Universidad de las Ciencias Informáticas

Facultad 1

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

MÓDULO DE SEGUIMIENTO DE OBJETIVOS EN TIEMPO REAL PARA EL SISTEMA DE GESTIÓN INTELIGENTE DEL TRASPORTE

Autor:

Julio César Baró Maza

Tutor(es):

Ing. Yaili Ledea Velázquez

Ing. Gerxis Sam Alcantara

Lic. Leonel Alejandro Araujo Taboada

La Habana, septiembre de 2020

DECLARACIÓN DE AUTORÍA

Declaración de autoría:

Declaro por este medio que yo **Julio César Baró Maza**, con carné de identidad **97112307587** soy el autor principal del trabajo titulado "**Módulo de seguimiento de objetivos en tiempo real para el sistema de gestión inteligente del trasporte.**" y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los ___14____ días del mes de __septiembre___ de __2020___

_____

Julio César Baró Maza

_____ _____

Ing. Yaili Ledea Velázquez Ing. Gerxis Sam Alcantara

_____

Lic. Leonel Alejandro Araujo Taboada

AGRADECIMIENTOS

Agradecimientos

Quiero agradecer primeramente a mi familia y amigos, especialmente a mis padres y hermano por su apoyo incondicional durante estos 5 años de estudio y sacrificio. También agradezco a mis tutores por sus consejos, ayuda y tenerme tanta paciencia ya que sin ellos no hubiera logrado todo esto. Agradezco además el apoyo de mis compañeros de clase, principalmente los de mi apartamento 126203 por los grandes momentos, a los cuales les deseo una larga, exitosa y prospera vida.

DEDICATORIA

Dedicatoria

Este trabajo de diploma va dedicado a toda mi familia y en especial a mis padres María Victoria y José Santiago, a mi hermano Erick Santiago y a mi abuelita Antonia, por su cariño, guía, comprensión y apoyo incondicional en todo momento de mi vida que me ayudaron a alcanzar esta meta y ser parte indispensable de la persona que hoy soy.

RESUMEN

Resumen

Los Sistemas de Control de Flotas constituyen una útil herramienta en el funcionamiento de entidades y organismos a nivel mundial que precisan de la vigilancia y seguimiento de sus activos móviles. Estos se han convertido en una de las principales bases en la gestión inteligente del transporte de las empresas contribuyendo a lograr un eficiente funcionamiento y control. Se desarrollan con el objetivo de conocer con precisión la ubicación de sus activos, dígase mercancías o medios de transportes, para en caso de eventualidades actuar con rapidez, exactitud, o por el simple hecho de tener información del estado del objetivo. El presente trabajo contiene la investigación de un Módulo de seguimiento de objetivos en tiempo real para el sistema de gestión inteligente del trasporte desarrollado por la empresa XETID ubicada en la Universidad de las Ciencias Informáticas (UCI) dirigido al control y seguimiento de activos pertenecientes a entidades que solicitan el servicio de dicha empresa. La aplicación desarrollada sobre el sistema GENESIG, producto de la nombrada entidad, fue creada sobre el marco de trabajo o framework OpenLayers en su versión 2, lo que permite llevar a cabo la gestión de la información de los mapas utilizados con más facilidad. Para el desarrollo de la investigación se empleó la metodología de desarrollo AUP en su variante UCI, posibilitando así la documentación de cada una de las etapas de vida del desarrollo del software. Además, se definieron las tecnologías y herramientas adecuadas para concretar la implementación de la aplicación propuesta.

Palabras claves: flota, gestión, herramientas, información, mapa, módulo, tecnologías.

ÍNDICE

ÍNDICE

INTRODUCCIÓN

Si hablamos de las TIC o Tecnologías de Información y Comunicaciones, nos referimos a un grupo diverso de prácticas, conocimientos y herramientas, vinculados con el consumo y la transmisión de la información y desarrollados a partir del cambio tecnológico vertiginoso que ha experimentado la humanidad en las últimas décadas, sobre todo a raíz de la aparición de Internet. No existe un concepto claro de las TICs, sin embargo, ya que este término se emplea de modo semejante al de la “Sociedad de la Información”, es decir, se usan para indicar el cambio de paradigma en la manera en que consumimos la información hoy en día, respecto a épocas pasadas. Esto tiene que ver con áreas tan distintas como las relaciones sociales, las finanzas corporativas, la industria del entretenimiento e incluso el trabajo cotidiano. Con ello se quiere decir que las nuevas Tecnologías de la Información y las Comunicaciones han revolucionado nuestra manera de vivir, permitiendo la invención de nuevos bienes y servicios.(1)

Los numerosos avances de la última década en la rama de la informática y las telecomunicaciones han traído consigo la creación y desarrollo de diversos sistemas informáticos, entre los que se encuentran los Sistemas de Información Geográfica (SIG). Los Sistemas de Información Geográfica se han constituido durante los últimos años en unas importantes herramientas de trabajo para la integración de datos espacio-temporales, facilitando la organización, análisis y modelamiento de resultados obtenidos a través de diversas investigaciones. Muchas de dichas investigaciones combinan los SIG con tecnología GPS.

Un Sistema de Posicionamiento Global (GPS) (del inglés Global Positioning System), es un sistema que cuenta con una red de satélites que orbitan sobre el planeta tierra con una trayectoria sincronizada que permite determinar en toda la Tierra la posición de cualquier objeto dígase una persona, un vehículo, entre otros. con una precisión de hasta centímetros. La integración de estos sistemas con los de información geográfica han facilitado el desarrollo de aplicaciones informáticas capaces de monitorear cualquier tipo de navegación ya sea terrestre, marítima o aéreo. Estos programas son conocidos como Sistemas de Control de Flotas, los cuales son usados para visualizar en un mapa vehículos que cuentan con dispositivos GPS.(2)

La gestión de flotas puede incluir una variedad de objetivos y funciones como el mantenimiento, el seguimiento y control, la detención remota de vehículos, el diagnóstico mecánico, la administración de

INTRODUCCIÓN

conductores, la gestión de combustible, la gestión de la seguridad y, en general, todo lo referido al análisis de los datos e información disponible y a la toma de decisiones vinculados a la flota de vehículos. Cada vez es mayor la tendencia a instalar sistemas de control de flotas ya que proporcionan un enorme beneficio tangible a corto y mediano plazo en cuanto a utilidades y el ahorro de capital.

Las empresas que más hacen uso de este tipo de sistemas son las relacionadas con el transporte de bienes por carreteras, equipos de ventas, ambulancias, bomberos, transportación de pasajeros o cualquier mercado que tenga la necesidad de mantener un control riguroso de su flota. Una necesidad de las empresas es administrar los recursos con los que cuenta, por lo cual se ven obligadas a crear estrategias que ayuden a mejorar la administración de los medios que tienen a su disposición, trayendo consigo una mejora sustancial en la prestación de servicios tanto a la población como al sector empresarial.

La administración de los medios de transporte dentro de las empresas cubanas resulta ser una de las áreas más perjudicadas debido al gran número de indisciplinas que son cometidas por algunos trabajadores en este sector. Habitualmente ocurre que numerosas entidades tienen pérdidas económicas debido a las ineficiencias en los servicios que se prestan a través de medios de transporte. Entre los principales fallos se encuentran las entregas fuera de tiempo, las rutas ineficaces, los desvíos en los recorridos con intereses no correspondientes con la empresa, aumentando de esta forma el kilometraje, el descontento de los clientes, el gasto de combustible y una disminución de la vida útil de los equipos.

En la Universidad de las Ciencias Informáticas (UCI) se encuentra la Empresa de Tecnología de la Información para la Defensa (XETID) la cual consta con el centro Tecnologías Espacio Temporales (TET) cuyo trabajo está dirigido al desarrollo de aplicativos a partir del Sistema de Información Geográfica (GENESIG). El principal producto desarrollado por el centro TET es el Sistema de Gestión Inteligente del Transporte (GIT) que cuenta con un grupo de funcionalidades para la gestión de las empresas, permitiendo la representación de información geoespacial en el tiempo, así como la visualización en tiempo real o diferido de los objetivos terrestres, navales y aéreos para así poder garantizar el seguimiento y control de estos elementos, centralizando la información, apoyando la toma de decisiones y posibilitar la definición de estrategias y acciones a seguir.

INTRODUCCIÓN

A su vez se tiene la información referente a los viajes planificados por vehículos con los datos de la actividad a realizar, la fecha de inicio, el vehículo que la realiza y el itinerario que debe seguir. La mayoría de los usuarios[1] que emplean el sistema son empresa que prestan un servicio de transportación y presentan la necesidad de informar a sus clientes el estado del viaje del vehículo que transporta su carga y de forma inmediata conozca las actividades que realiza en tiempo real, sin necesidad que el cliente sea usuario del sistema.

Por ejemplo: En caso de ocurrencia de algún incidente de cualquier índole en la vía con un vehículo asignado, se perderían valiosos minutos u horas entre la ocurrencia del suceso y el conocimiento del cliente de dicha situación. Luego de descrita la situación por la que atraviesan dichas empresas y sus clientes, surge el siguiente **problema a resolver:** ¿Cómo mantener informado a los clientes del estado y ubicación geoespacial de su carga?

Definiéndose como **objeto de estudio:** Sistemas de información geográfica en el trasporte y enmarcándose en el **campo de acción:** Sistema de seguimiento de carga asociado a los sistemas de gestión inteligentes del trasporte.

Planteando como **Idea a defender:** El desarrollo del módulo de seguimiento y monitoreo de la carga, para poder brindar información certera, con mayor calidad y en tiempo real a los clientes de empresas de transporte.

Teniendo como **objetivo general de la investigación:** Desarrollar un módulo que permita el seguimiento y la supervisión de vehículos activos para el sistema de gestión inteligente del trasporte.

Para dar cumplimento al objetivo general se plantea los siguientes **objetivos específicos**.

- Análisis del estado del arte de sistemas que realicen seguimiento de paquetería (Conjunto de productos o mercancías que se guardan o se venden en paquetes.(3)) en el mundo y en Cuba.

[1] Usuario: Son todas aquellas personas o entidades que tienen acceso al sistema GIT.

INTRODUCCIÓN

- Fundamentar el uso de las tecnologías a utilizar en el proceso de desarrollo del componente.
- Identificar, diseñar y describir los requerimientos funcionales y no funcionales, así como los prototipos de usuarios del componente a desarrollar.
- Desarrollar un componente que cumpla con los requerimientos y normativas establecidas para el producto.
- Realizar las pruebas que permitan la validación de la solución desarrollada.

Para la realización de las tareas de la investigación se utilizaron los siguientes **métodos de investigación:**

Métodos Teóricos:

Analítico sintético: Se aplica para analizar la documentación referente a servidores de mapa, las diferentes tecnologías y herramientas con las que se va a trabajar, para luego definir una síntesis específica de estas y lograr una mejor comprensión del objeto de estudio. Este método se evidencia en el capítulo uno en la definición de Tecnologías a Utilizar, Metodología de Desarrollo y Herramientas.

Histórico lógico: Este método ha sido usado con el fin de estudiar la evolución de los sistemas de control de flotas, tanto en Cuba como a nivel internacional. Este método se aplica en el capítulo uno en el análisis de las soluciones existentes.

El presente trabajo de diploma está compuesto por cuatro capítulos, de los cuales se muestra un resumen a continuación:

Capítulo I “Fundamentación Teórica”: Se exponen los aspectos teóricos relacionados con el desarrollo de este trabajo. Se hace un estudio de las tecnologías que se usan a nivel mundial y nacional en el ámbito de la gestión de flotas. Además, se les da fundamento a las metodologías, lenguajes, conceptos y herramientas utilizadas.

Capítulo II “Análisis y Diseño del Sistema”: Se da una propuesta del sistema, se describe el modelo de dominio, se especifican los requisitos que debe cumplir el sistema se define el diagrama de funcionalidades

del sistema. Se elaboran los diagramas del diseño que propone la metodología empleada y Muestra la descripción de la arquitectura y patrones de diseño empleados.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

1.1 Introducción

En este capítulo se hace un estudio de los principales sistemas de control de flotas en Cuba y el resto del mundo que presentan las características necesitadas por el sistema a desarrollar. Además, se definen las herramientas, lenguajes y metodologías que serán utilizadas para el desarrollo de la aplicación.

1.2 Estudio del estado del arte

Para conocer las características de las principales aplicaciones informáticas de gestión y control de flotas a nivel mundial se tomaron en cuenta los resultados de las búsquedas realizadas en Internet con diferentes buscadores como por ejemplo Google, Google Académico, Scielo, scholarpedia, entre otros. A raíz de los resultados obtenidos se decide tomar como posibles soluciones las siguientes empresas.

