UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:

INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:

INGENIEROS DE SISTEMAS

TEMA:

ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN GESTOR DE TRANSPORTE AUTOMOTRIZ PARA LA SUPERINTENDENCIA DE CONTROL DEL PODER DE MERCADO

AUTORES:

FABRICIO XAVIER PILICITA GARRIDO

JORGE ALEXIS ZÚÑIGA VILLAMAR

TUTOR:

JOSÉ LUIS VILLAGÓMEZ MENÉNDEZ

Quito, marzo de 2016

CESIÓN DE DERECHOS DE AUTOR

Nosotros, Fabricio Xavier Pilicita Garrido, con documento de identificación N° 1724528094, y Jorge Alexis Zúñiga Villamar, con documento de identificación N° 1722624226 respectivamente, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del trabajo de grado intitulado: ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN GESTOR DE TRANSPORTE AUTOMOTRIZ PARA LA SUPERINTENDENCIA DE CONTROL DEL PODER DE MERCADO, mismo que ha sido desarrollado para optar por el título de INGENIEROS DE SISTEMAS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autores nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Politécnica Salesiana.

.....	.....
FABRICIO XAVIER PILICITA GARRIDO	JORGE ALEXIS ZÚÑIGA VILLAMAR
CI: 1724528094	CI: 1722624226

Fecha: marzo de 2016

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo, declaro que bajo mi dirección y asesoría fue desarrollado el trabajo de titulación ANÁLISIS, DISEÑO, DESARROLLO E IMPLEMENTACIÓN DE UN GESTOR DE TRANSPORTE AUTOMOTRIZ PARA LA SUPERINTENDENCIA DE CONTROL DEL PODER DE MERCADO realizado por Fabricio Xavier Pilicita Garrido y Jorge Alexis Zúñiga Villamar, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana para ser considerados como trabajo final de titulación.

Quito, marzo de 2016

.....

