcionales y no funcionales que el sistema debe tener. Se realiza la descripción de los casos de uso identificados además de los actores que interactúan con el sistema garantizando un mejor entendimiento para el desarrollo del sistema.

2.1 Sistema de Visualización de Información al Viajero.

El Sistema de Visualización de Información al Viajero tiene como objetivo principal informar al viajero sobre el medio de transporte que va a abordar. Este sistema está desarrollado para todas las terminales del país garantizando una mejor organización con respecto a la información que se administra.

Este sistema está integrado por los siguientes subsistemas:

- ✓ **Subsistema de Administración:** es el encargado de gestionar toda la información que se le hace llegar al viajero.
- ✓ **Subsistema de Transmisión:** es el que transmite por medio de los televisores toda la información administrada por el Subsistema de Administración.

2.2 Modelo de Dominio.

El modelo de dominio permite la representación de un conjunto de conceptos, y las relaciones que se establecen entre ellos ayudarán tanto a usuarios como a desarrolladores, a usar un vocabulario común, que les permita entender el contexto en que se desarrolla el sistema. Se emplea el modelo de dominio

para el Subsistema de Administración de Información al Viajero pues en el mismo no se precisa una estructura de los procesos de negocio que debe existir para emplear el modelo de negocio (Larman, 1999).

2.2.1 Conceptos y principales eventos del entorno.

En las terminales de transporte es fundamental que el viajero esté informado sobre el estado del transporte. Una de las formas que se utiliza para brindar información y mantener al viajero actualizado es mediante el **departamento de información** que hay en cada terminal. La información que se maneja de cada itinerario se puede brindar de diferentes formas: a través de un **altavoz**, por medio de **pizarras**, o de forma **personal.** Cada **itinerario** regula la entrada, la salida y cualquier movimiento de los **medios de transporte** que pueden ser **ómnibus**, **aviones**, **barcos** y **trenes.** En el tiempo que el viajero está en la terminal se transmiten elementos **audiovisuales** como **videos** pregrabados o **señales** externas para hacer más agradable su estancia. En la **Figura 1** se muestra el modelo de dominio.

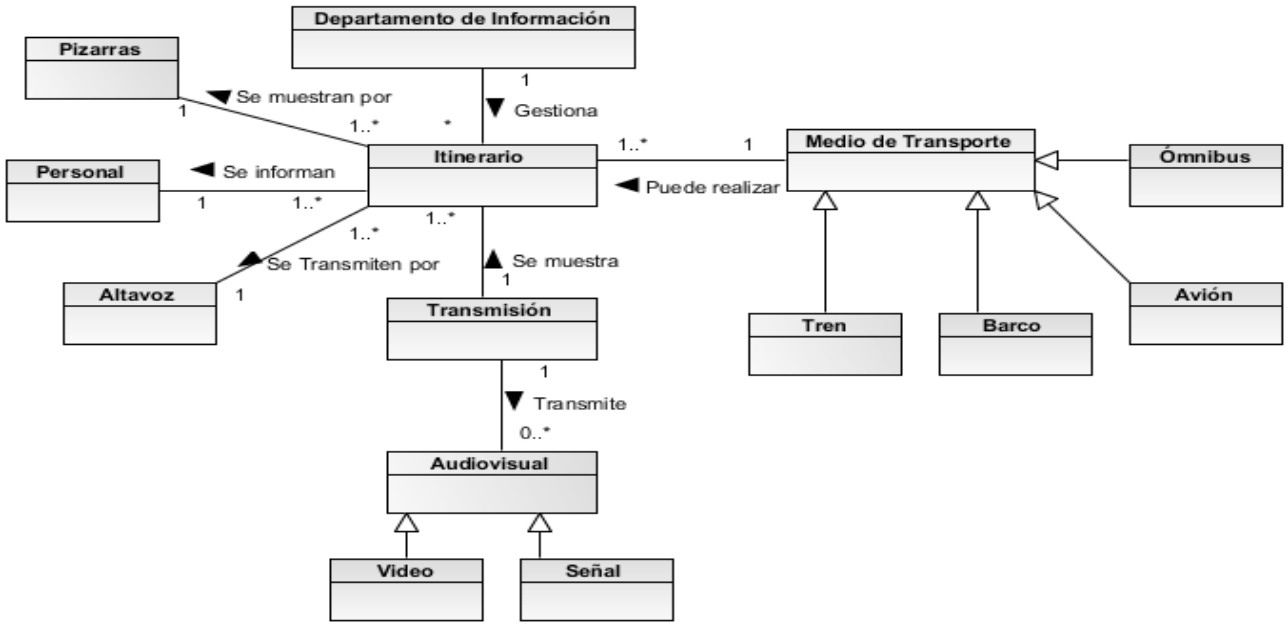

Figura 1: Diagrama de Dominio de la solución propuesta.

2.2.2 Glosario de Términos del Dominio.

A continuación se realiza una descripción de los conceptos para una mejor comprensión del diagrama del modelo de dominio.

- ✓ **Departamento de Información:** Departamento que hay en las terminales para dar información actualizada a los pasajeros sobre el medio de transporte que va a abordar.
- ✓ **Pizarras:** Medio que se utiliza en las terminales para mostrar información.
- ✓ **Personal:** Forma en que se da información en algunas terminales de transporte.
- ✓ **Altavoz:** Equipo que se utiliza para dar mensajes de alerta sobre el medio de transporte o alguna otra información de carácter relevante.
- ✓ **Itinerario:** Camino previsto por donde debe discurrir un recorrido o viaje.
- ✓ **Transmisión**: Envío y recepción de la información a través de un equipo informático por medio de señales encaminadas en un canal.
- ✓ **Audiovisual**: Es el concepto que une lo auditivo y lo visual (imagen y sonido).
- ✓ **Video**: Sistema de grabación de imágenes y sonidos en una cinta que después pueden reproducirse en un televisor.
- ✓ **Señales**: Cualquier canal de televisión existente que pertenece a una entidad en particular.

2.3 Especificación de los requisitos de software.

El propósito fundamental del flujo de trabajo de los requisitos es guiar el desarrollo hacia el sistema correcto. Esto se consigue mediante una descripción de los requisitos del sistema (las condiciones o capacidades que el sistema debe cumplir) suficientemente buena como para que pueda llegarse a un acuerdo entre el cliente (incluyendo a los usuarios) y los desarrolladores sobre que debe y que no debe hacer el sistema (Botta, 2010).

2.3.1 Captura de requisitos.

La captura de requisitos guía el desarrollo hacia el sistema correcto. Para llevar a cabo la captura de requisitos se aplicó la técnica de tormentas de ideas. Como técnica de captura de requisitos es sencilla de usar y de aplicar, teniendo en cuenta que la cantidad de personas que participen sean de un promedio de 10 personas. Su objetivo principal es que todo el personal muestre sus ideas de forma libre llegando a un acuerdo. Esta técnica permitió al equipo de desarrollo definir cómo llegar a cabo la administración de la información de forma organizada y eficiente.

Los requisitos de software se clasifican en 2 categorías: los requisitos funcionales y los requisitos no funcionales.

2.3.2 Requisitos Funcionales.

Un requisito funcional es una característica requerida del sistema que expresa una capacidad de acción del mismo (una funcionalidad); generalmente expresada en una declaración en forma verbal. Se traducen directamente en casos de uso. A continuación se describen los requisitos arquitectónicamente significativos identificados en el Subsistema de Administración para el Sistema de Visualización de Información al Viajero. Para mayor información puede consultar el documento de especificación de requisitos y ver el **Anexo 1**.

R11 Adicionar itinerario.
El subsistema permitirá adicionar itinerario con los siguientes datos:

- ✓ Nombre (Formato: Alfabético).
- ✓ Origen (Formato: Alfabético).
- ✓ Destino (Formato: Alfabético).
- ✓ Transporte (Formato: Alfabético).
- ✓ Estado (Formato: Alfabético).
- ✓ Región (Formato: Alfabético).
- ✓ Hora de llegada (Formato: Numérico).
- ✓ Hora de salida (Formato: Numérico).
- ✓ Precio para niño (Formato: Numérico).

- ✓ Precio para adulto (Formato: Numérico).
- ✓ Idioma (Formato: Alfabético).

R12 Editar itinerario.

El subsistema permitirá modificar el itinerario con los siguientes datos:

- ✓ Nombre (Formato: Alfabético).
- ✓ Origen (Formato: Alfabético).
- ✓ Destino (Formato: Alfabético).
- ✓ Transporte (Formato: Alfabético).
- ✓ Estado (Formato: Alfabético).
- ✓ Región (Formato: Alfabético).
- ✓ Hora de llegada (Formato: Numérico).
- ✓ Hora de salida (Formato: Numérico).
- ✓ Precio para niño (Formato: Numérico).
- ✓ Precio para adulto (Formato: Numérico).
- ✓ Idioma (Formato: Alfabético).

R13 Eliminar itinerario.

El subsistema permitirá eliminar un itinerario seleccionándolo y dando clic en el botón eliminar.