Amazon.com, Inc.: es una industria estadounidense de comercio electrónico y servicios de computación en la nube a todos los niveles con sede en la ciudad estadounidense de Seattle, Estado de Washington. Actualmente la empresa está totalmente diversificada en diferentes líneas de productos, ofreciendo DVD, CD de música, software, videojuegos, electrónica, ropa, muebles, comida, libros, entre otros. y es la marca de venta al por menor más valiosa del mundo según el índice Brand (4).

Amazon actualmente cuenta con un servicio de geolocalización que permite a los compradores rastrear la ubicación exacta de sus conductores de entrega en un mapa. Dicha funcionalidad llamada Amazon Map Tracking, proporciona actualizaciones en vivo de las rutas de entrega de los conductores, incluida la cantidad de paradas que quedan antes de que llegue su paquete. Amazon envía una notificación a los clientes cuando pueden ver sus controladores de entrega en el mapa de seguimiento. Los clientes también pueden acceder al mapa visitando la función "paquete de seguimiento" en el sitio web o aplicación de Amazon.(5)

CAPÍTULO 1

Grupo Empresarial Correos de Cuba: Correos de Cuba es la entidad estatal de servicios de mayor capilaridad del país. Es el Operador Designado por el Estado cubano para garantizar el Servicio Postal Universal (SPU) en todo el territorio nacional, a todos los ciudadanos, a precios asequibles y con adecuado nivel de calidad, así como otros servicios de valor añadido a cuenta de terceros, con la finalidad de lograr la máxima satisfacción de la sociedad cubana y su integración con el mundo.(6)

Correos de Cuba cuenta con una amplia red de empresas y oficinas de correos, sucursales, ventanillas, puntos de venta (estanquillos) y otras entidades a lo largo y ancho de toda la isla, incluyendo en los lugares más apartados, que brindan el Servicio Postal Universal y otros servicios a cuenta de terceros, por encargo estatal, para todos los ciudadanos, a precios asequibles y con aceptable nivel de calidad, conforme a lo establecido en las Actas y el Convenio Internacional de la Unión Postal Universal (UPU), de los que Cuba es signataria.

El Centro Principal Tecnológico Postal de la Empresa Correos de Cuba posee un servicio digital para rastrear los envíos que se realizan a través del Servicio Postal Universal, desde que se imponen en el país de origen hasta que llegan a la unidad de Correos de Cuba donde el destinatario debe recogerlo.(7)

Este servicio, al que se puede acceder directamente desde la página de Correos de Cuba o desde su apk, amplía el que hoy se ofrece, que solo posibilita el rastreo desde que llega a nuestro país hasta que se entrega al destinatario, o partir de que se impone en las oficinas de Correos hacia el exterior.

Luego de analizar la solución que ofrecen Amazon y correos de Cuba, se llega a la conclusión de que estas empresas disponen de un software capaz de solucionar el problema, pero ademas de requerir de una conexión a Internet rápida y accesible en todo momento, ya que la aplicación se encuentra publicada en los servidores de dichas empresa, requiere que el agente que utilice dicha funcionalidad sea usuario del sistema para el caso de Amazon, en cambio, en la variante de Correos de Cuba, existen dependencias que con enviar los datos (código) de la mercancía se puede rastrear el envío, por ejemplo: https://postal.ninja/es/p/correos-cuba/tracking.(8)

CAPÍTULO 1

1.3 Tecnologías utilizadas

Las tecnologías de la informática y las comunicaciones han tenido un gran avance en estos últimos años lo que ha propiciado la creación de muchas herramientas que facilitan la creación de sitios web. Para el desarrollo de la aplicación se hace necesario que las tecnologías que se usen ayuden a mejorar los resultados de la misma en cuanto a estabilidad y seguridad.

1.3.1 GPS (Global Positioning System)

El GPS es un sistema de radionavegación, basado en el espacio, que proporciona servicios fiables de posicionamiento, navegación, y cronometría gratuita e ininterrumpidamente a usuarios en todo el mundo. A todo el que cuente con un receptor del GPS, el sistema le proporcionará su localización y la hora exacta en cualesquiera que sean las condiciones atmosféricas, de día o de noche, en cualquier lugar del mundo y sin límite al número de usuarios simultáneos.

El GPS se compone de tres elementos: los satélites en órbita alrededor de la Tierra, las estaciones terrestres de seguimiento y control, y los receptores GPS propiedad de los usuarios. Desde el espacio, los satélites GPS transmiten señales que reciben e identifican los receptores del GPS; ellos, a su vez, proporcionan por separado sus coordenadas tridimensionales de latitud, longitud y altitud, así como la hora local.(2)

1.3.2 Lenguaje de programación

Un lenguaje de programación no es más que un conjunto de símbolos, que combinados con reglas sintácticas y semánticas definen la estructura y el significado de sus elementos y expresiones. Los lenguajes de programación son utilizados para controlar el comportamiento físico y lógico de una máquina.(9)

La selección de los lenguajes de programación depende en gran medida de la arquitectura y el tipo de aplicación a desarrollar. En la presente investigación se pretende desarrollar un sistema de tipo Web, por lo que los lenguajes de programación serán enfocados en este tipo de tecnología.

CAPÍTULO 1

Puesto que el módulo a desarrollar utiliza como base el sistema de Gestión Inteligente del Transporte desarrollado por la empresa XETID, serán utilizados los lenguajes de programación definidos en dicho sistema.

JavaScript

JavaScript es un lenguaje de programación interpretado, dialecto del estándar ECMAScript. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico. Este lenguaje permite crear las diferentes vistas de la aplicación que interactuarán con el usuario en las diferentes situaciones.

Se utiliza principalmente en su forma del lado del cliente (client-side), implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario, páginas web dinámicas y en bases de datos locales al navegador, aunque existen implementaciones de dicho lenguaje del lado del servidor conocidos como SSJS (Server-side JavaScript). Para el desarrollo de la aplicación se eligió la biblioteca ExtJS que está desarrollada en JavaScript, usando tecnologías como AJAX (Asynchronous JavaScript And XML) (JavaScript asíncrono y XML) que no es más que una técnica de desarrollo web para crear aplicaciones interactivas(10).

PHP

PHP (Pre-procesador de Hipertexto) es un lenguaje interpretado de propósito general ampliamente usado y diseñado especialmente para desarrollo Web. Se ejecuta en un servidor, que toma el código PHP como su entrada y generando HTML como salida. Puede ser desplegado en todos los sistemas operativos que contengan un intérprete PHP compatible. Posee soporte para la mayoría de los gestores de bases de datos, comúnmente usados. Todas estas ventajas hacen de PHP el lenguaje a seleccionar del lado del servidor para el desarrollo de la solución, adicionando que a partir de la versión 5 se hicieron mejoras sustanciales al rendimiento y un mejor soporte para XML.(11)

Como el sistema GENESIG está programado en PHP 5 está enfocado al desarrollo de aplicaciones web en el mismo lenguaje de programación. Se hace obligatorio disponer de unos conocimientos

avanzados de PHP 5 para sacar el máximo partido al framework. La versión mínima de PHP requerida para ejecutar GENESIG es PHP 5.2.

1.3.2.3 Framework

Un framework es una estructura de archivos y utilidades que aceleran la programación de una aplicación informática, que provee una metodología de trabajo que sistematiza y facilita la generación de formularios, funciones y módulos de uso común, permitiendo al desarrollador dedicar su atención hacia los aspectos específicos de cada aplicación.(12)

OpenLayers

La opción seleccionada para este aspecto fue OpenLayers que no es más que una herramienta de código libre que es ejecutada del lado del cliente. Está diseñada para trabajar específicamente con mapas y es compatible con la mayoría de los navegadores web. Este framework funciona íntegramente del lado del cliente, no es necesario ninguna configuración específica o software del lado del servidor para ser ejecutado. OpenLayers ha alcanzado un alto nivel de madurez debido a la gran cantidad de desarrolladores que colaboran entre sí, formando parte activa de la comunidad de dicho framework.(13)

1.3.2.4 Lenguaje de Modelado UML 2.0 (Unified Modeling Language)

El UML es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir y tiene como objetivo entender, diseñar, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios. El lenguaje de modelado pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.(14)

CAPÍTULO 1

Lo fundamental de una herramienta UML es la capacidad de diagramación, los diferentes tipos de diagramas que soporta y sus esquemas de apoyo de diseño, documentación, construcción e implantación de sistema. Así mismo, su flexibilidad para admitir cambios no previstos durante el diseño o el rediseño.

El modelo UML consta de dos categorías principales de los elementos que lo integran, cada uno de los cuales pueden ser utilizados para hacer declaraciones sobre los diferentes tipos de elementos individuales dentro del sistema que está siendo modelado. Estas categorías son:

- Clasificadores. Un clasificador describe un conjunto de objetos. Un objeto es un individuo con un estado y las relaciones de otros objetos. El estado de un objeto identifica los valores para ese objeto de propiedades del clasificador.

Comportamientos. Un comportamiento describe un conjunto de posibles ejecuciones. Una ejecución es una representación de un conjunto de acciones que pueden generar y responder a las ocurrencias de eventos. (14)

1.4 Herramientas

1.4.1 Entorno de desarrollo integrado (IDE)

Un IDE (Integrated Development Environment), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen autocompletado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos.(15)

PhpStorm

PhpStorm es un IDE comercial (multiplataforma) (entorno de desarrollo integrado) para PHP, creado por la empresa checa JetBrains, con sede en la República Checa. PhpStorm proporciona un editor para PHP, HTML y JavaScript con análisis de código sobre la marcha, prevención de errores y

refactorizaciones automatizadas para código PHP y JavaScript. La finalización del código de PhpStorm admite PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1, 7.2, 7.3 y 7.4 (proyectos modernos y heredados), incluidos generadores, cortinas, la palabra clave finalmente, lista en foreach, espacios de nombres, cierres, rasgos y sintaxis de matriz corta. Incluye un editor SQL completo con resultados de consulta editables.

La versión gratuita es una herramienta con enorme potencial. Desarrollado por JetBrains, PHPStorm funciona muy bien con los principales frameworks (Symfony, Zend Framework, Yii, CakePHP, Laravel) y con los principales sistemas CMS como Drupal, Magento y Wordpress . Pero no solo destaca en la parte back. Este IDE ofrece edición en directo con tecnologías como CSS, HTML5, JavaScript o TypeScript. La popularidad de PHPStorm se puede medir por el hecho de que grandes marcas como Expedia, Yahoo, Cisco, Salesforce y wikipedia también utilizan PHPStorm como su editor de código.(16)

1.4.2 Servidor Web

Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor realizando conexiones bidireccionales y/o unidireccionales, así como síncronas o asíncronas con el cliente generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se utiliza el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al ordenador que ejecuta el programa.

Apache 2

Apache 2 es un servidor web multiplataforma de código abierto. Es modular (basado en módulos), donde cada módulo ofrece un grupo de funcionalidades específicas al servidor. Es uno de los servidores web más utilizado en Internet, lo que facilita el acceso a la documentación. Provee un alto nivel de seguridad y eficiencia, permitiendo además el uso de una versión local, la cual hace posible

que el servidor actúe como servidor y cliente al mismo tiempo, creando así la posibilidad de pre visualizar y probar el código mientras este es desarrollado. Apache presenta entre otras características altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración(17). El autor decidió emplear Apache por las características que presenta, su uso extendido a nivel mundial y fundamentalmente porque es el utilizado en el sistema GENESIG.

1.4.3 Servidor de bases de datos

Los servidores de bases de datos surgen con motivo de la necesidad de las empresas de manejar grandes y complejos volúmenes de datos, al tiempo que requieren compartir la información con un conjunto de clientes de una manera segura. Ante este enfoque, un SGBD (Sistema Gestor de Bases de Datos) deberá ofrecer soluciones de forma fiable, rentable y de alto rendimiento el cual no es más que un sistema bajo arquitectura cliente/servidor que proporciona servicios de gestión, administración y protección de la información a través de conexiones de red, gobernadas por unos protocolos definidos y a los que acceden los usuarios, de modo concurrente, a través de aplicaciones clientes.

PostgreSQL

Es un sistema de gestión de base de datos relacional orientada a objetos, libre y publicado bajo la licencia BSD. Además es el gestor de bases de datos de código abierto más avanzado en la actualidad, ofreciendo control de concurrencia multiversión, soportando casi todas las sintaxis SQL como subconsultas, transacciones y funciones definidas por el usuario, cuenta además con un amplio conjunto de enlaces con lenguajes de programación como C, C++, Java y Python (18). El autor rechazó el empleo de PostgreSQL debido a que presenta una complejidad de uso un poco más alta que MySQL y además su principal uso no está enfocado en los sistemas web.