JOSÉ LUIS VILLAGÓMEZ MENÉNDEZ

CI: 1307794360

ÍNDICE

INTRODUCCIÓN 1

CAPÍTULO 1 2

ESTADO DEL ARTE 2

1.1. Antecedentes 2

1.2. Justificación 7

1.3. Objetivo general 8

1.4. Objetivos específicos 8

1.5. Marco metodológico 9

1.5.1. Metodología UWE (UML Web Engineering) 11

1.5.1.1. Diagrama de casos de uso. 13

1.5.1.2. Diagrama de clases. 13

1.5.1.3. Diagrama de navegación. 14

1.5.1.4. Diagrama de presentación. 14

1.5.1.5. Diagrama de estructura de proceso. 14

1.5.1.6. Diagrama de flujo de proceso. 14

1.5.2. Metodología XP. 14

1.6. Marco institucional 16

1.6.1. La Superintendencia de Control del Poder de Mercado (SCPM). 16

1.7. Marco teórico 18

1.7.1. Programación Orientada a Objetos. 18

1.7.1.1. Abstracción. 20

1.7.1.2. Encapsulación. 21

1.7.1.3. Polimorfismo. 21

1.7.1.4. Herencia. 21

1.7.2. Lenguaje Unificado de Modelado. 22

1.7.3. Modelo Vista Controlador (MVC). 26

1.7.3.1. El modelo. 26

1.7.3.2. Las vistas. 27

1.7.3.3. El controlador. 27

1.7.4. Bases de datos relacionales. 28

1.7.5. Gestión de activos. 30

CAPÍTULO 2 32

ANÁLISIS Y DISEÑO 32

2.1. Análisis de requisitos 32

2.1.1. Caso de uso solicitud de movilización. 33

2.1.1.1. Descripción de caso de uso solicitud de movilización. 33

2.1.2. Caso de uso registro de movilización de vehículos. 36

2.1.2.1. Descripción de caso de uso registro de movilización de vehículos. 36

2.1.3. Caso de uso gestión de vehículos. 38

2.1.3.1. Descripción de caso de uso gestión de vehículos. 38

2.2. Modelo de contenido 40

2.2.1. Diagrama de clases del Sistema de Gestión de Transporte. 40

2.2.2. Clases de vistas y controladores. 42

2.2.3. Clases del modelo. 43

2.3. Modelo de navegación 44

2.3.2. Navegación del perfil empleado..... 46

2.3.3. Navegación del perfil personal de seguridad. 48

2.3.4. Navegación del perfil administrador financiero..... 49

2.3.5. Navegación del perfil súper administrador. 50

2.4. Modelo de presentación 51

2.4.1. Diagramas de Presentación. 52

2.4.1.1. Vehículos..... 52

2.4.1.2. Solicitud de movilización..... 54

2.4.1.3. Preautorización de solicitud. 55

2.4.1.4. Asignación..... 55

2.4.1.5. Solicitudes que requieren archivo adjunto. 57

2.4.1.6. Registro de ingreso de vehículo particular..... 57

2.4.1.7. Registro de salida de vehículo particular. 59

2.4.1.8. Registro de entrada y salida de vehículo institucional..... 59

2.4.1.9. Registro de salida no autorizada de vehículo institucional. 60

2.5. Modelo de proceso 61

2.5.1. Diagramas de estructura de proceso..... 62

2.5.1.1. Gestión de vehículo..... 62

2.5.1.2. Gestión de solicitud..... 63

2.5.1.3. Preautorización de solicitud. 64

2.5.1.4. Autorización de solicitud. 65

2.5.1.5. Registro de ingreso y salida de vehículos. 66

2.5.2. Diagramas de flujo de proceso..... 68

2.5.2.1. Creación de vehículo. 68

2.5.2.2. Creación de solicitud. 69

2.5.2.3. Preautorización de solicitud. 71

2.5.2.4. Autorización de solicitud. 71

2.5.2.5. Registro de recorrido de vehículo institucional. 72

2.5.2.6. Registro de arribo de vehículo de transporte. 74

2.5.2.7. Registro de historial de vehículo particular. 75

CAPÍTULO 3 78

CONSTRUCCIÓN Y PRUEBAS 78

3.1. Herramientas 79

3.2. Diagrama de base de datos 79

3.3. Construcción MVC 80

3.3.1. Capa de datos. 81

3.3.2. Capa de presentación. 82

3.3.3. Capa de negocio. 83

3.4. Construcción de aplicación móvil 84

3.4.1. Proyecto WebViewApp. 85

3.5. Construcción de reportes 89

3.6. Implementación 91

3.6.1. Implementación de la aplicación web en Internet Infomation Services 91

3.6.2. Implementación de los reportes en Reporting Services 93

3.7. Pruebas 95

3.7.1. Pruebas de unidad. 96

3.7.2. Pruebas de rendimiento. 97

3.7.3. Pruebas de carga y estrés 98

CONCLUSIONES 100

RECOMENDACIONES 102

REFERENCIAS 103

ANEXOS 106

ÍNDICE DE FIGURAS

Figura 1. Diagrama de caso de uso de solicitud de movilización 36

Figura 2. Diagrama de caso de uso de registro de movilización de vehículos 38

Figura 3. Diagrama de caso uso de gestión de vehículos 39

Figura 4. Diagrama de clases 41

Figura 5. Diagrama de navegación del perfil administrador de movilización 45

Figura 6. Diagrama de navegación del perfil empleado 47

Figura 7. Diagrama de navegación del personal de seguridad 48

Figura 8. Diagrama de navegación del perfil administrador financiero 50

Figura 9. Diagrama de navegación del perfil súper administrador 51

Figura 10. Diagrama de presentación de vehículos 53

Figura 11. Diagrama de presentación de solicitud de movilización 54

Figura 12. Diagrama de presentación de preautorización de solicitud 55

Figura 13. Diagrama de presentación de asignación 56

Figura 14. Diagrama de presentación de solicitudes que requieren archivo adjunto 57

Figura 15. Diagrama de presentación de registro de ingreso de vehículo particular 58

Figura 16. Diagrama de presentación de registro de salida de vehículo particular 59

Figura 17. Diagrama de presentación de registro de entrada y salida de vehículo institucional 60

Figura 18. Diagrama de presentación de registro de salida no autorizada de vehículo institucional 61