R14 Mostrar itinerario.

El subsistema permitirá mostrar el listado de todos los itinerarios y la posibilidad de seleccionar uno de ellos. Además de las opciones Adicionar, Eliminar y Editar.

R15 Filtrar itinerario.

El sistema debe ser capaz de filtrar itinerarios de acuerdo a un criterio de búsqueda.

R16 Asignar tramo.

El subsistema permitirá adicionar un tramo con los siguientes datos:

- ✓ Origen (Formato: Alfabético).
- ✓ Destino (Formato: Alfabético).
- ✓ Precio de niño (Formato: Numérico).

- ✓ Precio de adulto (Formato: Numérico).
- ✓ Tramo (Formato: Alfabético).

2.3.3. Requisitos No Funcionales.

Los requisitos no funcionales (RNF) son propiedades que debe cumplir el sistema. Son características que hacen al producto, atractivo, usable, rápido y confiable. En muchos casos los requisitos no funcionales son fundamentales en el éxito del producto. Definen que el sistema tenga una interfaz amigable y de fácil manejo para al usuario (Hernandez González, 2010). Los requisitos no funcionales se clasifican en múltiples categorías, a continuación se mencionan aquellos necesarios para el Subsistema de Administración.

Requisitos de usabilidad: La aplicación debe configurarse y administrarse por un usuario administrador que debe tener conocimientos medios en informática y de forma general debe brindar gran facilidad de uso para los demás usuarios con poca experiencia con las computadoras, pero con nivel calificado. El Subsistema de Administración para el Sistema de Visualización de Información al Viajero debe mostrar mensajes al usuario, que le ayuden a llevar a cabo la tarea que realiza. Se debe hacer uso de botones con imágenes que indiquen de modo intuitivo la función que realizan, así mismo los controles visuales deben mostrar mensajes aclarando sus propósitos. Las funcionalidades deben ser claras y se debe mostrar la información de forma lógica y correctamente estructurada. Se requiere que pueda utilizarse por personas que hablen diferentes idiomas.

Requisitos de seguridad: Identificar al usuario antes de que este realice cualquier acción en el portal. Preservar la información y garantizar el acceso a la misma por aquellos que tienen el derecho a manipularla y trabajar con ella. Asegurar que las funciones del sistema sean visibles según el nivel de acceso que tenga el usuario autenticado. Se usa el algoritmo Sha512 para la autenticación de manera segura.

Requisitos de confiabilidad: Se debe garantizar un tratamiento adecuado de las excepciones y validación de las entradas del usuario.

Requisitos de portabilidad: El sistema deberá funcionar en los sistemas operativos GNU[20]/Linux o Windows, garantizando de esta manera un producto multiplataforma.

Restricciones en el diseño y la implementación: Para el Subsistema de Administración del Sistema de Visualización de Información al Viajero se debe utilizar:

- ✓ Apache v2.4.4 como servidor web.
- ✓ PostgreSQL v9.0 o superior como sistema gestor de base de datos.
- ✓ Se debe acceder a través de cualquier navegador, fundamentalmente Mozilla Firefox v14.
- ✓ Se deben usar como lenguajes de programación PHP v5.4, JavaScript v1.2 y framework de presentación se utilizará JQuery v1.9.1.

Requisitos de hardware:

Pc Servidor

- Intel Pentium4 2.0 GHz.
- 2 GB de RAM.[21]
- 500 GB de espacio libre en el disco duro.
- Tarjeta de red.

Pc Cliente

- Intel Pentium4 1.0 GHz.
- 512 MB de RAM.
- Tarjeta de red.

2.4 Modelado del sistema.

El Modelo de Caso de Uso del Sistema permite que los desarrolladores de software y clientes lleguen a un acuerdo sobre las condiciones y posibilidades que debe cumplir el sistema. Proporciona la entrada fundamental para el análisis, el diseño y las pruebas (Jacobson, et al., 2000).

[20] Acrónimo recursivo que significa "GNU No es Unix".

[21] Memoria de acceso aleatorio.

2.4.1 Actores del Sistema.

Encontrar los actores es uno de los primeros pasos en la definición del uso del sistema. Cada tipo de fenómeno externo con el que debe interactuar el sistema está representado por un actor. Se debe definir cada actor especificando una breve descripción que incluya el área de responsabilidad del actor. Como los actores representan elementos externos al sistema, no es necesario describirlos en detalle (IBM, 2006).

Durante el desarrollo de la aplicación se definieron una serie de actores, los mismos se describen a continuación en la **Tabla 1**:

Tabla 1: Actores que interactúan con el Sistema.

Actor	Objetivo
Administrador	Es el encargado de gestionar los usuarios que van a interactuar con el sistema, además de administrar toda la información que se transmitirá en el canal.
Operador	Es la encargada de gestionar toda la información que va a ser transmitida en el canal.

2.4.2 Diagrama de Casos de Uso del Subsistema Administración.

Un Diagrama de Casos de Uso (CU) representa la comunicación y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. Los casos de uso ayudan a comunicar de manera eficiente las funciones del sistema al cliente. Para lograr una comunicación más eficiente, los casos de uso deben describirse de manera natural utilizando el lenguaje del cliente.

A partir de la identificación del actor que interactúa con la aplicación, así como la recopilación del conjunto de funcionalidades escritas en forma de requisitos, que a su vez se agruparon en CU según sus peculiaridades, se conforma el Modelo de CU que se representa en la **Figura 2** (IBM, 2006).

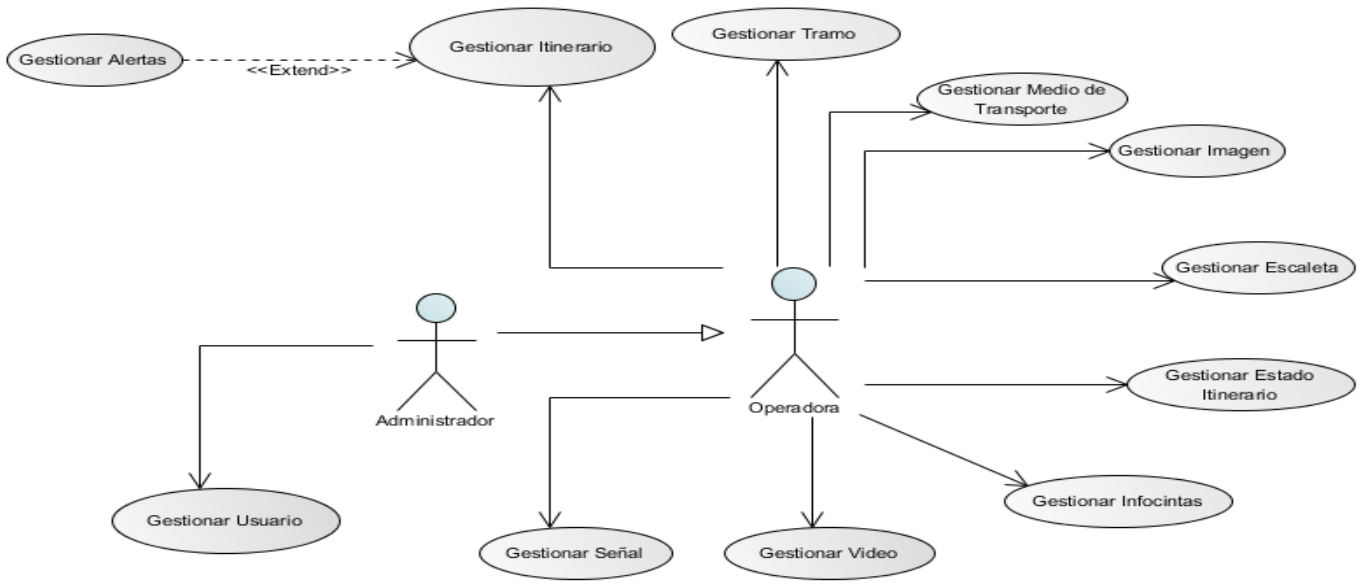

Figura 2: Diagrama de CU Subsistema de Administración para Sistema de Visualización de Información al Viajero.

2.4.3 Casos de Uso del Subsistema de Administración.

Los casos de uso son el componente clave del modelado. Permiten ilustrar las funcionalidades de un sistema y la relación con el actor para cumplir un objetivo. Definir un caso de uso es una de las tareas del flujo de trabajo de requisitos en RUP teniendo como objetivo describir uno o varios de los flujos de sucesos del caso de uso con el detalle suficiente para que el desarrollo de software pueda empezar en él (IBM, 2006).

Para entender la funcionalidad asociada a cada caso de uso no es suficiente con la representación gráfica del diagrama de casos de uso del sistema, es necesaria la descripción textual de cada caso de uso donde se especifican de manera clara todas las acciones para la realización del mismo. Para la realización del Subsistema de Administración para el Sistema de Visualización de Información al Viajero se identificaron 56 requisitos funcionales lo cual se agruparon en 11 casos de uso. En la **Tabla 2** se describe el caso de

uso arquitectónicamente significativo de la solución propuesta. Para una mejor comprensión, dirigirse al documento de especificación de casos de uso del presente trabajo.