Límite	Valor
Máximo tamaño base de dato	Ilimitado (Depende del sistema de almacenamiento)
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB
Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 (dependiendo del tipo)
Máximo número de índices por tabla	Ilimitado

Tabla 1.1: Principales características de PostgreSQL Fuente: Elaboración propia

1.4.4 Herramienta de modelado CASE

Las herramientas CASE son un conjunto de aplicaciones informáticas, usadas para automatizar actividades del ciclo de vida de desarrollo de sistemas (SDLC). Las herramientas CASE son usadas por los directores de proyectos de software, analistas e ingenieros para desarrollar sistemas de software (19).

Según Pressman (20) son un conjunto de herramientas y métodos asociados que proporcionan asistencia automatizada en el proceso de desarrollo del software a lo largo de su ciclo de vida,

mediante su uso se reduce el tiempo, costo del desarrollo y mantenimiento del software, así como se mejora su calidad antes de proseguir con la siguiente etapa del desarrollo de software

1.4.4.1 Visual Paradigm 8.0

Visual Paradigm es una herramienta CASE que utiliza UML como lenguaje de modelado, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación.

Visual Paradigm ofrece:

- Entorno de creación de diagramas para UML 2
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa (versión profesional) e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IDEs (Entorno de Desarrollo Integrado).
- Disponibilidad en múltiples plataformas.

Visual Paradigm 8 se ha escogido como herramienta para el modelado de la solución propuesta teniendo en cuenta que es una herramienta multiplataforma, permite modelado orientado a objeto, y a la vez, orientado UML, el cual fue escogido para la realización del presente trabajo. Además, tiene algunas versiones gratuitas. Soporta aplicaciones Web, exportación/importación XML y exporta en formato HTML. Cuenta además con gran cantidad de tutoriales, es fácil de instalar y de actualizar, tiene características gráficas muy cómodas y se encuentra disponible en varios idiomas.(21)

1.5 Metodología de desarrollo de software

CAPÍTULO 1

Se trata del proceso cuya finalidad es desarrollar productos o soluciones para un cliente o mercado en particular, teniendo en cuenta factores como los costes, la planificación, la calidad y las dificultades asociadas**.** A todo esto, es a lo que denominamos metodologías de desarrollo de software. Es decir, se trata del proceso que se suele seguir a la hora de diseñar una solución o un programa específico. Tiene que ver, por tanto, con la comunicación, la manipulación de modelos y el intercambio de información y datos entre las partes involucradas. O para ser más precisos, las metodologías de desarrollo de software son enfoques de carácter estructurado y estratégico que permiten el desarrollo de programas con base a modelos de sistemas, reglas, sugerencias de diseño y guías (13).

En el proceso de desarrollo de software la metodología es considerada como el elemento rector a lo largo del mismo. Por tal motivo se propone para el desarrollo del trabajo de diploma, el uso de la metodología AUP en su versión para la UCI debido a que aplica técnicas ágiles y la misma ha sido definida por la universidad para su empleado en el ciclo de vida de sus proyectos productivos.

1.5.1 Metodología AUP variación para la UCI

Como metodología utilizada para guiar el proceso de desarrollo de software en la presente investigación se seleccionó el Proceso Unificado Ágil (AUP por sus siglas en inglés) variación UCI. La metodología AUP-UCI cuenta con varios escenarios (cuatro exactamente) que definen las pautas del diseño de la solución. El autor determinó luego de un estudio realizado a cada escenario que se utilice el escenario dos de AUP-UCI el cual emplea los casos de uso del sistema. El autor justifica la elección debido a que la metodología variación de AUP para la UCI está formada por tres fases, (Inicio, Ejecución y Cierre) para el ciclo de vida de los proyectos de la universidad, las cuales contienen las características de las cuatro fases (Inicio, Elaboración, Construcción y Transición) propuestas en AUP. Las características de las fases de la metodología de la universidad son:

- Inicio: En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si se ejecuta o no el proyecto.

- Ejecución: En esta fase se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura. Durante el desarrollo se modela el negocio, se obtienen los requisitos, se elaboran la arquitectura y el diseño, se implementa y se libera el producto.

- Cierre: En el cierre se analizan tanto los resultados del proyecto como su ejecución y se realizan las actividades formales de cierre del proyecto.

Disciplinas de variación AUP-UCI

- Modelado de negocio: El modelado del negocio es la disciplina destinada a comprender los procesos de negocio de una organización. Se comprende cómo funciona el negocio que se desea informatizar para tener garantías de que el software desarrollado va a cumplir su propósito. Para modelar el negocio se proponen las siguientes variantes:

  1. Casos de Uso del Negocio (CUN)

  2. Descripción de Proceso de Negocio (DPN)

  3. Modelo Conceptual (MC)

- Requisitos: El esfuerzo principal en la disciplina Requisitos es desarrollar un modelo del sistema que se va a construir. Esta disciplina comprende la administración y gestión de los requisitos funcionales y no funcionales del producto. Existen tres formas de encapsular los requisitos [Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP)], agrupados en cuatro escenarios condicionados por el modelado de negocio.

- Análisis y diseño: En esta disciplina, si se considera necesario, los requisitos pueden ser refinados y estructurados para conseguir una comprensión más precisa de estos, y una descripción que sea fácil de mantener y ayude a la estructuración del sistema (incluyendo su arquitectura).

- Implementación: En la implementación, a partir de los resultados del análisis y diseño se construye el sistema.

- Pruebas internas: En esta disciplina se verifica el resultado de la implementación probando cada construcción, incluyendo tanto las construcciones internas como intermedias, así como las versiones finales a ser liberadas. Se deben desarrollar artefactos de prueba como: diseños de casos de prueba, listas de chequeo y de ser posible, componentes de prueba ejecutables para automatizar las pruebas.

- Pruebas de liberación: Pruebas diseñadas y ejecutadas por una entidad certificadora de la calidad externa, a todos los entregables de los proyectos antes de ser entregados al cliente para su aceptación.

Pruebas de aceptación: Es la prueba final antes del despliegue del sistema. Su objetivo es verificar que el software está listo y que puede ser usado por usuarios finales para ejecutar aquellas funciones y tareas para las cuales el software fue construido

Herramienta CASE	Lenguaje principal	Lenguajes auxiliares	Frameworks	Servidor web	Servidor BD	IDE
Visual Paradigm 8	PHP7.3.2	HTML5, CSS3, JavaScript ExtJS	OpenLayers 2	Apache 2	PostgreSQL	PhpStorm 2019 2.3

Tabla 1.2: Herramientas para el desarrollo de la aplicación. Fuente: elaboración propia

1.5 Conclusiones parciales

CAPÍTULO 1

En este capítulo se esclarecieron los conceptos asociados con el dominio del problema y se definieron las principales tecnologías que contribuirán con el desarrollo de la solución propuesta, aportando así una base para que los investigadores tengan una mejor visión del ámbito donde se desarrolla la investigación. A partir de estos conceptos se encaminó el estudio del estado del arte de sistemas homólogos, evidenciándose que no resuelven el problema planteado. Aun así, estos sistemas aportan elementos importantes para el desarrollo del módulo que se propone realizar. Se realizó además un análisis profundo del objeto de estudio, el cual permitió tener una noción más nítida del problema a resolver. Finalmente, la selección de las herramientas y tecnologías seleccionadas permitió definir el entorno técnico donde se desarrollará la solución propuesta.

CAPÍTULO 2

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DEL SISTEMA

2.1 Introducción

En este capítulo se da una propuesta del sistema, se describe el modelo de dominio, se especifican los requisitos que debe cumplir el sistema y se define el diagrama de funcionalidades del sistema. Contiene el diagrama de despliegue, a través del cual se puede ver la distribución del sistema de forma física. Contiene además la realización y descripción de los casos de prueba. Además, se establecen los requisitos funcionales y no funcionales con los que debe cumplir el sistema, así como el diagrama de funcionalidades del sistema.

2.2 Modelo Conceptual

Un modelo conceptual es una representación de un sistema, hecho de la composición de conceptos que se utilizan para ayudar a las personas a conocer, comprender o simular un tema que representa el modelo. También es un conjunto de conceptos. Algunos modelos son objetos físicos; por ejemplo, un modelo de juguete que se puede ensamblar y se puede hacer que funcione como el objeto que representa. El término modelo conceptual puede usarse para referirse a modelos que se forman después de un proceso de conceptualización o generalización. Los modelos conceptuales son a menudo abstracciones de cosas en el mundo real, ya sean físicas o sociales. Los estudios semánticos son relevantes para varias etapas de la formación de conceptos. La semántica se trata básicamente de conceptos, el significado que los seres pensantes dan a varios elementos de su experiencia (22).

CAPÍTULO 2

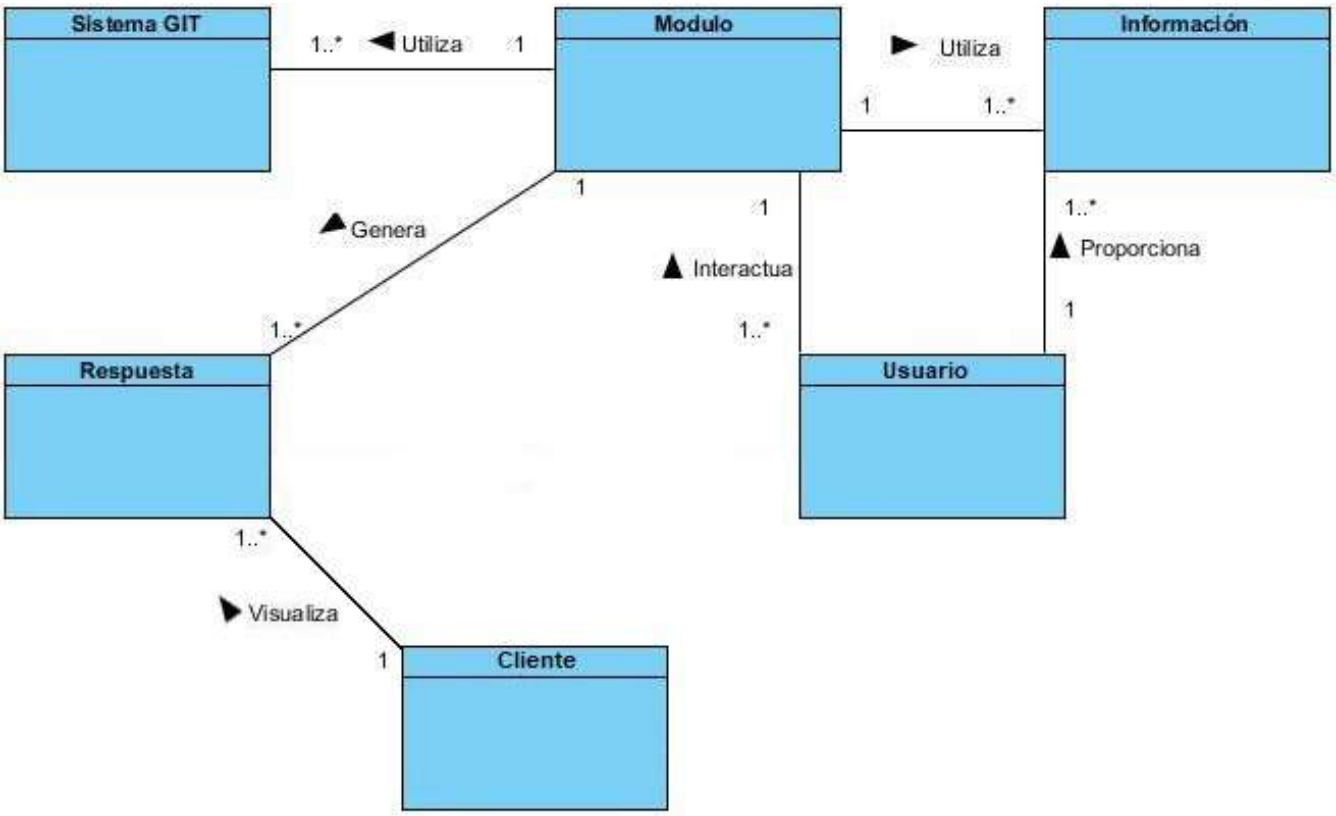


Ilustración 2.1: Modelo Conceptual. Fuente: Elaboración propia

Definición de los conceptos del modelo

1. **Sistema GIT**: Sistema informático que reúne y gestiona toda la información que precisa el centro TET.
2. **Módulo**: Complemento del sistema GIT que proporciona la vista deseada a los clientes a partir de la información insertada
3. **Usuario**: Como se definió en el capítulo anterior, son todas aquellas personas o entidades que tienen acceso al sistema GIT.
4. **Respuesta**: Vista de mapa del sistema que interactúa con el cliente.
5. **Cliente**: Son todas aquellas personas o entidades que precisan de los servicios del sistema sin ser usuarios del mismo.