Figura 19. Diagrama de estructura de proceso de gestión de vehículo 62

Figura 20. Diagrama de estructura de proceso de gestión de solicitud 63

Figura 21. Diagrama de estructura de proceso de preautorización de solicitud.....64
Figura 22. Diagrama de estructura de proceso de autorización de solicitud.....65
Figura 23. Diagrama de estructura de proceso de registro de ingreso y salida de vehículos 66
Figura 24. Diagrama de flujo de proceso de creación de vehículo 68
Figura 25. Diagrama de flujo de proceso de creación de solicitud 70
Figura 26. Diagrama de flujo de proceso de preautorización de solicitud.....71
Figura 27. Diagrama de flujo de proceso de autorización de solicitud.....72
Figura 28. Diagrama de flujo de proceso de registro de recorrido de vehículo institucional.....73
Figura 29. Diagrama de flujo de proceso de registro de arribo de vehículo de transporte 75
Figura 30. Diagrama de flujo de proceso de registro de historial de vehículo particular 76
Figura 31. Metodología UWE y XP.....78
Figura 32. Diagrama de bases de datos.....80
Figura 33. Solución creada en IDE 81
Figura 34. Proyecto Base 81
Figura 35. Proyecto FontEnd 82
Figura 36. Capa de negocio.....83
Figura 37. Proyecto APP.....85
Figura 38. Archivo Main.axml.....86
Figura 39. Archivo MainActivity.cs 86
Figura 40. Modo Release 87
Figura 41. Versión Android 87

Figura 42. Archivos generados 88

Figura 43. Aplicación móvil 88

Figura 44. Proyecto de servidor de informes 89

Figura 45. Creación de informes 90

Figura 46. Administrador de IIS 91

Figura 47. Nueva aplicación web 92

Figura 48. Árbol de IIS 92

Figura 49. Aplicación web desde el navegador 93

Figura 50. Configuración del servidor Reporting Services 93

Figura 51. URL del servidor de informes 94

Figura 52. Publicación de informes 94

Figura 53. Informes publicados 95

Figura 54. Visualización de reportes 95

Figura 55. Configuración de prueba de carga 98

Figura 56. Petición HTTP 99

Figura 57. Reporte grafico de pruebas de carga y estrés 99

ÍNDICE DE TABLAS

Tabla 1. Fases de la Metodología UWE....12

Tabla 2. Objetivos del modelo de Codd....29

Tabla 3. Administración de activos....30

Tabla 4. Curso normal de una solicitud de movilización....34

Tabla 5. Cursos alternos de una solicitud de movilización....35

Tabla 6. Curso normal del registro de movilización de vehículos....37

Tabla 7. Cursos alternos del registro de movilización de vehículos....37

Tabla 8. Curso normal de gestión de vehículos....40

Tabla 9. Clases de vistas y controladores....42

Tabla 10. Clases de Modelo....43

Tabla 11. Informes generados....90

Tabla 12. Pruebas de unidad....96

Tabla 13. Pruebas de rendimiento....97

Tabla 14. Resultado de pruebas de carga y estrés....99

ÍNDICE DE ANEXOS

Anexo 1. Especificación de requisitos….....…….....…106

Anexo 2. Diccionario de datos…….....…106

Anexo 3. Pruebas de unidad del Sistema de Gestión de Transporte……..…106

Anexo 4. Pruebas de rendimiento del Sistema de Gestión de Transporte……....106

Anexo 5. Manuales de usuario……106

RESUMEN

La aplicación web que se desarrolló plantea solventar la necesidad de automatización de procesos que busca la Superintendencia de Control del Poder de Mercado (SCPM) dentro del ámbito de gestión de vehículos creando un sistema web de Gestión de Transporte que se encargue de administrar la información referente a los vehículos pertenecientes a la institución o particulares que entren a las instalaciones de la misma.

También permitirá gestionar solicitudes de movilización que serán creadas por los empleados de la superintendencia. El administrador del portal se encargará de autorizar las solicitudes asignándole un vehículo y un conductor o negarla según considere conveniente.

El documento está dividido en 3 capítulos los cuales se detallan a continuación.

El capítulo uno presenta información referencial a la Superintendencia de Control del Poder del Mercado, el motivo por el que fue creada y sus objetivos. Además se presenta el marco teórico que contiene los conceptos necesarios para la construcción de la solución informática.

El capítulo dos presenta el análisis y el diseño del sistema. Se centra en definir los requerimientos del cliente y a crear la solución del problema utilizando una metodología de diseño de software junto con sus respectivos diagramas.

El capítulo tres presenta las herramientas utilizadas, la construcción de la aplicación y las pruebas realizadas para asegurar el correcto funcionamiento de la solución informática, para finalizar con las conclusiones y recomendaciones que se obtuvieron del trabajo realizado.