Tabla 2: Descripción del CUS Gestionar Itinerario.

Caso de Uso	Gestionar Itinerario.
Objetivo	Adiciona, edita, muestra, elimina y filtra los itinerarios en el sistema. Además adiciona una alerta y un tramo al itinerario.
Actores	Operadora y Administrador.
Resumen	El caso de uso se inicia cuando el usuario adiciona, edita, muestra o elimina un itinerario y también cuando adiciona una alerta y un tramo. Termina cuando se haya realizado alguna de estas operaciones.
Precondiciones	El usuario debe estar autenticado como Operadora o Administrador.
Postcondiciones	Itinerario agregado. Bloque agregado. Bloque modificado. Itinerario modificado. Bloque eliminado. Listado de bloques filtrado. Itinerario eliminado. Itinerario mostrado. Itinerario filtrado. Tramo asignado.
Referencia	R11, R12, R13, R14, R15 y R16.
Complejidad	Alta

Flujo normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. Puede realizar las siguientes acciones: a)Mostrar itinerario, ver sección 1: Mostrar itinerario. b) Crear nuevo itinerario, ver sección 2: Adicionar itinerario. c) Editar itinerario, ver sección 3: Editar itinerario.	

d) Eliminar itinerario, ver sección 4: Eliminar itinerario. e) Filtrar itinerario, ver sección 5: Filtrar itinerario f) Asignar tramo, ver sección 6: Asignar tramo.	
Sección "Mostrar itinerario"	
Flujo Normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario elige la opción Gestionar Itinerario.	2. El sistema muestra el listado actualizado de todos los itinerarios registrados y la posibilidad de seleccionar uno de ellos. Además de las opciones Adicionar, Editar, Eliminar y Filtrar. Además de Adicionar alerta y Adicionar tramos.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1. El sistema no puede mostrar el listado de los itinerarios al no existir ninguno registrado.
Sección "Adicionar itinerario"	
Flujo Normal del Evento	
Acción del Actor	**Respuesta del Sistema**
1. El usuario accede a la opción Adicionar itinerario.	2. Muestra la interfaz para agregar un itinerario, con los campos nombre, origen, destino, transporte, estado, hora de salida, hora de llegada, precio de niños, precio de adultos, región e idioma.
3. El usuario inserta los datos del nuevo itinerario.	4. El sistema verifica la completitud de los datos para saber si la información está correcta.

5. Presiona el botón Aceptar.	6. El sistema añade el itinerario.
	7. Muestra la lista de itinerarios con los cambios realizados. Termina el CU. Si el usuario desea adicionar alguna alerta al itinerario, ir al caso de uso extendido Gestionar Alerta.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 En caso de que los datos introducidos estén incorrectos o se deje algún campo en blanco se muestra un mensaje de error y se remite al paso 3.
Sección "Editar itinerario"	
Flujo Normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario accede a la opción Editar itinerario.	2. El sistema busca el itinerario y muestra todos los campos que contiene para que se pueda modificar.
3. El usuario edita los datos que desea cambiar.	4. El sistema verifica la completitud de los datos para saber si la información está correcta.
5. Presiona el botón Aceptar.	6. El sistema añade el itinerario modificado.
	7. Muestra la lista de itinerarios con los cambios realizados. Termina el CU.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	4.1 En caso de que los datos introducidos estén incorrectos se muestra un mensaje de error.
Sección "Eliminar itinerario"	

Flujo Normal de evento	
Acción del Actor	**Acción del Sistema**
1. El usuario presiona el botón Eliminar de un itinerario determinado.	2. Se muestra un mensaje de confirmación.
3. Presiona el botón Aceptar.	4. Elimina el itinerario del listado de itinerarios. Termina el CU.
Flujos Alternos	
Acción del Actor	**Acción del Sistema**
3.1 Si el usuario selecciona Cancelar se termina el escenario.	
Sección "Filtrar itinerario"	
Flujo Normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. El usuario introduce los valores que desee en los campos nombre, origen, destino, transporte, estado, región e idioma.	2. Muestra un listado con los itinerarios que se correspondan con los criterios de búsqueda introducidos. Termina el CU.
Flujos Alternos	
Acción del Actor	**Respuesta del Sistema**
	2.1. El sistema no puede filtrar el listado de itinerarios al no existir ninguno registrado.
Sección "Asignar tramo"	
Flujo Normal del Evento	
Acción del Actor	**Acción del Sistema**

1. El usuario accede a la opción Asignar tramo.	2. El sistema muestra un listado de los tramos disponibles.
3. El usuario selecciona el tramo.	4. El sistema asigna el tramo y muestra un listado actualizado con todos los tramos asignados al itinerario.

2.5 Conclusiones Parciales.

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- ✓ Realizar el diagrama del modelo de dominio y la descripción del flujo de dominio permitió establecer un punto de partida para lograr un correcto diseño del sistema, garantizando además la obtención de un mejor razonamiento del problema a resolver mediante la presentación de un diagrama y los principales conceptos que intervienen.
- ✓ Se identificaron los requisitos de software funcionales y no funcionales, para un total de 56 requisitos funcionales que fueron agrupados en 11 casos de uso, de ellos 1 arquitectónicamente significativo y el resto secundarios, permitiéndole a los desarrolladores un mejor entendimiento de la aplicación que se desea desarrollar.
- ✓ Se identificaron dos actores: Administrador y Operador que interactúan con la aplicación, obteniéndose finalmente el diagrama de casos de uso del sistema, lo cual facilitó la realización de las especificaciones que permitirán a los desarrolladores conocer el flujo de cada una de las funcionalidades.

Capítulo III: Diseño del Subsistema de Administración del Sistema de Visualización de Información al Viajero.

Introducción

En el presente capítulo se muestran los diferentes diagramas que propone la metodología de desarrollo en la fase de construcción, para facilitar el desarrollo del sistema. Se especifican los patrones de arquitectura y diseño que utiliza el *framework* Symfony2, modelándose a través de los diagramas de clases del diseño.

3.1 Descripción de la arquitectura.

En un sistema informático la arquitectura es el conjunto de decisiones significativas sobre la organización de un software. Para describir y diferenciar una arquitectura se hace necesario emplear estilos arquitectónicos que ayuden a definir una estructura para todos los componentes del sistema.

3.1.1 Patrón arquitectónico.

Los patrones arquitectónicos definen la estructura general del software, indican las relaciones entre los subsistemas y los componentes del software y definen las reglas para especificar las relaciones entre los elementos (Clases, paquetes, subsistemas) de la arquitectura (Pressman, 2005). Se hace necesario emplear patrones arquitectónicos para describir y diferenciar una arquitectura de tal manera que ayuden a definir una estructura para todos los componentes del sistema.

Symfony utiliza el patrón arquitectónico Modelo Vista Controlador (MVC) garantizando un modelo, una vista y una controladora de forma independiente. El mismo accede a realizar modificaciones en cada una de las partes sin afectar las demás, permitiendo que el desarrollo de aplicaciones sea rápido y sencillo.
La **Figura 3** se muestra el funcionamiento interno de Symfony2 con la utilización del patrón arquitectónico MVC.

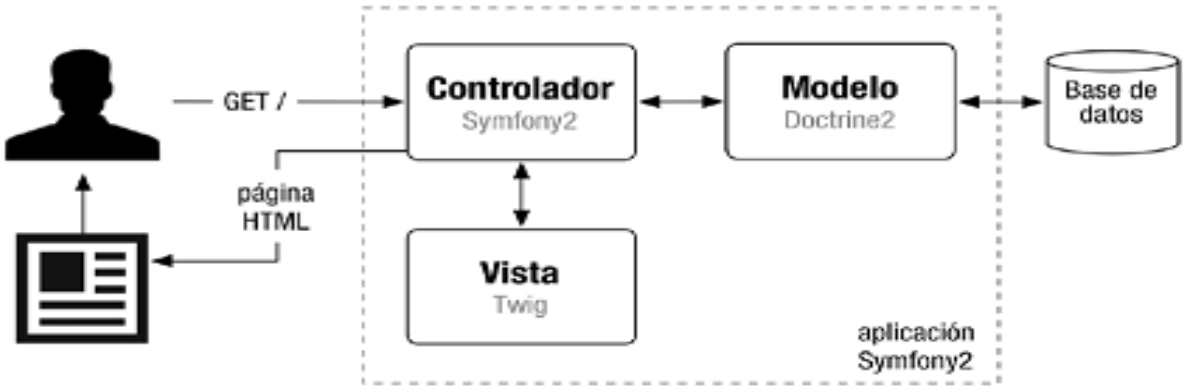

Figura 3: Esquema simplificado del funcionamiento interno de Symfony2 (Eguiluz, 2013).