6. **Información**: Conjunto de datos Introducidos por los usuarios al módulo para general una respuesta del lado del cliente.

2.3 Propuesta de solución

Para dar solución a la problemática planteada en el capítulo anterior se decide desarrollar un módulo auxiliar para el sistema GENESIG. Este módulo contempla la generación de vistas de mapas con la ubicación de vehículos a partir de datos introducidos por los usuarios del sistema con el objetivo de satisfacer las necesidades de sus clientes. Dicho módulo generará también un localizador de recursos uniforme (URL) que al ser accedido por el cliente podrá visualizar el mapa generado con los objetivos en movimientos. Se llevará un control de los objetivos activos en una tabla, así como los datos de los clientes, de los objetivos y de la URL. Para acceder al módulo solo bastará ser un usuario previamente autenticado en el sistema GENESIG. Las vistas del módulo constan de una tabla donde se introducirán y controlaran los datos del activo deseado una vista de mapa del lado del cliente a la cual podrá acceder a través de una URL enviada por un usuario del sistema y donde podrá apreciar el activo en cuestión en movimiento con los datos del recorrido. Los usuarios podrán consultar el módulo y su información en cualquier momento y los clientes solo podrán acceder a la URL mientras esté activa. El módulo en cuestión se implementará siguiendo la arquitectura Modelo-Vista-Controlador (MVC) la cual será detallada en los acápites posteriores.

Para tener una mejor visión de la estructura de la solución se propone el siguiente diagrama que representa como se distribuye la información del sistema a desarrollar. La arquitectura de información no es más que la disciplina y arte encargada del estudio, análisis, organización, disposición y estructuración de la información en espacios de información, y de la selección y presentación de los datos en los sistemas de información interactiva y no interactiva (23).

CAPÍTULO 2

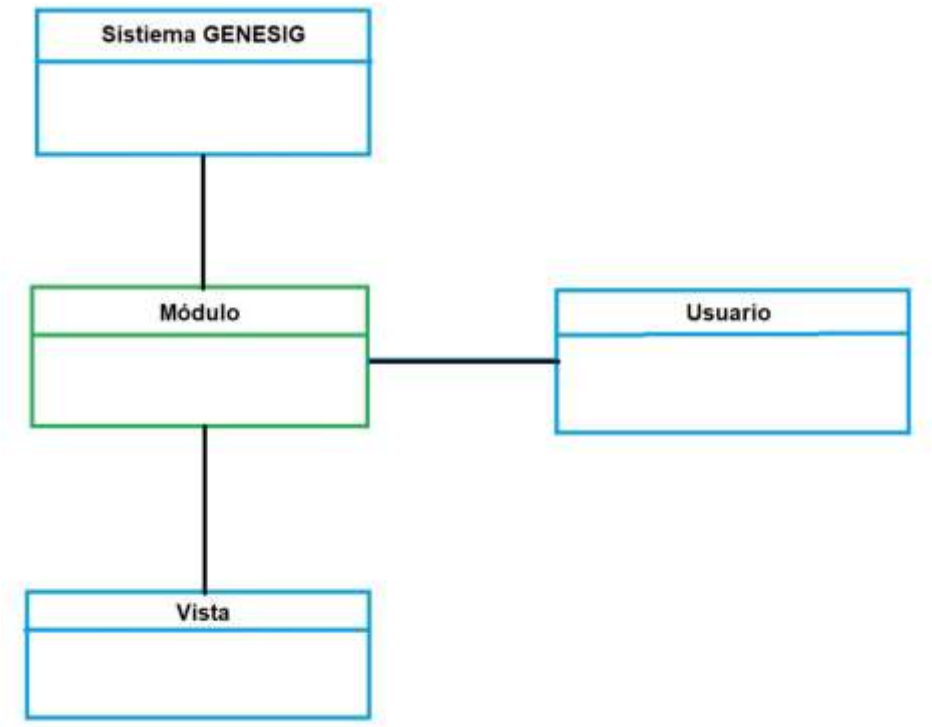


Ilustración 2.1 Arquitectura de la información de la propuesta de solución. Fuente: elaboración propia

2.3 Requisitos

Según Pressman y Ramdhani (24) los requisitos de software son las necesidades de los clientes, los servicios que los usuarios desean que proporcione el sistema de desarrollo y las restricciones en las que debe operar. Los requisitos se dividen en Funcionales y No Funcionales y muestran las capacidades o condiciones que el sistema debe cumplir y las propiedades o cualidades que el producto debe tener. En este acápite se listarán los requisitos funcionales y los no funcionales que serán aplicados en la solución a implementar.

2.3.1 Fuente de obtención de los requisitos

Uno de los elementos más importantes del proceso de desarrollo del software es la obtención de los requisitos, debido a que ayuda a conciliar conflictos de intereses entre los involucrados, y determinar qué

tipo de software se desea desarrollar(25). En este proceso intervienen diferentes fuentes que permiten identificar los requisitos que forman parte de una aplicación informática. Durante esta etapa de la investigación se tuvieron en cuenta como fuentes de obtención de requisitos:

- **Especialistas de Gobierno Electrónico XETID**
- **Análisis de sistemas similares de control de flotas**

2.3.2 Técnicas de obtención de requisitos

Una correcta comprensión de los requisitos favorece el desarrollo de los sistemas informáticos y que estos cumplan con las necesidades y expectativas del cliente. Para realizar este procedimiento existen diversas técnicas que guían al equipo de proyecto en el proceso de comunicación con el cliente. (25)

En la obtención de los requisitos de Módulo de seguimiento de objetivos en tiempo real para el sistema de gestión inteligente del trasporte se empleó como técnicas de identificación de requisitos la entrevista ideas. Su definición y forma de aplicación se enuncian a continuación.

Entrevista

La entrevista es de gran utilidad para obtener información cualitativa como opiniones, o descripciones subjetivas de actividades. Es una técnica muy utilizada, y requiere una mayor preparación y experiencia por parte del analista. La entrevista se puede definir como un “intento sistemático de recoger información de otra persona” a través de una comunicación interpersonal que se lleva a cabo por medio de una conversación estructurada. Debe quedar claro que no basta con hacer preguntas para obtener toda la información necesaria. Es muy importante la forma en que se plantea la conversación y la relación que se establece en la entrevista. La técnica se aplicó a través de una entrevista realizada a miembros del centro de Gobierno electrónico de XETID para conocer cuáles son las tecnologías compatibles con el entorno de desarrollo, los sistemas con que se integra, así como los requerimientos y las características que debe tener una aplicación para poder pertenecer al sistema de gestión inteligente del trasporte.

2.3.3 Especificación de Requisitos de Software

La Especificación de Requisitos de Software (ERS) es una descripción completa del comportamiento y características del sistema que se va a desarrollar. Sus principales objetivos son ayudar a los clientes a describir claramente lo que se desea obtener de un determinado software y ayudar a los desarrolladores a entender qué quiere exactamente el cliente. La ERS incluye un conjunto de requisitos funcionales y no funcionales que sirven para describir las necesidades de los clientes y de los usuarios, así como los requerimientos que debe tener el sistema para tener un mejor funcionamiento(26).

Requisitos Funcionales

Un requisito funcional define una función del sistema de software o sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Los requisitos funcionales pueden ser: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que se supone, un sistema debe cumplir. Los requerimientos de comportamiento para cada requerimiento funcional se muestran en los casos de uso. Son complementados por los requisitos no funcionales, que se enfocan en cambio en el diseño o la implementación. Como se define en la ingeniería de requisitos, los requisitos funcionales establecen los comportamientos del sistema (27).

A continuación, se muestra en formato de tabla, los requisitos funcionales que serán implementados en la solución informática:

Requerimientos	Prioridad/Complejidad	Descripción
RF1 Crear cliente RF2 Actualizar cliente RF3 Eliminar cliente RF4 Listar clientes	Alta/Baja	Permite la gestión por parte del usuario de la información referente al o a los objetivos solicitados de un cliente tales como: nombre o id del cliente, empresa a la que pertenece, identificador del carro, fecha de inicio del recorrido y fecha final del recorrido. El usuario con

Gestionar Cliente (RF 1-4)		estos datos podrá crear, editar, listar y eliminar clientes.
RF5 Crear URL	Alta/Baja	Permite crear un localizador de recursos uniformes URL a partir de los datos insertados en el módulo para acceder al mapa con el objetivo deseado.
RF6 Desactivar URL	Alta/Baja	Permite inhabilitar una URL una vez cumplida su función.
RF7 Mostrar mapa	Alta/Media	Permite mostrar un mapa del territorio nacional a partir de los servicios brindados por el sistema GENESIG.
RF8 Mostrar ruta	Alta/Media	Permite mostrar la ruta que seguirá el objetivo mediante una línea finita sobre el mapa.
RF9 Mostrar información del objetivo en el mapa	Alta/Media	Permite mostrar la información del objetivo a seguir en el mapa a partir de los datos brindados por el sistema GENESIG tales como: Nombre del chofer, chapa del vehículo y velocidad en Km/h.
RF10 Mostrar tiempo de estimación de llegada	Alta/Media	Permite mostrar el tiempo estimado de llegada del objetivo a su destino a partir

		del cálculo realizado entre la velocidad y la distancia del recorrido.
RF11 Mostrar objetivo	Alta/Media	Permite representar el objetivo a seguir mediante un símbolo definido por el sistema sobre el mapa.

Tabla 2.1 Requisitos funcionales. Fuente: elaboración propia

Requisitos no funcionales

Un requisito no funcional o atributo de calidad es, en la ingeniería de sistemas y la ingeniería de software, un requisito que determina los criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requisitos funcionales(28).

A continuación, se muestra los requisitos no funcionales divididos por categorías que serán implementados en la solución:

Usabilidad

RNF1. El diseño del sistema estará enfocado en los usuarios del mismo. Contará con acceso sencillo e intuitivo para su uso (llenado de tabla y botones).

RNF2. Se tendrá en cuenta los errores que cometan los usuarios en la entrada de datos alertándolos que error realizaron.

Fiabilidad

RNF3. La información estará disponible todo el tiempo para los usuarios del sistema.

Diseño e implementación

RNF4. Lenguaje de programación: PHP 7.3.2, ExtJS, JavaScript, HTML5 (lenguaje etiquetado), CSS 3.

RNF5. Servidor Web: Apache 2.

RNF6. Servidor de Base de Datos: PostgreSQL.

RNF7. El sistema puede ser accedido desde cualquier navegador web que tenga soporte para los estándares de la WC3.

Seguridad

RNF8. Los clientes no tendrán acceso al sistema GENESIG, solo tendrán acceso al mapa y a los objetivos solicitados mediante la utilización una URL.

Requisitos de software

RNF9. Para las PC clientes solo deben contar con un navegador que soporte el estándar WC3.

RNF10. Las PC clientes deben tener conexión a la misma red que utiliza GENESIG.

Requisitos de hardware

RNF11. Para las PC clientes se requerirán máquinas con procesador de velocidad igual o mayor a 2.3 GHz de micro, tarjeta de red a 100 Mb/s o equivalente y 512 Mb de RAM.

RNF12. Para las PC servidor el hardware donde se instalará debe cumplir las especificaciones solicitadas por el sistema GENESIG.

2.4. Descripción o modelo del sistema

En el presente capítulo, luego de haber realizado la descripción de los requisitos, se le da paso a la siguiente etapa de conceptualización de un software, que es la descripción del Modelo de Casos de Uso del Sistema (DCUS), que contiene actores, casos de uso y sus respectivas relaciones. Los requisitos del software se agrupan según sus características, utilidad y usabilidad conformando así los llamados casos de uso, los

cuales no son más que fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores.

Los casos de uso son descripciones funcionales del sistema que permiten definir los límites del software y las relaciones entre la aplicación y el entorno; describen cómo los actores pueden usar un sistema. Especifican una secuencia de acciones que debe devolver algún resultado de valor a un actor(29).

El módulo de seguimiento de objetivos en tiempo real para el sistema de gestión inteligente del trasporte cuenta con un actor el cual se especifican a continuación

Actor	Descripción
Usuario	Son las personas que interactúan con el sistema y cuentan con todos los permisos para gestionar la información requerida por el módulo.