ABSTRACT

The web application that was developed to meet the need arises automation processes search the Superintendencia de Control del Poder del Mercado(SCPM) within the scope of vehicle management creating a web Transportation Management system that is responsible for managing information concerning vehicles belonging to the institution or individuals entering the premises of the same. It will also allow management applications to mobilize that will be created by employees of the superintendency. The portal administrator is responsible for assigning requests to authorize a vehicle and a driver or deny as it deems appropriate.

The document is divided into 3 chapters which are detailed below.

Chapter one presents reference to the Superintendencia de Control del Poder del Mercado information, why it was created and objectives. Besides the theoretical framework that contains the necessary concepts for building the software solution it is presented.

Chapter two presents the analysis and design of the system. It focuses on defining customer requirements and create the solution using a software design methodology with their respective diagrams.

Chapter three presents the tools used to build the application and testing to ensure the proper functioning of the software solution, to finish with the conclusions and recommendations obtained from the work done.

INTRODUCCIÓN

Los requerimientos actuales a los que están sujetos los procesos de desarrollo de software son de vital importancia para proyectos que provienen del sector público. Esto puede brindar una perspectiva de amplio análisis y cierta consideración con posibles limitaciones y oportunidades que pueden surgir a lo largo del desarrollo de un trabajo de esta naturaleza. Un claro ejemplo es la automatización de procesos que busca la Superintendencia de Control del Poder de Mercado (SCPM) dentro del ámbito de gestión de vehículos, el cual se basa fundamentalmente en un cambio tecnológico con el fin de solventar necesidades reales e implantar soluciones de problemas que posee la entidad. La finalidad del trabajo que se ha desarrollado es la implementación de un Gestor de Transporte que se encargue de administrar la información referente a la gestión vehicular de la institución, con el objetivo de brindar una solución en base a la situación actual de los conocimientos de desarrollo en este tipo de aplicaciones. De igual forma se ha estudiado varios ámbitos en los cuales se ha aplicado este tipo de sistemas, los mismos que se pueden apreciar a continuación.

CAPÍTULO 1

ESTADO DEL ARTE

1.1. Antecedentes

Un sistema de gestión de movilización tiene un gran impacto en varios ámbitos tanto de estudio e implementación, debido a lo siguiente: la mayoría de los procesos son manuales y costosos en tiempo, utilizan la capacidad de un recurso humano dedicado, y acumulan continuamente grandes cantidades de documentos físicos. Esto resulta ineficiente y posteriormente puede conllevar problemas legales, aún más, en una institución pública que es en donde se desarrollará el proyecto ya que estas cuentan con normativas que deben ser cumplidas y en caso de no realizarlo iniciarían procesos legales.

En el año 2003 el Contralor General del Estado emite el acuerdo N007-CG-2003 en donde expide el reglamento para el control del sector público, que permite normar la utilización de los vehículos pertenecientes al Estado, sus conductores y mantenimiento correspondiente. También se detalla los trámites de órdenes de movilización que deben ser efectuados y los registros estadísticos que se deben llevar por parte de cada institución.

Es por este antecedente que es de vital importancia la creación de un sistema de gestión vehicular para la SCPM ya que permitirá cumplir con esta disposición.

En el ámbito académico del Ecuador, los sistemas de gestión vehicular que permitan cumplir con esta normativa han sido desarrollados en diferentes instituciones del Estado, al respecto se puede mencionar la tesis que en el año 2012, Patricia Alejandra Terán Sevilla presenta: Titulada “Análisis, Diseño E Implementación De Un Sistema De Control

Vehicular En Instituciones Públicas”[1]. En esta tesis se analiza, diseña e implementa un sistema de control vehicular utilizando herramientas de software libre para la empresa municipal de agua potable y saneamiento ambiental del cantón espejo que es una institución del sector público encargada de la prestación de servicios de agua potable y saneamiento ambiental.

La Empresa Municipal de Agua Potable y Saneamiento Ambiental anteriormente realizaba un control vehicular de una manera informal y no acorde a las exigencias de las entidades de control gubernamentales. Las órdenes de movilización, partes de novedades y accidentes se aplicaban de forma manual y se encontraban en documentos preimpresos, lo cual implicaba un registro desordenado e inexacto de la información. El inadecuado manejo de la administración de los vehículos de la institución, dificulta el registro de entrada y salida de los mismos; y su vez la entrega y recepción a los conductores.