Modelo: Encapsula los datos y las funcionalidades. Es independiente de cualquier representación de salida y comportamiento de entrada. Representa la información con la que trabaja la aplicación, es decir, su lógica de negocio. Se encarga de la abstracción de la lógica relacionada con los datos. En Symfony, el acceso y la modificación de los datos se realizan mediante objetos. Doctrine2 es el motor que se encarga de esta generación automática para construir sus clases, creando la estructura y generando el código de las mismas. Las clases y archivos relacionados con el modelo se guardan en el directorio *src/mi_Bundle/Entity/.*

Vista: Transforma el modelo en una página web que permite al usuario interactuar con ella. Muestra la información al usuario. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador. La vista en Symfony2 está formada por plantillas *twig*[22] que se almacenan en directorio *src/mi_Bundle/Resources /Views/*.

Controlador: Es el intermediario entre el modelo y a la vista. Se encarga de procesar las interacciones del usuario y realizar los cambios apropiados en el modelo o la vista. En Symfony2 todas las peticiones se realizan a través del controlador frontal *app.php*, el cual a través del enrutamiento delega las responsabilidades en las clases *Controller* de cada *bundle*.

[22] Es un motor de plantillas para el lenguaje de programación PHP.

3.1.2 Patrones de diseño.

Se aplican a un elemento específico del diseño como un agregado de componentes para resolver algún problema de diseño, relaciones entre los componentes o los mecanismos para efectuar la comunicación de componente a componente.

Patrones para asignar responsabilidades (GRASP).

Describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones, además constituyen un apoyo para entender el diseño. A continuación se describe la solución que brinda estos patrones y el problema que resuelve.

- ✓ Experto: este patrón plantea que se debe asignar una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad (Larman, 2003). Este patrón se evidencia con la utilización de *Doctrine2*, que es la librería externa que se utiliza para realizar la capa de abstracción en el modelo, encapsula toda la lógica de los datos y poseen todas las funcionalidades comunes de las entidades.

- ✓ Creador: este patrón plantea que se debe guiar la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. Se utiliza este patrón debido a la característica del sistema de ser orientado a objetos (Larman, 2003). Este patrón se evidencia en la clase *DefaultController*. En esta clase se crean los objetos de las clases que representan las entidades, evidenciando de este modo que la clase *DefaultController* es "creador" de dichas entidades.

- ✓ Controlador: este patrón plantea que se debe asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema, a una clase que represente una de las siguientes opciones (Larman, 2003). Sirve de intermediario entre una clase interfaz y la clase que contiene el algoritmo de la funcionalidad. El controlador es una parte vital de todo proyecto web ya que es el encargado de procesar las diferentes peticiones hechas por el cliente y se utiliza en el sistema con esos fines. Cada entidad (Infocinta, Itinerario, Escaleta etc.) con que cuenta el subsistema que se implementa tiene asociado un Controlador para el manejo de los eventos.

- ✓ Bajo Acoplamiento: asignar una responsabilidad para mantener bajo acoplamiento (una clase con bajo acoplamiento no depende de muchas otras). El acoplamiento es una medida de la fuerza con que una clase está conectada a otras. El patrón propone el diseño de clases más independientes, lo que reduce el impacto del cambio y facilita la reutilización en otros sistemas. Es un patrón evaluativo que el diseñador aplica al juzgar sus decisiones de diseño (Larman, 2003).

Patrones GOF

Los patrones GOF (*Gang of Four*, en español: Pandilla de los Cuatro) se clasifican en 3 categorías basadas en su propósito: creacionales, estructurales y de comportamiento.

- ✓ Creacionales: los patrones creacionales se encargan de la creación de instancias de los objetos. Abstraen la forma en que se crean los objetos, permitiendo tratar las clases a crear de forma genérica, dejando para después la decisión de qué clase crear o cómo crearla.

- ✓ Estructurales: los patrones estructurales se ocupan de cómo las clases y objetos se combinan para formar grandes estructuras y proporcionar nuevas funcionalidades. Tratan de conseguir que los cambios en los requisitos de la aplicación no ocasionen cambios en las relaciones entre los objetos.

- ✓ Comportamiento: los patrones de comportamiento plantean la interacción y cooperación entre las clases. Son utilizados para organizar, manejar y combinar comportamientos.

A continuación se describe el patrón GOF aplicado en el diseño del Subsistema de Administración:

- ✓ Decorador: Añade responsabilidades a un objeto de una forma dinámica y transparente, brindando una alternativa flexible a la herencia. En Symfony2 el archivo *base.html.twig* almacena el código HTML que es común a todas las páginas de la aplicación, para no tener que repetirlo en cada página. El contenido de las plantillas se integra en este archivo.

3.2 Modelo de Diseño.

El modelo de diseño es una abstracción de la implementación del sistema. Se utiliza para concebir y documentar el diseño del sistema de software. Es un producto de trabajo integral y compuesto que contiene los artefactos, clase de diseño, interfaz, paquete de diseño, subsistema de diseño, sucesos, señal, ejecución de guiones de uso, operación, realización de una operación, y componente de servicio (IBM, 2006).

3.2.1 Diagrama de Clases del Diseño.

Un diagrama de clases proporciona una perspectiva estática que representa el diseño estructural del sistema mostrando un conjunto de clases, sus atributos y las relaciones entre ellos (Larman, 2003). Los diagramas de clases del diseño (DCD) son subproductos del modelo de diseño, para el presente trabajo fue necesario personalizarlos haciendo uso de las posibilidades permitidas por la metodología RUP, para así lograr una mejor representación del diseño del nuevo Subsistema de Administración.

A continuación se representan los elementos considerados esenciales, con los que el programador se va a relacionar para la implementación de cada una de las funcionalidades.

- ✓ **Modelo (Entity), Vista (Views), y Controlador (Controller):** representan paquetes lógicos en correspondencia con la arquitectura del *framework* como se representa en la **Figura 4**.

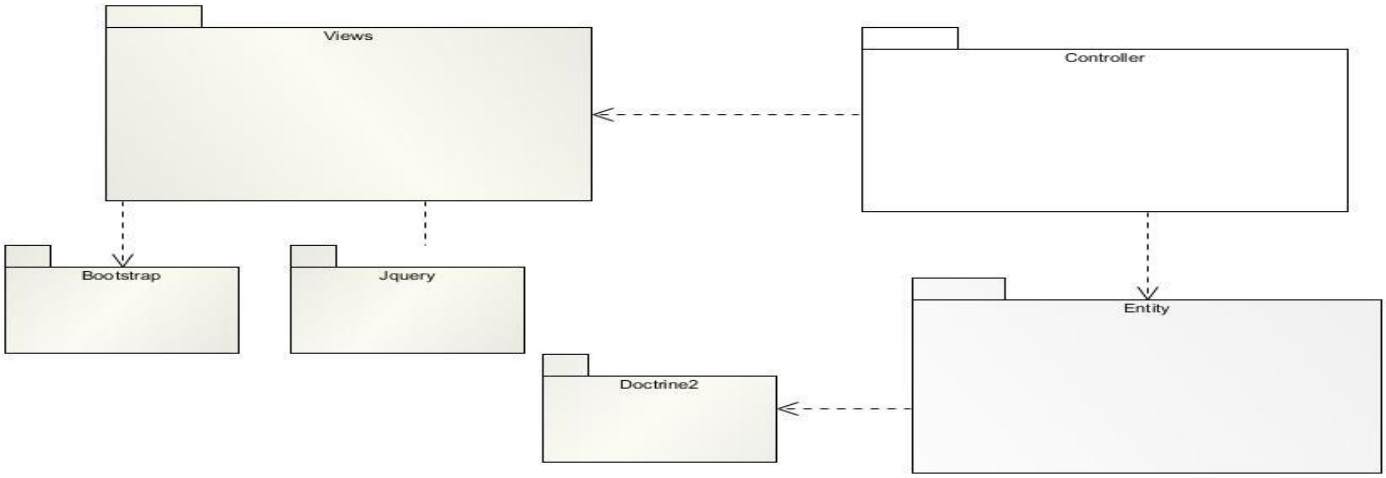

Figura 4: Diagrama de paquetes.

A continuación en la **Figura 5** se muestra el diagrama de clase del diseño del caso de uso Gestionar Itinerario. Los restantes diagramas de los casos de usos del sistema se pueden consultar en el expediente del proyecto Primicia.