Tabla 2.2 Actores del sistema. Fuente: elaboración propia

2.4.1. Diagrama de casos de uso del sistema

La solución al problema planteado de la presente investigación del módulo de seguimiento de objetivos en tiempo real para el sistema de gestión inteligente del trasporte cuenta con un DCUS, el cual se presenta a continuación. Este diagrama representa gráficamente a los procesos y su interacción con los actores.

CAPÍTULO 2

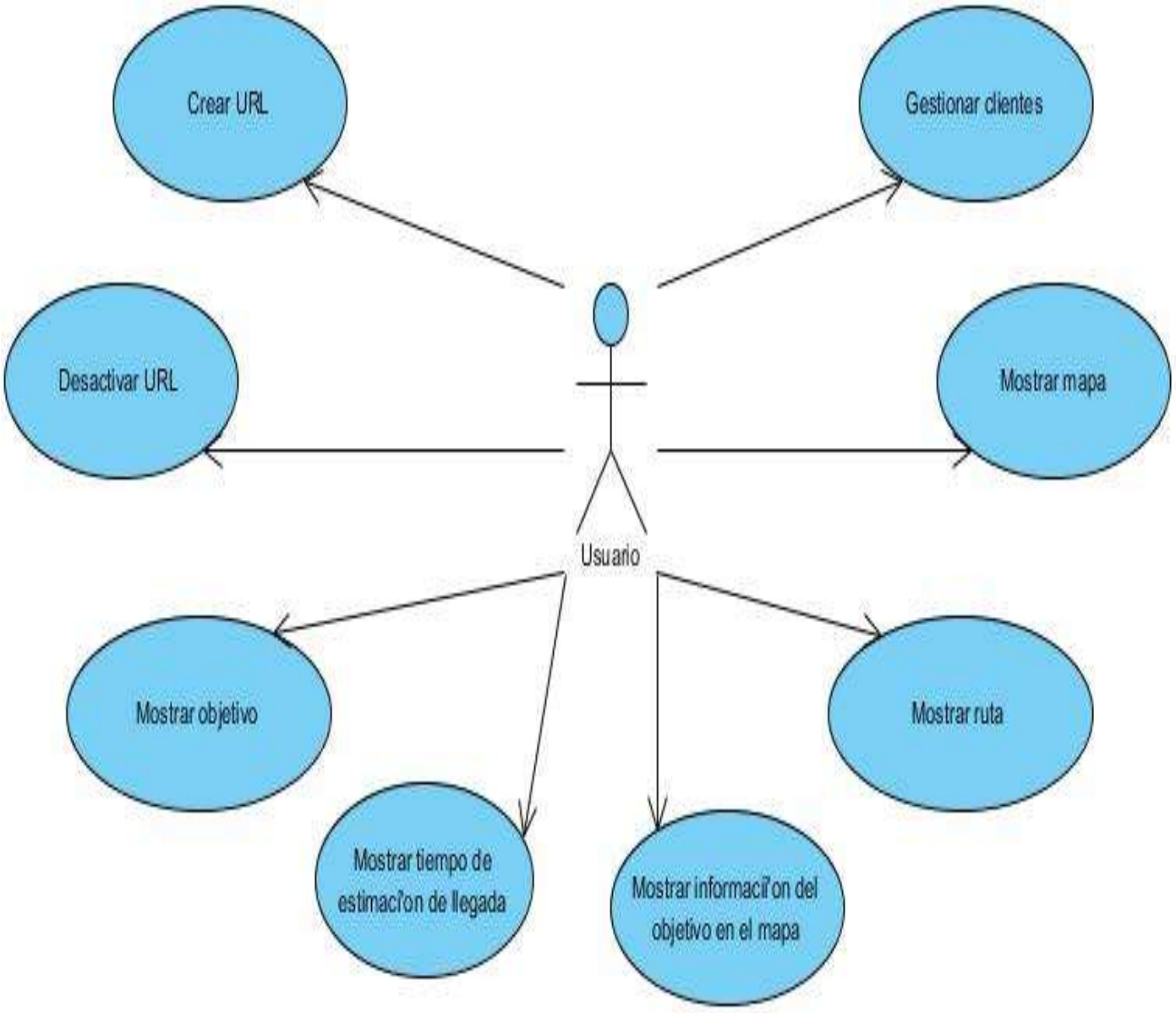


Ilustración 2.2 Diagrama de casos de uso del sistema. Fuente: elaboración propia

2.4.2 Descripción textual de los casos de uso del sistema

En la siguiente tabla se especifica la descripción textual del caso de uso del Sistema “Gestionar Cliente”.

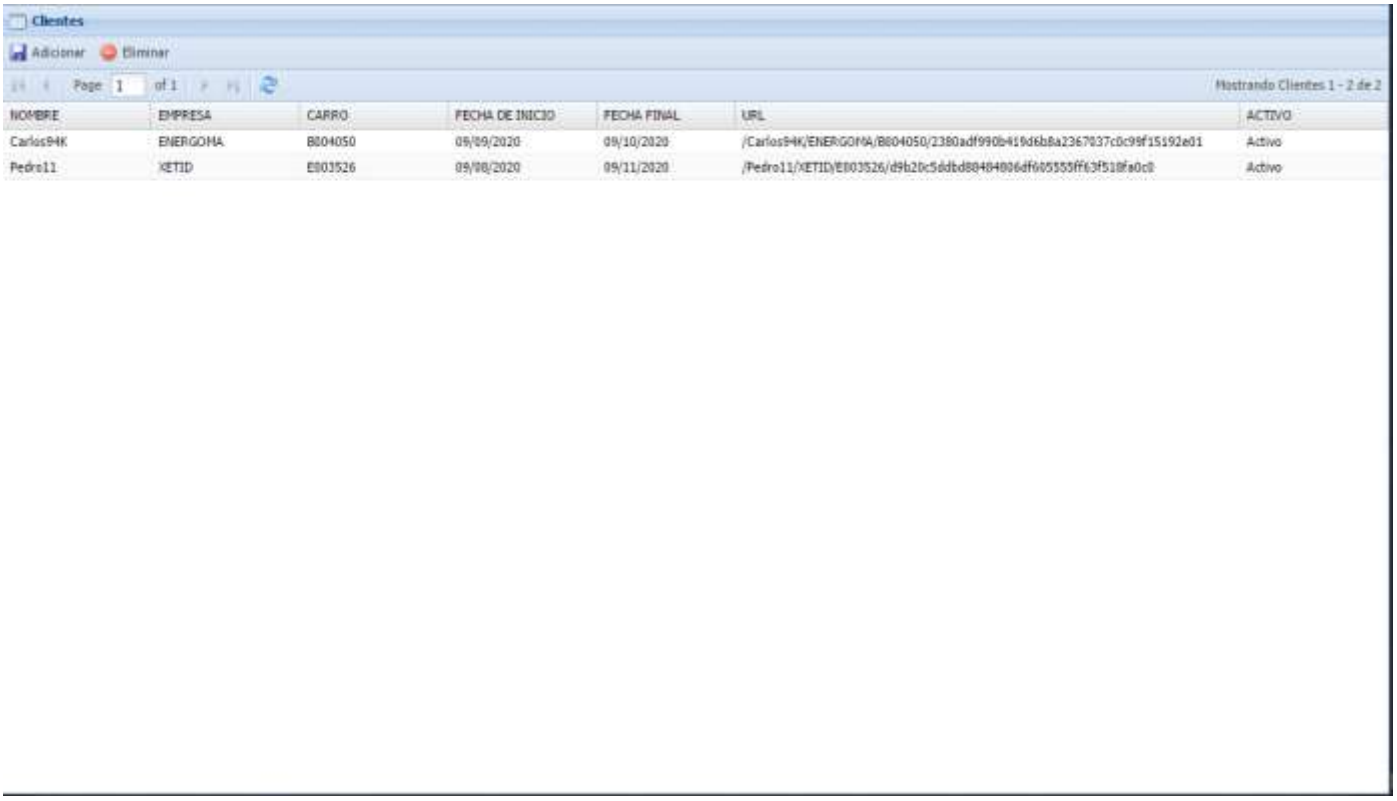
CAPÍTULO 2

Caso de uso	**Gestionar Cliente**
Actores	Usuario autenticado en el sistema GENESIG.
Propósito	Este caso de uso se lleva a cabo con el objetivo de gestionar los clientes que usarán los servicios del sistema
Resumen	Este caso de uso se inicia cuando el usuario accede al módulo, representado por un botón en el sistema GENESIG y culmina con algunas de las siguientes opciones sobre un cliente: guardar, editar o eliminar.
Precondiciones	El usuario debe estar autenticado en el sistema GENESIG y este sistema a su vez debe tener conexión a su servidor de bases de datos para poder utilizar todas las características del módulo.
Referencias	RF1, RF2., RF3, RF4
Prioridad	Alta
Flujo Normal de Eventos	
Acción del actor	**Respuesta del Sistema**

CAPÍTULO 2

1- El caso de uso inicia al acceder al botón que representa al módulo en el sistema GENESIG.	2- Muestra una interfaz con una tabla de varias columnas donde se listan los usuarios activos e inactivos en el momento y las opciones “adicionar y eliminar”.
3- El usuario selecciona la acción que desea realizar	

Prototipo de interfaz



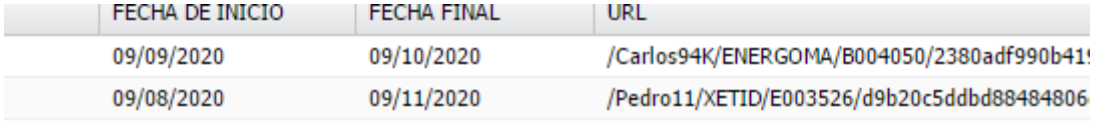
Opción Adicionar

Flujos Básicos

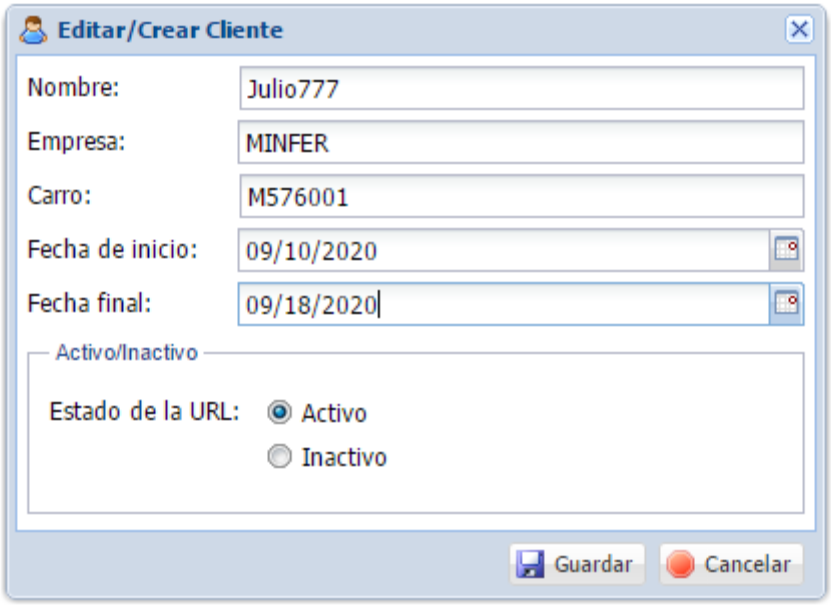
CAPÍTULO 2

Acción del actor	**Respuesta del Sistema**
1- El usuario da clic sobre el botón adicionar	2- Se muestra una ventana de un formulario para rellenar los datos solicitados
3- Se introducen los datos necesarios y selecciona el botón guardar.	4- Se verifica que no existan campos vacíos.
	5- Se crea una nueva fila que incluye la autogeneración de una URL que representa los datos del activo seleccionado en el formulario
Prototipo de interfaz	