Cabe mencionar que “El sistema informático permite conocer la disponibilidad de un vehículo en una determinada fecha, dispone de la cantidad adecuada de lubricantes y combustibles en los vehículos, mantenimientos oportunos, mejorando la vida útil de los vehículos” (Terán, 2012).

Otra tesis que intenta dar cumplimiento a la disposición de la Contraloría General el Estado es la de Reyes y Urgiles que en el año 2007 presentan su tesis titulada “Modelo Administrativo Para Uso Y Control De La Flota Vehicular Del Cuerpo De Ingenieros Del Ejército Del Grupo Mazar”[2] para la obtención de sus títulos de ingenieros mecánico

[1] Tesis desarrollada por Terán (2012).
[2] Tesis desarrollada por Reyes y Urgiles (2007).

automotriz en la Universidad Politécnica Salesiana en la carrera de Ingeniería Mecánica Automotriz. En esta tesis se plantea todo un estudio de la situación actual de la flota vehicular, sus costos y especificaciones técnicas, así como gastos del funcionamiento de la flota, que incluyen mano de obra, mantenimiento, combustibles, lubricantes, etc.

Lo que interesa de esta tesis tan amplia es el software para la gestión vehicular que desarrollaron con el objetivo de hacer aplicable los puntos más importantes expuestos en su estudio.

Más concretamente los autores se enfocan en el proceso de utilización de los vehículos, ya que estos se deben registrar con su respectivo conductor, la obra en la cual trabajarán, la jornada de trabajo, kilometrajes y horas de salida y llegada. Reyes y Urgiles afirman: "El mantener un adecuado sistema de control es fundamental para lograr un comportamiento eficiente de la empresa y generar una cultura de control del uso que tenga en cuenta los factores, requisitos, objetivos, adaptación, flexibilidad y limitaciones del mismo" (Reyes & Urgiles, 2007).

Como se puede observar, un sistema que permita la gestión vehicular ha sido profundamente explotado en Ecuador tanto en instituciones públicas como en universidades ya que cada entidad requiere un sistema a su medida y es por este motivo que se desarrollan sistemas separados que si bien intentan dar cumplimiento a la misma disposición legal, éstas varían dependiendo del contexto de la institución ya que no es lo mismo por ejemplo desarrollar un sistema de gestión vehicular para un municipio que para el cuerpo de ingenieros del ejército ya que las dos instituciones tienen requerimientos diferentes.

Esto no solo aplica en el Ecuador y en el sector público, ya que en el mundo entero existe la necesidad por parte de los gobiernos y empresas privadas de mantener un control de sus activos, ya que estos incluyen a sus flotas vehiculares. Se puede destacar la tesis de Rivera Trejo Juan Pablo que se titula "Sistema de Control de Acceso", como parte de los requisitos para obtener su título de Ingeniero en Tecnologías de Automatización en la Universidad Tecnológica de Querétaro (Rivera Trejo, 2014). En esta tesis se desarrolla una propuesta de implementación de un control de acceso vehicular mediante un sistema de identificación por radiofrecuencias para las empresas Servicios Pontiac S.A. y Grúas S.A., cuya finalidad será controlar las entradas y salidas de los vehículos pertenecientes a la empresa.

La problemática de la empresa surgió debido a que no se contaba con un control de acceso vehicular que administre entradas y salidas de los vehículos tanto internos como externos, al no contar con un sistema de control se generaban problemas con relación al control de acceso autorizados y los no autorizados.

Era necesario un control que genere un registro de acceso para cada vehículo que determine la hora y fecha de su entrada o en defecto su salida con la finalidad de tener un sustento de las horas de entrada y salida.

Ya que no se contaba con ninguna medida sobre los accesos, cualquier vehículo podía acceder al estacionamiento de la empresa generando conflictos.

El objetivo de la propuesta fue ofrecer un control de acceso vehicular que cumpla con los requisitos de la empresa. Rivera Trejo afirma lo siguiente.

> Desarrollando una propuesta de un Sistema de Control de Acceso Vehicular que administrará las entradas y salidas de los vehículos internos mediante la automatización de una barrera vehicular, a la vez que también se generará un registro de los eventos de acceso de cada uno de los vehículos ya que estos tendrán una identificación única y por lo tanto cada vez que entre o salga un vehículo se sabrá que unidad era (Rivera Trejo, 2014).