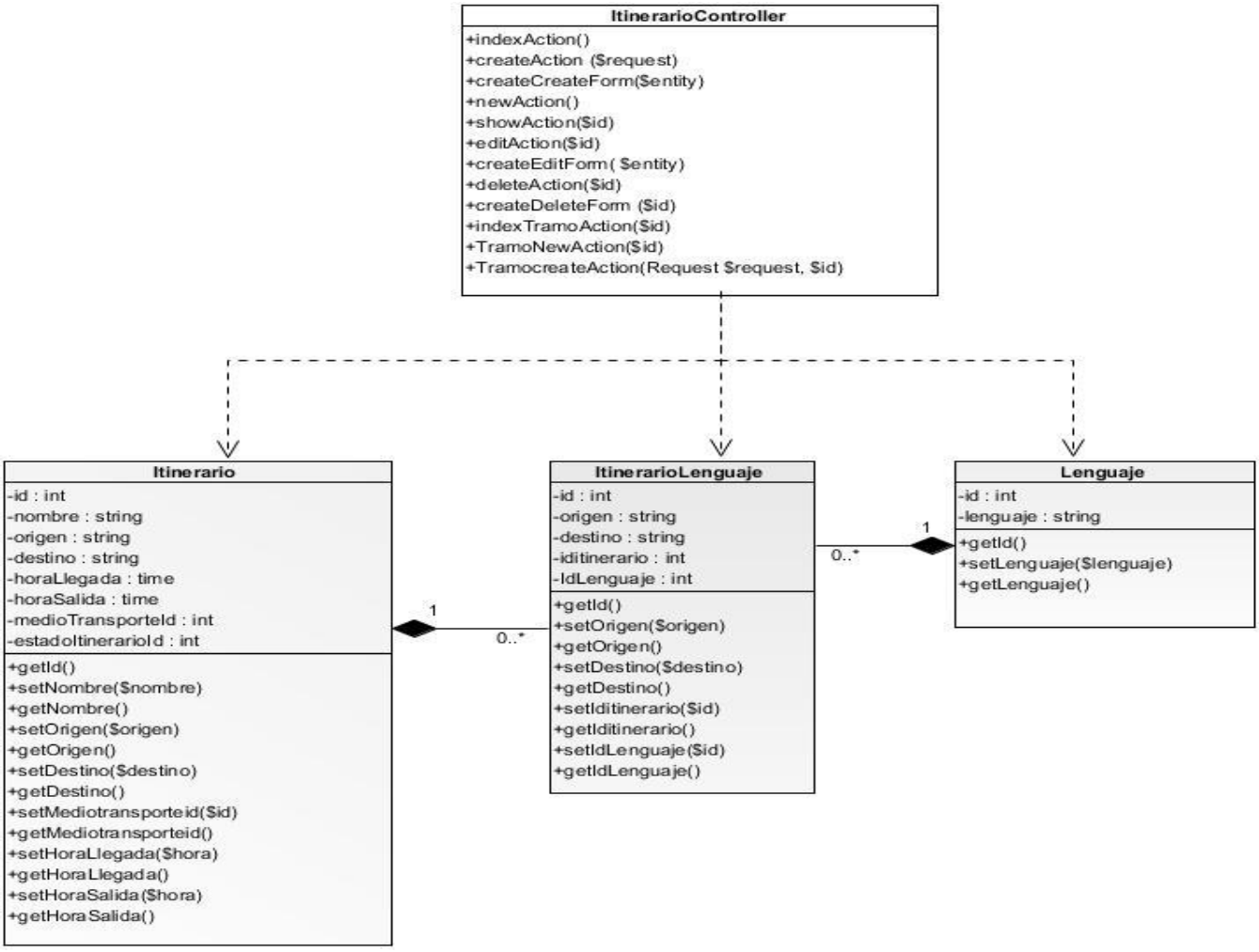

Figura 5: Diagrama de Clases del Diseño del CU Gestionar Itinerario

3.3 Modelo de Datos.

Un modelo de datos es la combinación de una colección de estructuras de datos, operadores o reglas de inferencia y de reglas de integridad, las cuales definen un conjunto de estados consistentes. El cual puede utilizarse como una herramienta para especificar los tipos de datos y la organización de los mismos.

Además para la manipulación de consultas y datos, así mismo es el elemento clave en el diseño de la arquitectura de un manejador de BD (Hernandez, 2010).

3.3.1 Diagrama de Clases Persistentes.

El objetivo fundamental del diseño de la base de datos es garantizar que los datos persistentes se almacenan de forma coherente y eficaz (IBM, 2006). Para diseñar la base de datos se utilizan el diagrama de clases persistentes y el diagrama entidad-relación. El diseño de la base de datos está compuesto por 19 tablas debidamente normalizadas para evitar la repetición de datos. A continuación se presentan en la **Figura 6** el Diagrama de Clases Persistentes de la solución propuesta.

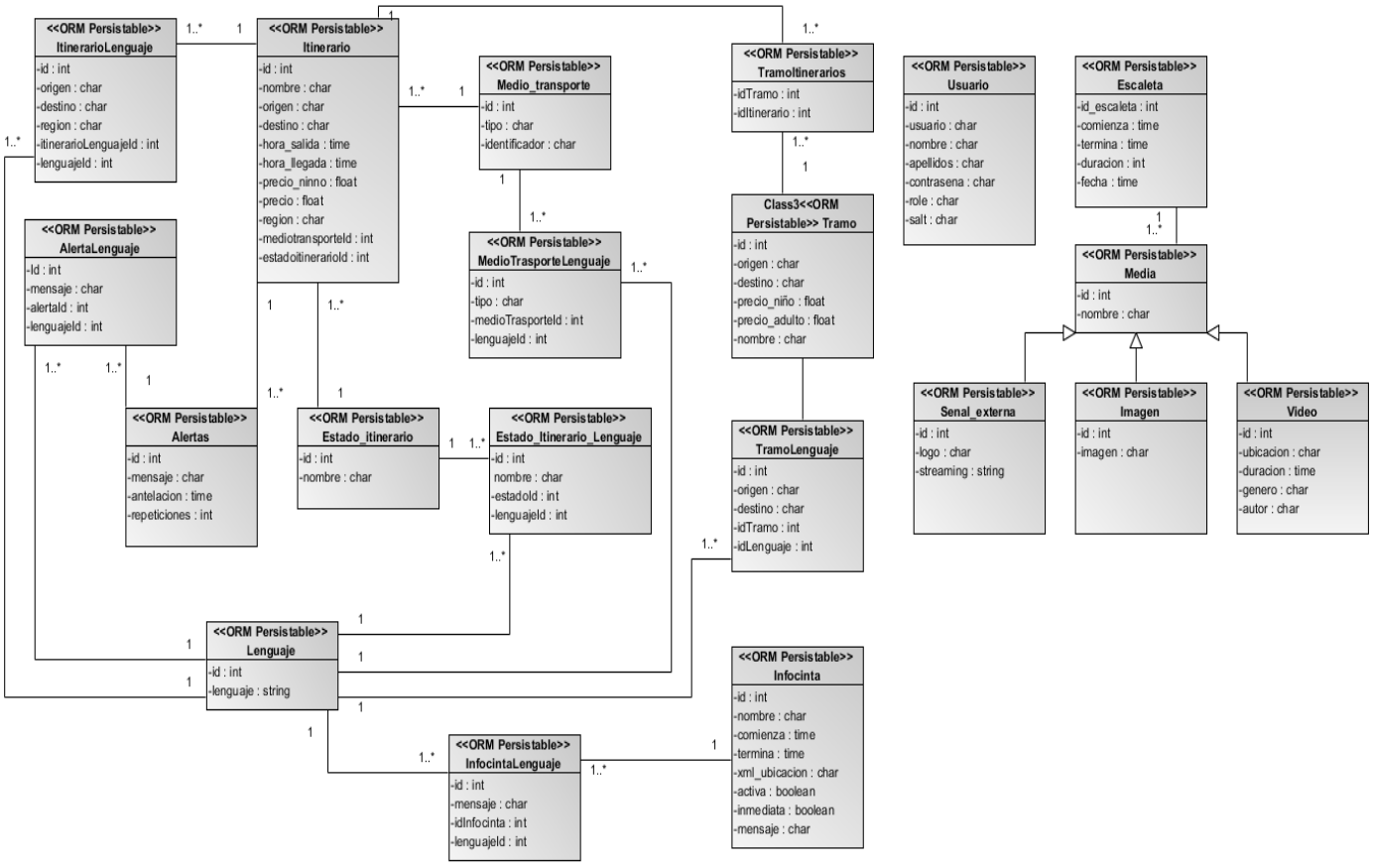

Figura 6: Diagrama de Clases Persistentes de la solución propuesta.

3.3.2 Modelo de Despliegue

Un modelo de despliegue es un modelo de objeto que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño (Jacobson, et al., 2000). En la **Figura 7** se representa el diagrama de despliegue del Subsistema de Administración para el Sistema Visualización de Información al Viajero.

El diagrama de despliegue está compuesto por:

- ✓ **Nodos:** Elementos de procesamiento con al menos un procesador, memoria y otros dispositivos.
- ✓ **Conectores:** Expresan el tipo de conector o protocolo utilizado entre el resto de los elementos del modelo.
- ✓ **Dispositivos:** Nodos estereotipados sin capacidad de procesamiento en el nivel de abstracción que se modela.