CAPÍTULO 2



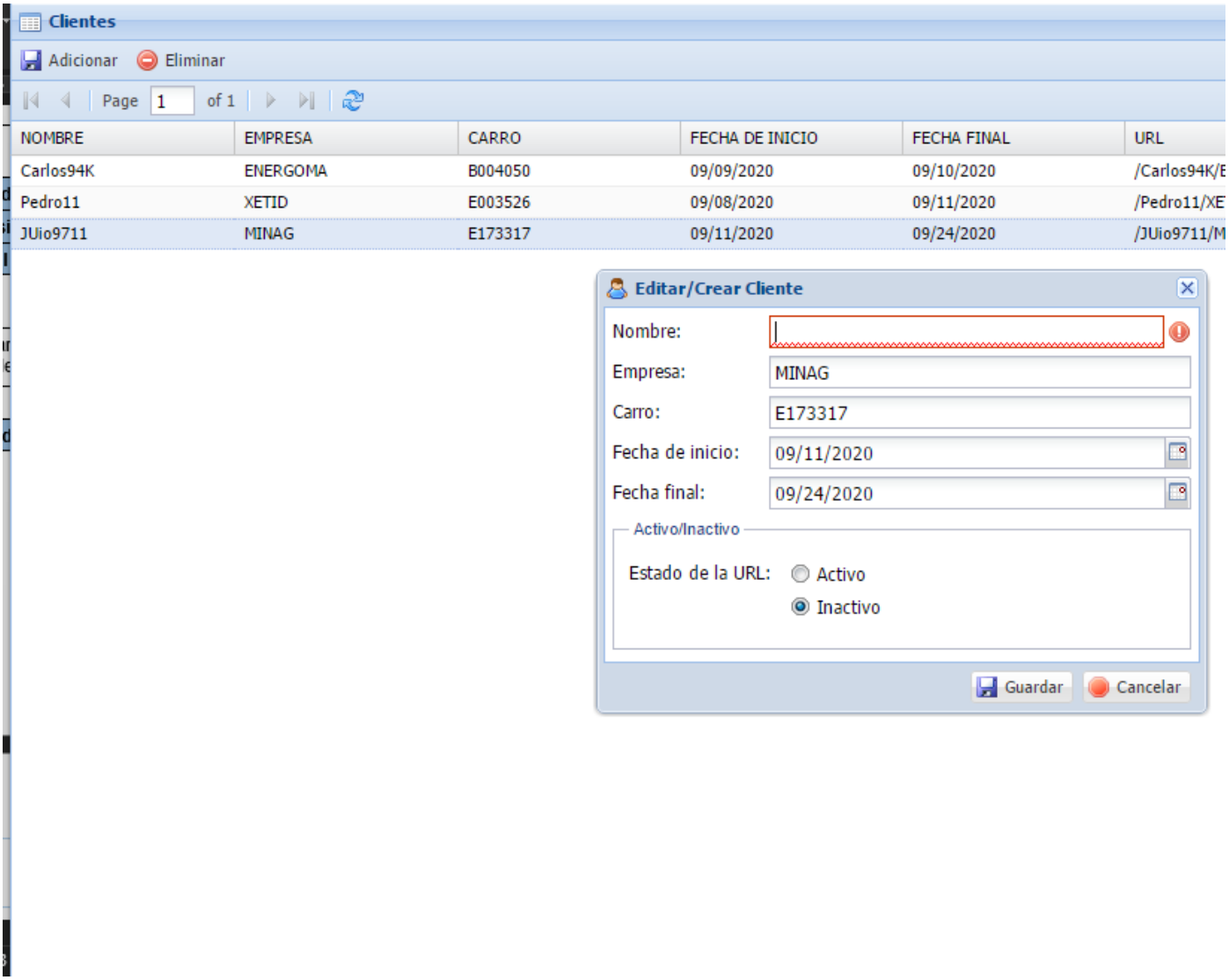
	FECHA DE INICIO	FECHA FINAL	URL
	09/09/2020	09/10/2020	/Carlos94K/ENERGOMA/B004050/2380adf990b41
	09/08/2020	09/11/2020	/Pedro11/XETID/E003526/d9b20c5ddbd88484806



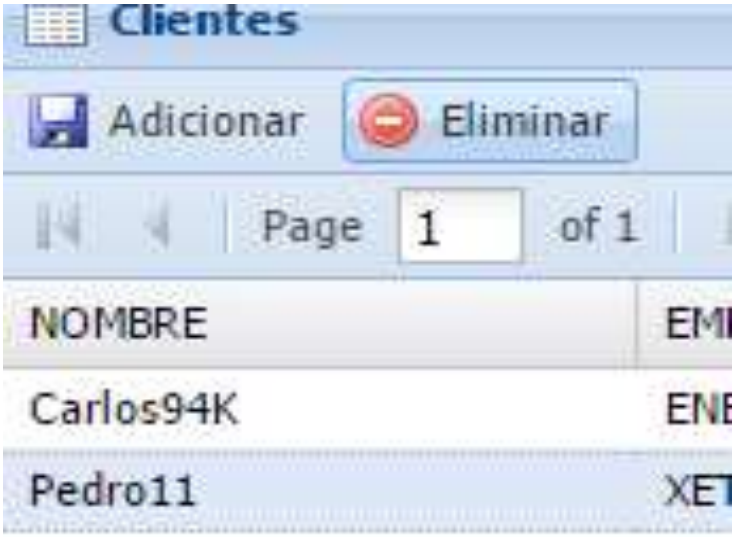
Flujos alternos	
	4- SI una de los campos se encuentra vacío se señala con una línea roja para obligarlo a rellenar
Opción Actualizar	
Flujos básicos	
Acción del actor	**Respuesta del sistema**

1- El usuario da doble clic izquierdo sobre la fila correspondiente al usuario que pretende modificar.	2- Se muestra una ventana de un formulario para editar los datos solicitados.
3- El usuario determina los datos que desea editar y selecciona el botón guardar.	4- SI una de los campos se encuentra vacío se señala con una línea roja para obligarlo a rellenar
Prototipo de interfaz	

CAPÍTULO 2



Flujos alternos	
Acción del actor	**Respuesta del sistema**
	4- SI una de los campos se encuentra vacío se señala con una línea roja para obligarlo a rellenar

<table>
<tr><th colspan="2">Opción Eliminar cliente</th></tr>
<tr><th colspan="2">Flujos básicos</th></tr>
<tr><th>Acción del actor</th><th>Respuesta del sistema</th></tr>
<tr><td>1- El usuario selecciona el cliente que desea eliminar dando un clic sobre la fila que lo representa quedando la misma marcada en azul.</td><td></td></tr>
<tr><td>2- El usuario da clic sobre el botón eliminar que se encuentra en el encabezado de la vista</td><td>3- El sistema borra la fila seleccionada, así como todos sus datos de la base de datos.</td></tr>
<tr><th colspan="2">Prototipo de interfaz</th></tr>
<tr><td colspan="2"> 
</td></tr>
<tr><th colspan="2">Flujos alternos</th></tr>
</table>

Acción del actor	Respuesta del sistema

Tabla 3 Diagrama de caso de uso del sistema "Crear URL". Fuente: elaboración propia

2.4.3 Patrones de casos de uso

La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con más precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento. Los patrones se presentan a modo de herramientas que permiten resolver los problemas que se les planteen a los desarrolladores de una forma ágil y sistemática. Estos patrones se enfocan hacia el diseño y las técnicas utilizadas en modelos de alta calidad, y no en cómo modelar usos específicos.(14)

Los patrones de casos de uso son los siguientes:

- **Reglas de negocio**
- **Concordancia (Commonality)**
- **Componente jerárquico (Component hierarchy)• Extensión concreta o Inclusión**
- **CRUD (Creating, Reading, Updating, Deleting)**
- **Caso de uso grande (Large Use case)**
- **Sistema de Capas**
- **Múltiples actores**
- **Servicio opcional**
- **Vistas ortogonales**

• **Secuencia de casos de uso**.

El patrón de caso de uso que se utilizó en la construcción del diagrama de casos de uso del Sistema fue:

CRUD Completo

Este patrón consiste en un caso de uso para administrar información, nos permite modelar las diferentes operaciones para administrar una entidad de información, tales como crear, Modificar, eliminar y listar. El mismo se manifiesta en los casos de uso: Gestionar Cliente(30) el cual posee las funciones CRUD.

2.5 Elementos fundamentales de la arquitectura

La arquitectura de software es la estructura o estructuras del sistema, que incluyen los componentes del software, las propiedades visibles externamente de esos componentes y las relaciones entre ellos. Debemos tener en cuenta que la arquitectura no es el sistema operativo es una representación que permite que un ingeniero del software analice la efectividad del diseño para cumplir con los requisitos establecidos, considere opciones arquitectónica en una etapa en que aún resulta relativamente fácil hacer cambios al diseño, reduzca los riesgos asociados con la construcción del software(31).

2.5.1 Patrones arquitectónicos y de diseño

Estilo Arquitectónico

Un estilo arquitectónico es una transformación impuesta al diseño de todo un sistema. El objetivo es establecer una arquitectura para todos los componentes del sistema. En caso de que una arquitectura existente se valla a someter a la reingeniería, la imposición de un estilo arquitectónico desembocará en cambios fundamentales en la estructura del software, incluida una reasignación de la funcionalidad de los componentes.

Patrón arquitectónico

Un patrón arquitectónico al igual que un estilo, impone una transformación en el diseño de una arquitectura. Sin embargo, un patrón difiere de un estilo de varios elementos fundamentales: el alcance de un patrón es

menor, ya que se concentra en un aspecto en lugar de hacerlo en toda la arquitectura, un patrón impone una regla sobre la arquitectura, pues describe la manera en que el software manejara algún aspecto de su funcionalidad al nivel de la infraestructura. Los patrones arquitectónicos tienden a abarcar aspectos específicos del comportamiento dentro del contexto de la arquitectura. Los patrones se usan junto con un estilo arquitectónico para determinar la forma de la estructura general de un sistema(31).

2.5.2 Patrón Modelo-Vista-Controlador

El MVC surge con el objetivo de reducir el esfuerzo de programación, necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos, a partir de estandarizar el diseño de las aplicaciones. El patrón MVC divide las partes que conforman una aplicación en el Modelo, las Vistas y los Controladores, permitiendo la implementación por separado de cada elemento, garantizando así la actualización y mantenimiento del software de forma sencilla. A partir del uso de *frameworks* basados en el patrón MVC se puede lograr una mejor organización del trabajo y mayor especialización de los desarrolladores y diseñadores(32).

- El Modelo representa toda la información con la que trabaja el sistema, es decir, su lógica de negocio.
- La Vista transforma el modelo en una página web que permite que los usuarios interactúen con el sistema.
- El Controlador se encarga de procesar las interacciones o peticiones realizadas por el usuario y realiza los cambios pertinentes tanto en el modelo como en la vista.

CAPÍTULO 2

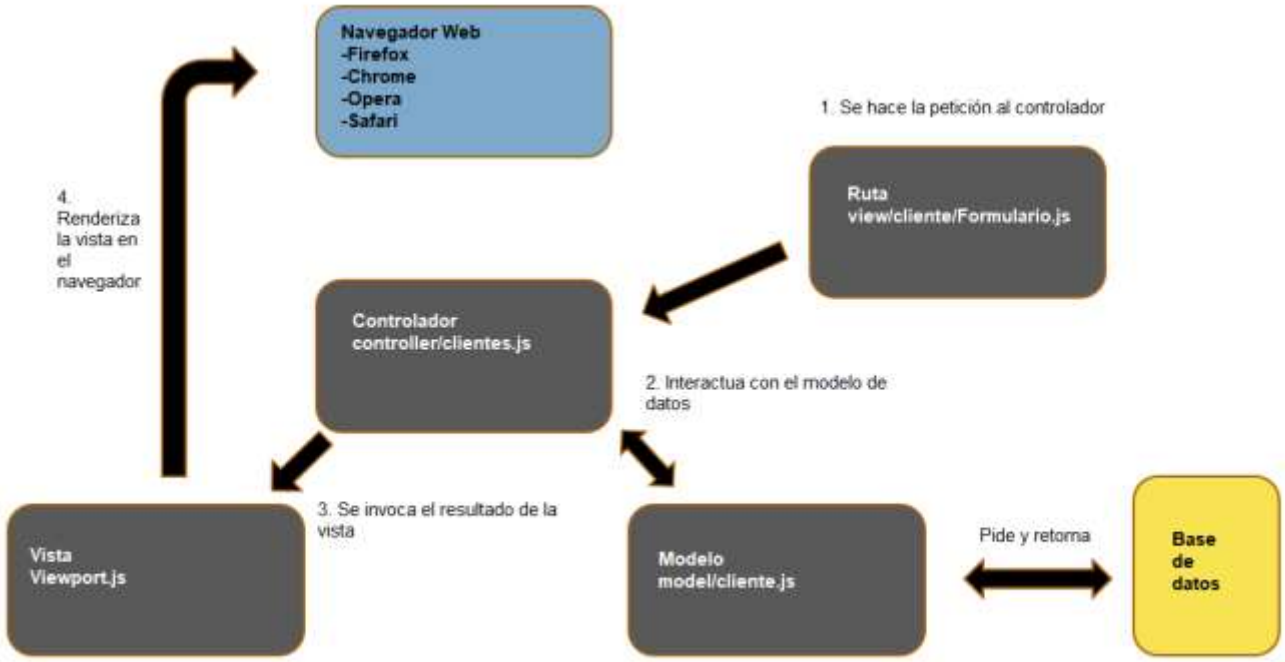


Ilustración 2.3 Ejemplo de MVC (Adicionar cliente) Fuente: elaboración propia

2.6 Patrones de Diseño

Los patrones de diseño son un conjunto de estrategias, o buenas prácticas, que pueden facilitar el trabajo en muchas situaciones a la hora de realizar una aplicación orientada a objetos. Los patrones GRASP describen los principios fundamentales del diseño de objetos para la asignación de responsabilidades. Por su parte los patrones GOF se clasifican en tres grupos: estructurales, creacionales y comportamiento. Los estructurales tratan la combinación de clases u objetos, su relación y la formación de estructuras de alta complejidad, mientras que los creacionales tratan la creación de instancias y los de comportamientos tratan la interacción y la cooperación entre clases(33).