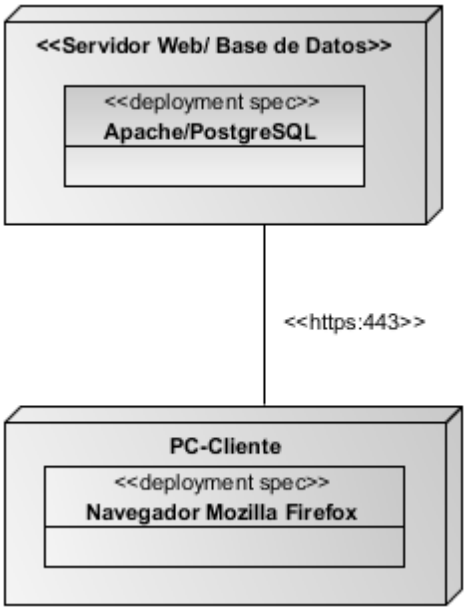

Figura 7: Diagrama de Despliegue de la Solución Propuesta.

Descripción de los Nodos.

- ✓ **Servidor de Base de Datos:** Es el nodo encargado de contener el servidor web y la base de datos, además de responder a las peticiones de los clientes.
- ✓ **PC Cliente:** Nodo cliente que interactúa con el Subsistema de Administración.
- ✓ **HTTPS:** Es un protocolo de aplicación basado en el protocolo HTTP[23], destinado a la transferencia segura de datos.

3.4 Conclusiones Parciales.

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- ✓ La identificación de los patrones de diseño GRASP y GOF permitió que la programación estuviera bien estructura. Además facilitó la asignación de responsabilidades logrando un diseño de software que sirva de apoyo a la implementación del sistema.
- ✓ Con la realización del diagrama de clases del diseño y el diagrama de paquetes se obtuvo una visión más exacta del sistema en términos de implementación, siendo de gran ayuda para el equipo de desarrollo.
- ✓ La realización del modelo de despliegue brinda una distribución completa del acople de los distintos componentes por lo que está compuesto el sistema.

[23] Protocolo para Transferencia de Hipertexto utilizado para transmitir hipertextos en la Web

Capítulo IV: Implementación y Prueba del Subsistema de Administración para el Sistema de Visualización de Información al Viajero.

Introducción.

El presente capítulo trata sobre los flujos de trabajo implementación y prueba. En este se desarrolla el diagrama de componentes donde se describen los elementos físicos del sistema y sus relaciones. También se aplican las pruebas al sistema para comprobar si cumple los requisitos del cliente, en este caso se aplicarán las pruebas de Caja Negra.

4.1 Modelo de Componentes.

Los modelos de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. Además de mostrar las dependencias entre elementos de implementación y los correspondientes elementos de diseños que son implementados (Cockburn, 2005). En la **Figura 8** se representa el diagrama de componentes del CU Gestionar Itinerario.

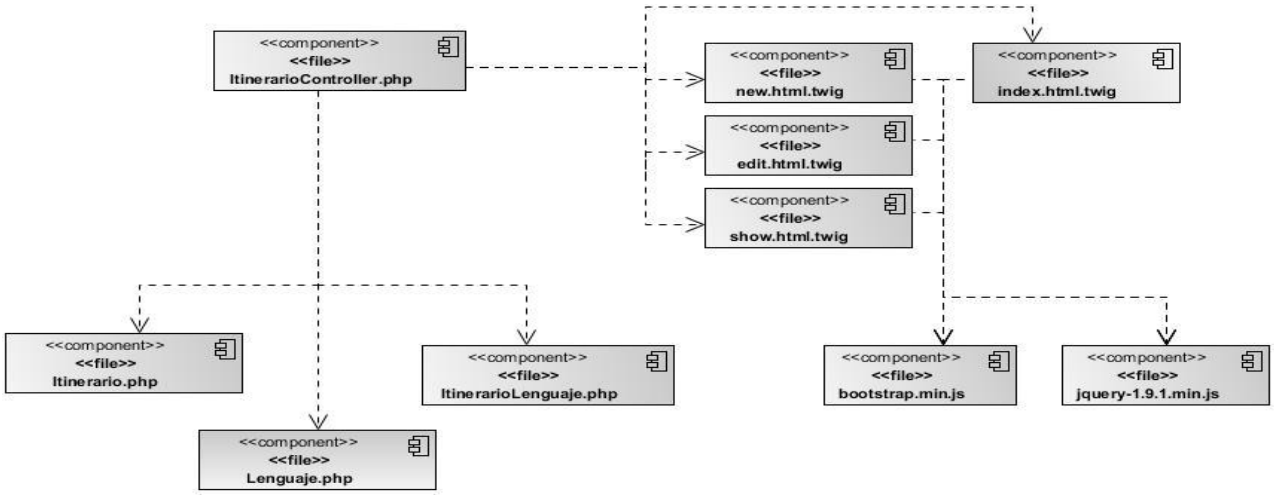

Figura 8: Diagrama de Componentes del CU Gestionar Itinerario.

4.2 Estándares de codificación

El estándar en el código de programación es muy importante en cualquier proyecto, principalmente si este involucra muchos desarrolladores. Esto asegura que el código sea de alta calidad, que contenga una cantidad baja de errores y sea fácil de mantener. Es importante que durante la codificación se consideren permanentemente los siguientes criterios de calidad:

- ✓ **Facilidad de Comunicación:** Proporcionar al usuario entradas y salidas fácilmente asimilables.
- ✓ **Autodescripción:** Proporcionar en el código, explicaciones sobre la implantación realizada**.**

El estándar utilizado para el desarrollo del sistema fue el siguiente:

4.3.1 Notación Camello

Consiste en escribir los identificadores de variables o funciones con la primera letra en mayúscula y el resto en minúscula o viceversa en dependencia de la variante que escojas. El nombre se debe porque los identificadores recuerdan las jorobas de un camello. Un ejemplo de su utilización se muestra en la **Figura 9**:

```
public function showAction($id)
{
    $em = $this->getDoctrine()->getManager();

    $entity = $em->getRepository('ViajeBundle:Itinerario')->find($id);

    if (!$entity) {
        throw $this->createNotFoundException('Unable to find Itinerario entity.');
    }

    $deleteForm = $this->createDeleteForm($id);

    return array(
        'entity'      => $entity,
        'delete_form' => $deleteForm->createView(),
    );
}
```

Figura 9: Ejemplo de notación Camello.

4.3 Pruebas del Subsistema Administración para el Sistema de Visualización de Información al Viajero.

Las pruebas constituyen una actividad en la cual el sistema o los componentes se ejecutan bajo ciertas condiciones o requisitos especificados donde los resultados son observados y registrados para realizar una evaluación de algún aspecto del sistema o del componente. Entre las buenas prácticas para realizar pruebas al sistema es que dichas pruebas se realicen por un actor externo que no sea el desarrollador de la aplicación (Quesada, 2007).

4.3.1 Objetivos de las pruebas

Las pruebas buscan encontrar los posibles fallos en la implementación, en la usabilidad y en la calidad de un programa determinada para validar su correcto funcionamiento. Entre los objetivos de la realización de las pruebas se tiene (Quesada, 2009):

- ✓ Diseñar los casos de prueba que permitan identificar diferentes clases de errores utilizando la menor cantidad de tiempo y esfuerzo.
- ✓ Detectar problemas en la aplicación.
- ✓ Verificar que todos los requisitos se han implementado satisfactoriamente.
- ✓ Identificar y asegurar que los problemas encontrados se han corregido antes de entregar el producto al usuario o cliente final.
- ✓ Mostrar hasta qué punto las funciones del software operan de acuerdo con las especificaciones y requisitos del cliente.
- ✓ Encontrar el mayor número de errores con la menor cantidad de tiempo y esfuerzo posibles.

4.3.2 Pruebas de integración.

Las pruebas de integración son ejecutadas para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación. Estas pruebas descubren errores o incompletitud en las especificaciones de las interfaces de los paquetes. Esta prueba debe ser responsabilidad de desarrolladores y de independientes, sin solaparse las pruebas. El objetivo es tomar

los componentes probados en unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño (Vegas, y otros, 2004).

El Sistema de Visualización de Información al Viajero está integrado por el Subsistema de Administración y el Subsistema de Transmisión. El Subsistema de Administración es el encargado de almacenar la información en la base de datos que será mostrada previamente por el Subsistema de Transmisión. Como el sistema está conformado por dos subsistemas, para realizarle las pruebas de integración se aplicó la estrategia de integración no incremental.

A continuación se describe el proceso de aplicar las pruebas centrándose en las funcionalidades: mostrar infocintas, reproducir medias y mostrar itinerario.

- Para comprobar la funcionalidad mostrar infocintas, se creó en el Subsistema de Administración una infocinta inmediata. Estos datos fueron enviados a la base de datos con la que trabaja también el Subsistema de Transmisión siendo el mismo el encargado de mostrarlos. A causa de que los datos enviados por el Subsistema de Administración no fueron correctamente obtenidos por el Subsistema de Transmisión, los datos de la infocinta no se mostraron. Para dar respuesta a esta *No conformidad* se tuvo en cuenta que todas funcionalidades encargadas de obtener la señal fueran modificadas. Al realizar nuevamente el mismo procedimiento se obtuvo una respuesta satisfactoria mostrándose la infocinta mediante dos transmisiones.

- En el caso de la funcionalidad reproducir medias, se creó una escaleta con una duración de 2 horas en el Subsistema de Administración. Las medias asignadas a la escaleta fueron transmitidas satisfactoriamente por el Subsistema de Transmisión mediante la duración planificada.