Patrones GRASP

1. **Experto**: utilizado con el objetivo de darle a las clases las responsabilidades necesarias siempre que cuentan con la información para cumplirlas. Un ejemplo de este patrón se evidencia en el fichero controller/clientes.js que actúa como controlador de las acciones que se realicen con los clientes.
2. **Alta cohesión**: es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Se evidencia en las clases controladoras actualizarCliente.php, crearCliente.php, deletCliente.php las cuales tienen sus respectivas funciones que facilitan en flujo de datos de una mejor manera a tener todas las funciones en una sola clase.
3. **Controlador**: consiste en asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad, etc.). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. La arquitectura del marco de trabajo (MVC) brinda una capa específicamente para los controladores, que son el núcleo del mismo y especifica la presencia de este patrón. Todo esto está evidenciado en las clases controladoras del sistema como controller/cliente.js que realiza funciones específicas.
4. **Creador**: Este patrón guía la asignación de responsabilidades relacionada con la creación de objetos. Se observa en el sistema propuesto, dentro de la capa controlador correspondiente a cada uno de los casos de uso la cual distribuye las responsabilidades de comunicación con las instancias de la capa de presentación, asignándole a un grupo de clases la responsabilidad de crear instancias de las clases que sirven la información necesaria para su manejo(34). Cada una de las clases controladoras del sistema que gestionan entidades como cliente.js tiene la responsabilidad de crear dicho objeto (en este caso un cliente).

Patrones GoF

- **Patrón Singleton**: Este patrón garantiza que solamente se cree una instancia de las clases y provee un punto de acceso global a toda la aplicación ya que se encarga de enrutar todas las peticiones realizadas, es por ello que esta clase es utilizada por el controlador frontal de la aplicación y se evidencia su implementación en la clase app.js, utilizándose para activar los distintos componentes del sistema que son necesarios para ejecutar la acción encomendada por el usuario.
- **Patrón Decorator**: Este patrón se evidencia en la vista Viewport.js, padre de todas las vistas, que contiene un decorador para permitir agregar funcionalidades dinámicamente a cada página (gracias a

la función @extend), esta clase contiene elementos de estilos *css* y validaciones JavaScript y jQuery que contiene los elementos de diseño que son comunes para todas las páginas y de esta forma no tener que repetirlas en cada una(35).

2.7 Modelo de Diseño

El modelo del diseño se encarga de describir la realización física de los casos de uso y se corresponde directamente con los elementos físicos del ambiente de implementación. Consiste en colaboraciones de clases, que pueden ser agregadas en paquetes y/o subsistema(36).

2.7.1 Diagrama de clases del diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación y permiten modelar la vista de diseño del sistema. A continuación, el diagrama de clases del diseño para los casos de uso: Adicionar cliente, editar cliente, eliminar cliente y el Diagrama Entidad-Relación.

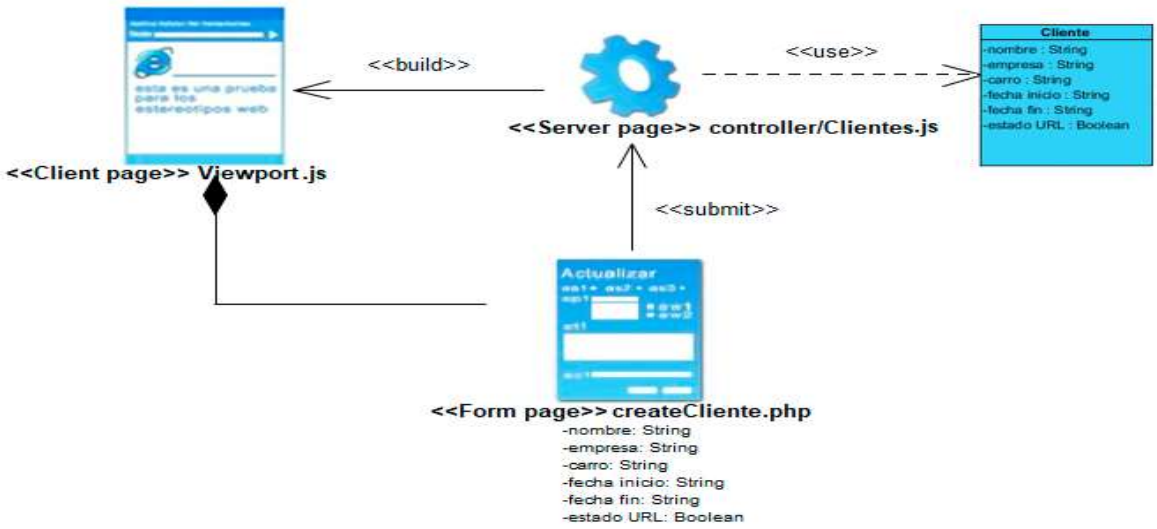


Ilustración 2.4 Diagrama de clases con estereotipos web: Crear cliente. Fuente: elaboración propia.

CAPÍTULO 2

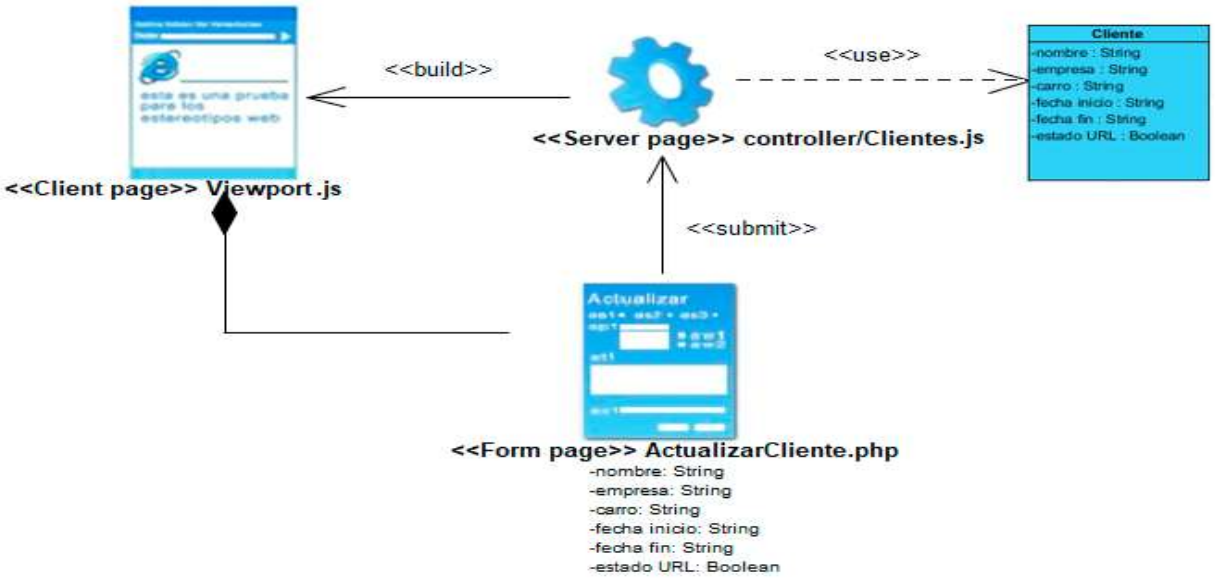


Figura 2.4 Diagrama de clases con estereotipos web: Actualizar cliente. Fuente: elaboración propia.

CAPÍTULO 2

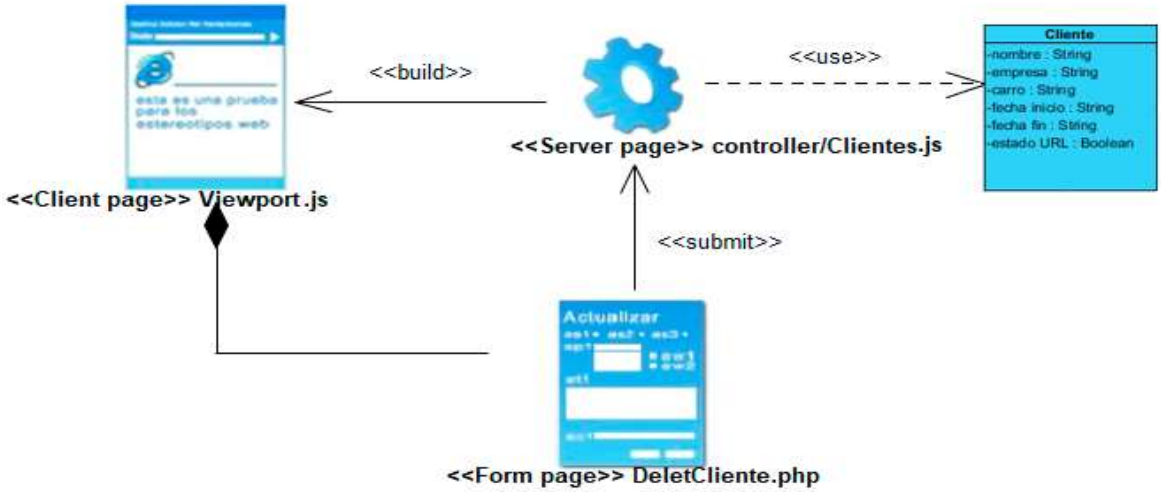


Ilustración 2.5 Diagrama de clases con estereotipos web: Eliminar cliente. Fuente: elaboración propia.

2.8 Diagrama de secuencia

Un diagrama de secuencias muestra la interacción de un conjunto de objetos de una aplicación a través del tiempo, en el cual se indican los módulos o clases que formaran parte del programa y las llamadas que se hacen cada uno de ellos para realizar una tarea determinada. Por esta razón permite observar la perspectiva cronológica de las interacciones. Es importante recordar que el diagrama de secuencias se realiza a partir de la descripción de un caso de uso(33).

Así como se representaron los Diagramas de clases del diseño para los CU de la gestión de clientes, también se muestran los diagramas de secuencia de los mismos.

CAPÍTULO 2

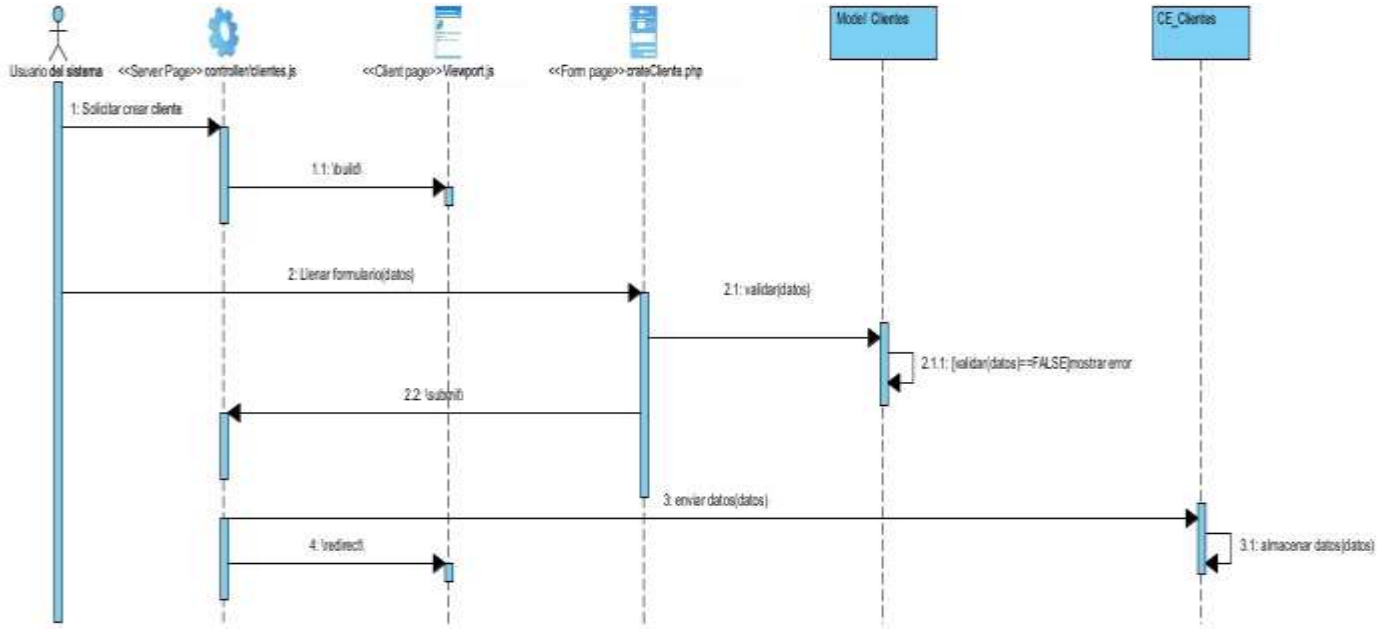


Figura 2.6 Diagrama de secuencia del caso de uso Crear cliente. Fuente: elaboración propia.