- Para probar la funcionalidad mostrar itinerario, en el Subsistema de Administración fueron creados algunos itinerarios, estos fueron mostrados satisfactoriamente por el Subsistema de Transmisión.

Para la generación de casos de prueba de integración, ya sea descendente o ascendente se utilizan técnicas de caja negra.

4.3.3 Pruebas funcionales o de caja negra.

Las pruebas de caja negra se llevan a cabo sobre la interfaz del software. Se centran principalmente en los requisitos funcionales del software. Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

La prueba de caja negra para Pressman (Pressman, 2005), intenta encontrar errores de las siguientes categorías:

- ✓ Funciones incorrectas o ausentes.
- ✓ Errores de interfaz.
- ✓ Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- ✓ Errores de rendimiento.
- ✓ Errores de inicialización y terminación.

Dentro del método de caja negra existen algunas técnicas como:

- ✓ Técnica de la Partición de Equivalencia: esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- ✓ Técnica del Análisis de Valores Límites: esta Técnica prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- ✓ Técnica de Grafos de Causa-Efecto: es una técnica que permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Para el Subsistema de Administración dentro del método de caja negra se aplicó la técnica de la Partición de Equivalencia.

4.3.3.1 Casos de Pruebas (CP).

Un caso de prueba es un conjunto de entradas, condiciones de ejecución y resultados esperados desarrollados para un objetivo particular. Los casos de pruebas se pueden derivar de los casos de usos

del sistema o de la realización de estos en el modelo de diseño, permitiendo así validar los requisitos funcionales del sistema.

Caso de Prueba del Caso de Uso Gestionar Usuario.

Descripción general: El caso de uso se inicia cuando el usuario adiciona, edita, muestra o elimina un usuario. Termina cuando se haya realizado alguna de estas operaciones.

Condiciones de Ejecución: No tiene.

Tabla 3: Descripción de variables del CU Gestionar Usuario.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
V1	Nombre	Campo de texto	No	Cualquier combinación de letras, números, guiones bajo y espacios.
V2	Apellidos	Campo de texto	No	Cualquier combinación de letras, y espacios.
V3	Usuario	Campo de texto	No	Cualquier combinación de letras, números y guiones bajo.
V4	Contraseña	Campo de texto	No	Cualquier combinación de letras, números, guiones bajo y espacios.
V5	Rol	Campo de texto	No	Cualquier combinación de letras, guiones bajo y espacios.

SC 1: Adicionar Usuario.

Tabla 4: Prueba del CU Gestionar Usuario, sección Adicionar Usuario.

Escenario	Descripción	V1	V2	V3	V4	V5	Respuesta del Sistema	Resultado de las Pruebas

EC1.1 Adicionar usuario satisfacto-riamente.	Se introduce en todos los campos, datos válidos.	V Zay	V De paz Cuba	V zayh	V admin	V Administra-dor	Refresca la página, con el usuario ya auten-ticado.	Satisfacto-rio
EC1.2 Adicionar usuario incorrecta-mente	Se introduce algún campo inválido.	I 45466	V Cuba	V Alih2	V Zay67	V Operadora	Se muestra un mensaje "Datos inválidos"	Satisfacto-rio
		V Elisa	I Sad45	V Elih2	V Za*34	V Operadora		
		V	V	I ___2 3	V	V		
		V	V	V	I	V		
EC 1.3 Adicionar con campos vacíos.	Se introduce algún campo vacío.	V	I Campo vacío	V	V	I Campo vacío	Se muestra el mensaje: "Los campos no pueden estar vacíos"	Satisfacto-rio

SC 2: Editar Usuario.

Tabla 5: Prueba del CU Gestionar Usuario, sección Editar Usuario.

Escenario	**Descripción**	**V1**	**V2**	**V3**	**V4**	**V5**	**Respuesta del Sistema**	**Resultado de las Pruebas**
EC1.1 Editar usuario satisfacto-riamente.	Se introduce en todos los campos, datos válidos.	V	V	V	V	V	Refresca la página, con el usuario ya auten-ticado.	Satisfacto-rio
EC1.2 Editar usuario	Se introduce algún campo	I	V	V	V	V	Se muestra un mensaje	Satisfacto-rio

incorrecta-mente	inválido.						"Datos inválidos"	
		V	I	V	V	V		
		V	V	I	V	V		
		V	V	V	I	V		
EC 1.3 Editar con campos vacíos.	Se introduce algún campo vacío.	V	I Campo vacío	V	V	I Campo vacío	Se muestra el mensaje: "Los campos no pueden estar vacíos"	Satisfacto-rio

SC 2: Eliminar Usuario.

Tabla 6: Prueba del CU Gestionar Usuario, sección Eliminar Usuario.

ID del Escenario	Descripción	Usuario Registrado	Respuesta del Sistema	Resultado de las Pruebas
EC1.1 Eliminar usuario satisfacto-riamente.	El administrador selecciona el usuario a eliminar.	V	Se elimina el usuario correctamente y se muestra un mensaje informando que la operación fue satisfactoria.	Satisfactorio

EC1.2 Eliminar usuario incorrecta-mente	El administrador no selecciona el usuario a eliminar.	I	Se muestra un mensaje indicando que el usuario no se pudo eliminar de forma satisfactoria.	Satisfactorio

Resultados de las pruebas de caja negra

En las pruebas de caja negra realizadas, de los 11 casos de uso se identificaron 11 casos de pruebas. En la primera iteración se detectaron 10 no conformidades a las cuales se les dio solución. En la segunda y última iteración se detectaron 5 no conformidades, las mismas se resolvieron en su totalidad. En las dos iteraciones efectuadas se detectaron un total de 15 no conformidades, las cuales en su mayoría respondían a errores de bajo impacto en el correcto funcionamiento del sistema y todas tuvieron solución un tiempo máximo de 4 días.

4.3.4 Pruebas de Usabilidad

La facilidad de uso es un punto de vital importancia porque un sistema puede ser rechazado por los usuarios si no encuentran sencillo su uso. Para la validación de los requisitos no funcionales de usabilidad se utilizó la Lista de Chequeo Usabilidad de Sitios Web, elaborada por el centro de calidad CALISOFT. Esta cuenta con varios puntos, los cuales serán clasificados antes de ser aplicados, teniendo en cuenta su pertinencia, además de tener un punto donde se referencia las observaciones en cuanto al aspecto a evaluar. Para mayor información puede consultar el expediente del proyecto Primicia.

A continuación se muestra en la **Figura 10** los resultados de las pruebas de usabilidad:

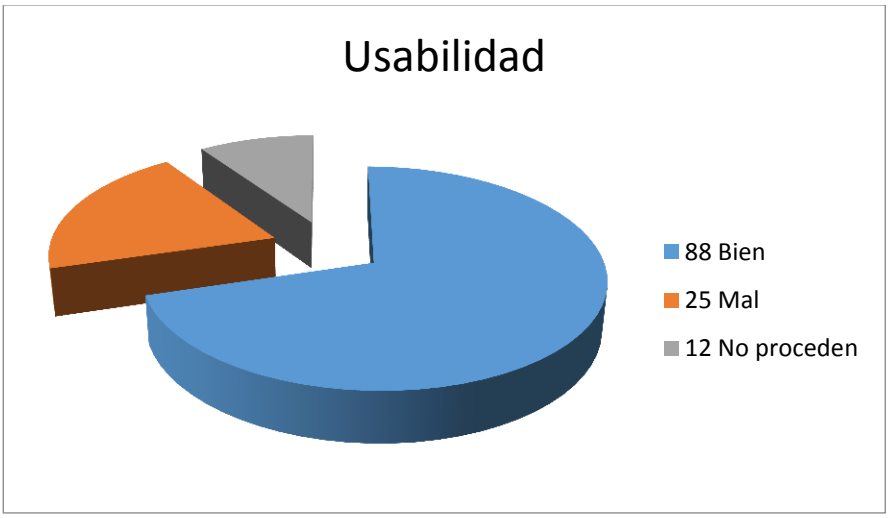

Figura 10: Resultado de las Pruebas de Usabilidad de los Requisitos No Funcionales en la primera iteración.

En la segunda iteración de las pruebas de usabilidad se erradicaron las 25 no conformidades detectadas en la primera iteración.

4.4 Conclusiones Parciales.

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones:

- ✓ La confección del modelo de componentes permitió conocer cómo interactúan los paquetes de clases y las diferentes capas de la aplicación dentro del marco de trabajo.
- ✓ El uso de los estándares de codificación garantizó que la implementación se realizara de manera organizada, permitiendo un mayor entendimiento de la misma.
- ✓ Se realizaron pruebas de integración, de usabilidad, de seguridad y de caja negra demostrando que las funciones de la aplicación son operativas, producen un resultado satisfactorio y el sistema cumple con los objetivos trazados.

Conclusiones Generales

A partir del desarrollo de la presente investigación se arribaron a las siguientes conclusiones:

- ✓ El análisis y estudio de las tecnologías y tendencias necesarias para el desarrollo del sistema permitió definir los lenguajes, herramientas y métodos a seguir para su implementación.
- ✓ El uso del marco de trabajo Symfony ayudó a confeccionar una propuesta de solución que se ajustara a las pautas de codificación definidas para el desarrollo de software.
- ✓ Se definieron 56 requisitos funcionales, 7 no funcionales, y 11 casos de uso con sus descripciones permitiendo definir las características del sistema y sirviendo de guía para los flujos y fases posteriores.
- ✓ Para comprobar la calidad y correcto funcionamiento del sistema, se diseñaron y ejecutaron casos de prueba, los que arrojaron resultados satisfactorios, demostrando el cumplimiento de los requisitos funcionales establecidos en la fase inicial del proceso de desarrollo del componente.
- ✓ El subsistema implementado como resultado de la investigación realizada será de gran utilidad en las terminales de transportación del país, garantizando una mejor organización sobre todo el proceso de transportación.

Recomendaciones

Luego de la presente investigación se recomienda:

- ✓ Incorporar la difusión de la información al viajero no sólo en las terminales sino a bordo del medio de transporte, si este lo permite.

- ✓ Realizar un manual de usuario para un mejor entendimiento del sistema.

Bibliografía y Referencias Bibliográficas

Fitzgerald, Avison. 2008. *Information systems development: methodologies, techniques and tools.* McGraw Hill : s.n., 2008.

Hernandez González, A. 2010. *REQUISITOS A PARTIR DEL MODELO DEL NEGOCIO. Ingeniería Industrial.* Ciudad de La Habana : s.n., 2010. pág. 5.

Linaria. 2012. Sistema de horario más práctico. [En línea] 2012. [Citado el: 1 de febrero de 2014.] http://www.swarco.com/latinamerica/Productos/Transporte-p%C3%BAblico/Sistemas-de-informaci%C3%B3n-de-pasajeros/LINARIA.

Alvares, Ángel Miguel y Monteiro, Lázaro. 2008. Desarrolladores Web. [En línea] 4 de diciembre de 2008. [Citado el: 3 de febrero de 2014.].

Alvares, Rubén. 2001. Desarrollo Web. [En línea] 1 de enero de 2001. [Citado el: 12 de enero de 2014.] http://www.desarrolloweb.com/articulos/239.php.

Álvarez de Zayas, Dr. Carlos. 2007. *METODOLOGIA DE LA INVESTIGACION CIENTIFICA.* Santiago de Cuba : s.n., 2007.

Álvarez, Miguel Ángel. 2010. *Manual de jQuery.* 2010.

Biología y Geología Interactiva. 2012. [En línea] 2012. http://biologiaygeologia.org/unidadbio/.../concepto_de_sistema.html.

Botta, Adrián. 2010. Resumen de Teoría de: Diseño de Sistemas. [En línea] 2010. [Citado el: 3 de Abril de 2014.] http://es.scribd.com/doc/85235594/9/FLUJO-DE-TRABAJO-DE-CAPTURA-DE-REQUISITOS.

Carrillo, Ignacio., Pérez, Robert. y Rodríguez, Andres. 2008. *Metodología de desarrollo del Software.* 2008.

Castro, Diéz, y otros. 2009. *Administración y Dirección.* McGraw-Hill Interamericana : s.n., 2009.

Catédras Mendoza, Nacira. 2008. Material didáctico. [En línea] 30 de enero de 2008. [Citado el: 12 de noviembre de 2013.] http://materialdidacticonm.wordpress.com.

Chiavenato, Idalberto. 2007. *Introducción a la Teoría General de la Administración.* s.l. : Séptima Edición, 2007.

Cockburn, A. 2005. *Crystal Clear: A Human-Powered Methodology for Small Teams.* 2005.

Dans, Enrique. 2007. *Reflexiones sobre las tecnologías de la información.* [En línea] Information Managemen, 18 de nobiembre de 2007. [Citado el: 7 de enero de 2014.] http://informationmanagement.wordpress.com/category/gestion/gestion-de-la-informacion/..

De la Torre, Aníbal. 2006. [En línea] 2006. [Citado el: 23 de febrero de 2014.] http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html..

Definición. 2008. Definición de Lenguaje de Programación. [En línea] 25 de marzo de 2008. [Citado el: 15 de febrero de 2014.] http://www.definicion.org/lenguaje-de-programacion.

Eguiluz, Javier. 2013. *Desarrollo web ágil con Symfony2.* 2013.

Eslava Muñoz, Javier Vicente. 2012. *Aprendiendo a programar paso a paso con C.* España : s.n., 2012. págs. 10-12. ISBN papel:978-84-686-1061-0.

Española, Real Academia. 2013. [En línea] 2013. [Citado el: 1 de octubre de 201.] http://es.thefreedictionary.com/itinerario.

—. 2013. [En línea] 2013. [Citado el: 1 de octubre de 2013.] http://lema.rae.es/drae/?val=informaci%C3%B3n.

Espinoza, Humberto. 2005. *Postgres una alternativa de DBMS Open Source.* s.l. : OPEN WORLD, 2005.

Falgueras Campderrich, Benet. 2008. *Ingeniería del software.* s.l. : UOC, 2008.

Giráldez Reyes, Raudel, Díaz Pérez, Maidelyn y Armas Peñas, Dayron. 2008. ACIMED. [En línea] mayo de 2008. [Citado el: 12 de febrero de 2014.].

Hernández León, Sayda, Cantillo , Rolando Alfredo y González, Coello. 2011. El proceso de investigación científica. Ciudad de La Habana : s.n., 2011.

Hernandez, Aleida. 2010. Institución Tecnológico de Colima. Departamento de Sistemas y Computación, Tutorial de Fundamentos y Base de Datos. [En línea] 2010. [Citado el: 22 de Marzo de 2014.] http://labredes.itcolima.edu.mx/fundamentosbd/sd_u2_1.htm.

IBM. 2006. IBM Corp. Ayuda del Rational (Español). *Rational Unified Process Versión.* [En línea] 2006. Versión 7.0.1.

IKUSI. 2013. [En línea] Web Corporativa, 2013. [Citado el: 29 de enero de 2014.] http://www.ikusi.com/es/ferrocarril-soluciones/sistema-de-videoinformaci%C3%B3n.

—. 2013. [En línea] 2013. [Citado el: 26 de enero de 2014.] http://www.ikusi.com/es/aeropuertos..

—. 2013. [En línea] Web Corporativa, 2013. [Citado el: 28 de enero de 2014.] http://www.ikusi.com/es/ferrocarril-soluciones/sistema-de-informaci%C3%B3n-al-viajero.

—. 2013. [En línea] Web Corporativa, 2013. [Citado el: 27 de enero de 2014.] http://www.ikusi.com/es/ferrocarril-soluciones/sistemas-de-ayuda-la-explotaci%C3%B3n-sae.

INDRA. 2010. *Transporte Terrestre y Ferroviario.* Madrid (España) : s.n., 2010. 35 – 28108.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* La Habana: Félix Varela : Volumen I, 2000.

Kroll, Per y Kruchten, Philippe. 2003. *The rational unified process made easy.* 2003. ISBN: 0-321-16609-4.

Lamarca Lapuente, María Jesús. 2011. Hipertexto. [En línea] 5 de diciembre de 2011. [Citado el: 27 de febrero de 2014.] http://www.hipertexto.info/documentos/xml.htm.

Larman, Craig. 1999. *UML Y PATRONES.* México : s.n., 1999.

—. 2003. *UML Y PATRONES. UNA INTRODUCCIÓN AL ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS Y AL PROCESO UNIFICADO.* Madrid: PEARSON EDUCACIÓN : SEGUNDA EDICIÓN, 2003. ISBN: 84-205-3438-2.

Montero, Rafael. 2008. Definicion.es. [En línea] 2008. [Citado el: 22 de diciembre de 2014.]

Pérez Eguíluz, Javier . 2009. *Introducción a CSS.* 2009. 0188-7297.

Pérez, Yaros. 2006. Análisis y Diseño de Sistemas. [En línea] 2006. [Citado el: 2 de Marzo de 2014.] http://www.geocities.ws/german_orta/10/adsb/tf_adbs.html.

Potenciar, Fabier, Francois y Zaninotto. 2007. *Symfony la guía definitiva.* 2007. págs. 24-29.

Pressman, Roger. 2005. *Ingeniería de Software. Un enfoque práctico.* España: Mc Graw Hil : s.n., 2005. 9701054733.

Quesada, Juan Antonio. 2007. *Pruebas del software.* 2007.

Quesada, Juan Antonio. 2009. *Pruebas del software.* 2009.

Rodríguez, Txema. 2012. Genbetadev. [En línea] 16 de junio de 2012. [Citado el: 17 de Marzo de 2014.] http://www.genbetadev.com/frameworks/bootstrap.

Visual Paradigm. 2006. [En línea] 2006. [Citado el: 3 de febrero de 2014.] http://www.visual paradigm.com/product/vpuml/.