CAPÍTULO 2

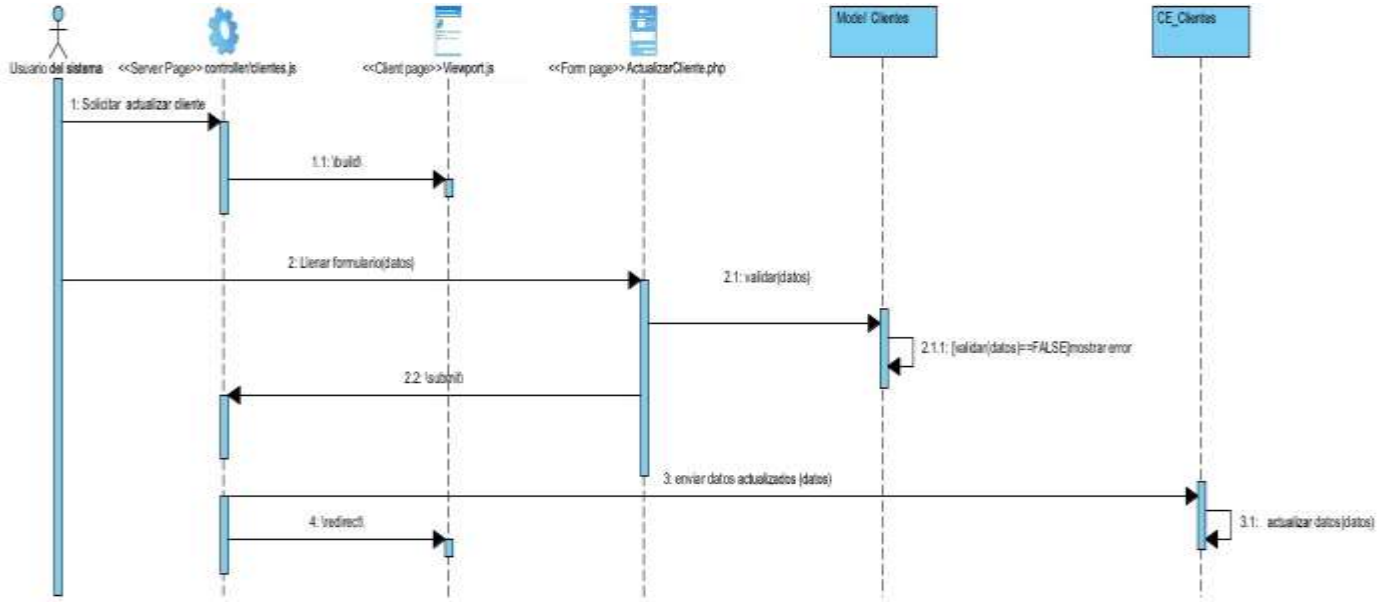


Figura 2.7 Diagrama de secuencia del caso de uso Actualizar cliente. Fuente: elaboración propia.

2.9 Conclusiones parciales

En este capítulo se especificó la propuesta de solución argumentada con una arquitectura de la información de dicha propuesta se determinó mediante la realización del modelo conceptual, los principales elementos que caracterizan la entidad donde será desplegada la solución. El diagrama de casos de uso del sistema diseñado facilitó la obtención de los requisitos funcionales y no funcionales, así como la descripción de los casos de uso del sistema con sus flujos e interfaces. Fue desglosado los elementos del patrón arquitectónico de la solución (Modelo-Vista-Controlador), así como los patrones de diseño empleados (GRASP y GoF). Se realizaron además los diagramas de clases con estereotipos web y los diagramas de secuencia que reflejan el flujo de la información en la solución, así como la respuesta dada por el sistema.

CONCLUSIONES

Conclusiones Generales

En la presente investigación se concluye lo siguiente:

- La elaboración del marco teórico metodológico permitió crear las bases del desarrollo de la investigación y constatar la novedad de la investigación.
- El estudio de homólogos del Sistema de Gestión Inteligente del Trasporte permitió obtener un grupo de elementos importantes asociados a la investigación.
- El análisis de las tecnologías, herramientas y lenguajes permitió definir la arquitectura de la del Sistema de Gestión Inteligente del Trasporte.
- El análisis y diseño aplicando la metodología definida permitieron identificar los requisitos funcionales y no funcionales, así como los artefactos ingenieriles que respaldan la solución propuesta, lo cual permitió cumplir con las expectativas del usuario.

BIBLIOGRAFÍA

Bibliografía:

1. MARÍA ESTELA RAFFINO. INroduccion TICS. *Concepto.de.* [online]. 2019. Available from: https://concepto.de/tics/#ixzz66hcx4pDC

2. EDUARDO HUERTA, ALDO MANGIATERRA and GUSTAVO NOGUERA. *GPS Posicionamiento Satelital*. UNR Editora, 2005.

3. RAE. Diccionario de la lengua española | Edición del Tricentenario. *«Diccionario de la lengua española» - Edición del Tricentenario* [online]. [Accessed 21 January 2020]. Available from: https://dle.rae.es/

4. Amazon.com. [online]. [Accessed 3 December 2019]. Available from: https://www.amazon.com/-/es/

5. BEZOS, Jeff. Amazon. [online]. 2019. Available from: https:/www.amazon.com

6. Correos de Cuba - EcuRed. [online]. [Accessed 4 December 2019]. Available from: https://www.ecured.cu/Correos_de_Cuba

7. DC, Redacción. Correos de Cuba: seguimiento de envíos postales. *Directorio Cubano 2019* [online]. 3 April 2019. [Accessed 10 December 2019]. Available from: https://www.directoriocubano.info/servicios/correos-de-cuba-seguimiento-de-envios-postales/

8. Correos de Cuba: seguimiento de envíos y paquetes | Postal Ninja. *Postal.Ninja* [online]. [Accessed 3 December 2019]. Available from: https://postal.ninja/es/p/correos-cuba/tracking

9. MIHAELA JUGANARU MATHIEU. *Introducción a la Programación*. 2014. Grupo Editorial Patria.

10. *1532804764_manual-javascript-19.pdf* [online]. [Accessed 27 February 2020]. Available from: http://88.27.64.3/files/1532804764_manual-javascript-19.pdf

BIBLIOGRAFÍA

11. ANGEL COBO, PATRICIA GÓMEZ, DANIEL PÉREZ and ROCÍO ROCHA. *PHP y MySQL Tecnologías para el desarrollo deaplicaciones web* [online]. Diaz de Santos, 2005. ISBN ISBN: 84-7978-706-6. Available from: http://zotero.org/support/quick_start_guide

12. ¿Qué es un Framework y para que sirve? - Neo Wiki. *Neoattack* [online]. 3 December 2019. [Accessed 3 December 2019]. Available from: https://neoattack.com/neowiki/framework/💻Un Framework es escencial para el desarrollo de diferentes programas. Si quieres conocer este concepto técnico de una forma sencilla, lee este Neowiki.

13. *girona-openlayers-workshop.pdf* [online]. [Accessed 27 February 2020]. Available from: https://buildmedia.readthedocs.org/media/pdf/girona-openlayers-workshop/latest/girona-openlayers-workshop.pdf

14. CRAIG, Larman. *UML y Patrones. 2ª Edición.* 2003. Prentice Hall.

15. ¿Qué son los IDEs y los editores de texto? [online]. 16 February 2020. [Accessed 16 February 2020]. Available from: https://platzi.com/blog/que-son-los-ides-y-los-editores-de-texto/

16. *phpstorm.pdf* [online]. [Accessed 27 February 2020]. Available from: https://riptutorial.com/Download/phpstorm.pdf

17. NGINX | High Performance Load Balancer, Web Server, & Reverse Proxy. NGINX [online]. [online]. 2019. [Accessed 3 December 2019]. Available from: https://platzi.com/blog/aplicaciones-escritorio-electron-js/

18. POSTGRESQL. About Postgresql. [online]. 2019. Available from: https://www.postgresql.org/about/

19. Software - CASE Herramientas. [online]. 2019. Available from: https://www.tutorialspoint.com/

BIBLIOGRAFÍA

20. PRESSMAN, Roger. *Ingeniería de Software. Un enfoque práctico. Parte 1*. Séptima edición. McGraw-hill, 2010.

21. About Visual Paradigm. [online]. [Accessed 10 December 2019]. Available from: https://www.visual-paradigm.com/aboutus/

22. MEDINA, Oscar Carlos, MARCISZACK, Marcelo Martín and GROPPO, Mario Alberto. Trazabilidad y validación de requerimientos funcionales de sistemas informáticos mediante la transformación de modelos conceptuales. *ReCIBE. Revista electrónica de Computación, Informática, Biomédica y Electrónica*. 2016. Vol. 1, no. 1.

23. RODRÍGUEZ CASTILLA, Liuris, GONZÁLEZ HERNÁNDEZ, Delly Lien and PÉREZ GONZÁLEZ, Yudeisy. De la arquitectura de información a la experiencia de usuario: Su interrelación en el desarrollo de software de la Universidad de las Ciencias Informáticas. *E-Ciencias de la Información*. 2017. Vol. 7, no. 1, p. 155–176.

24. RAMDHANI, Muhammad Ali, MAYLAWATI, Dian Sa'adillah, AMIN, Abdusy Syakur and AULAWI, Hilmi. Requirements elicitation in software engineering. *International Journal of Engineering & Technology (UEA)*. 2018. Vol. 7, no. 2.19, p. 772–775.

25. PRESSMAN, ROGER S. *Ingeniería de Software, un enfoque práctico*. Quinta edición. McGraw-Hill Companies, [no date]. ISBN ISBN 84-481-3214-9.

26. Obtención de Requerimientos. Técnicas y Estrategia | SG Buzz. [online]. [Accessed 27 February 2020]. Available from: https://sg.com.mx/revista/17/obtencion-requerimientos-tecnicas-y-estrategia

27. RAMDHANI, Muhammad Ali, MAYLAWATI, Dian Sa'adillah, AMIN, Abdusy Syakur and AULAWI, Hilmi. Requirements elicitation in software engineering. Functional requirements. *International Journal of Engineering & Technology (UEA)*. 2018. Vol. 7, no. 2.19.

BIBLIOGRAFÍA

28. RAMDHANI, MUHAMMAD ALI, MAYLAWATI, DIAN SA'ADILLAH, AMIN, ABDUSY SYAKUR AND AULAWI, HILMI. *Requirements elicitation in software engineering. Functional requirements. International Journal of Engineering & Technology (UEA)*. 2018.

29. *Ingeniería del Software. Un enfoque práctico. 7ma Ed (1).pdf*.

30. SCHMULLER, JOSEPH. *Learn UML in 24 hours. Full CRUD pattern. Third Edition. Indiana, USA*. Sams Publishing, [no date].

31. RICHARDS, MARK. *Software architecture patterns. Styles and Design. O'Reilly Media, Incorporated 1005 Gravenstein Highway North, Sebastopol, CA*. 2015.

32. SHARAN, KISHORI. *Model-view-controller pattern. In : Learn JavaFX 8. Springer,*. 2015.

33. CONNOLLY, RANDY. *Fundamentals of web development.* Pearson Education, 2015.

34. CONNOLLY, RANDY. *Fundamentals of web development. GRASP patterns. Sumary.* Pearson Education, 2015.

35. HUSSAIN, SHAHID, KEUNG, JACKY AND KHAN, ARIF ALI. *Software design patterns classification and selection using text categorization approach. Applied soft computing. 2017. Vol. 58, p. 225–244*. [no date].

36. ROSSI, BRUNO. *Entity relationship diagram*. 2014.