El contar con un sistema de gestión vehicular para el sector público y privado ha sido causa de interés por parte de las empresas y gobiernos ya que las leyes y normativas de cada país e institución buscan regularizar este ámbito, por tal motivo los desarrolladores tanto nacionales como extranjeros han intentado solventar esta necesidad, una muestra de esto son las tesis expuestas anteriormente en las que se puede observar que este campo de estudio ha sido explotado debido a su gran nivel de importancia.

Si bien el sistema que se desarrollará al final de este proyecto también intenta suplir la misma necesidad final que es el control vehicular, difiere con las anteriormente presentadas porque trata de ser un sistema construido en base a los requerimientos específicos para la Superintendencia de Control del Poder del Mercado permitiendo registrar órdenes de movilización, listando los vehículos disponibles en ese día y a esa hora, lo cual optimizará el uso de los vehículos y permitirá generar reportes para la toma de decisiones.

1.2. Justificación

La Superintendencia de Control del Poder de Mercado (SCPM), siendo una institución del Estado, recientemente creada, vinculada al servicio ciudadano, está interesada en alcanzar sus objetivos y brindar sus servicios de la manera eficaz y eficiente, por tal motivo se ve en la necesidad de optimizar sus procesos, lo que implica mejorar los mismos a través de la implementación de soluciones informáticas que permitan la reducción de tiempo en la ejecución de actividades y la priorización de prácticas más amigables con el medio ambiente.

Uno de los procesos que se desea automatizar es el control que se realiza a los vehículos pertenecientes a la SCPM. El procedimiento actual consiste en llevar un registro manual de la entrada y salida de automotores propios y ajenos por medio de una bitácora de control a cargo del personal de seguridad de la institución. De igual manera se procede con los vehículos de transporte diario destinados al uso del personal de la institución, a través de un control de los mismos en la entrada y salida laboral donde se registra el kilometraje, junto con el conductor designado.

Actualmente el control de todos estos procesos se lleva a cabo por medio de archivos planos, lo que implica el registro de cada transacción mediante formularios de solicitudes que deben ser llenadas manualmente por los funcionarios de la institución. Estas tareas forman parte de un mecanismo de control que posee desventajas; una de ellas consiste en la dificultad de realizar procesos de búsqueda de solicitudes aprobadas, negadas y las que se encuentren pendientes, así como la obtención de registros específicos, indicadores, e información relevante para la SCPM.

Con la implementación de un sistema informático que se encargue de gestionar de manera adecuada cada una de las tareas antes mencionadas para el uso y control de vehículos, se puede lograr un mejor seguimiento de cada proceso, registrar sus falencias, realizar planes y consultar información; es decir que desde el sistema sea posible generar un conjunto de reportes consolidados que permita la construcción de indicadores de información primordial para la institución. Cabe mencionar que algunos de los beneficios es la política de cero papeles, automatización de procesos de obtención de información, seguimiento de solitudes en línea y un control optimizado en la asignación de vehículos.

1.3. Objetivo general

Implementar un sistema informático de gestión de transporte automotriz para la Superintendencia de Control del Poder del Mercado, con la finalidad de optimizar los procesos de control y la asignación de vehículos.

1.4. Objetivos específicos

1. Analizar los requerimientos funcionales en cuanto al control del uso vehicular de la institución, para satisfacer las necesidades de los involucrados en cada una de las tareas de los procesos a optimizar.

2. Diseñar la arquitectura de la aplicación con el objetivo de partir de una base sólida tomando en cuenta el análisis previo en cuanto a funcionalidad y factibilidad técnica para un adecuado desarrollo del sistema informático.

3. Desarrollar los productos entregables requeridos con el fin de que los avances planificados sean controlados y verificados tanto por la SCPM como por la Universidad Politécnica Salesiana.

4. Realizar pruebas de funcionalidad, de integración, de carga y demás, para proporcionar un producto con calidad.

5. Implantar el sistema informático en la institución con el fin de que el mismo optimice los procesos previstos y brinde los resultados esperados.

1.5. Marco metodológico

El desarrollo de software a lo largo de la historia tecnológica ha sido un punto de vital importancia para la sociedad, debido a que el mismo tiene como objetivo brindar un producto que solvente una necesidad especifica o solucione un problema. El desarrollo en cierta forma implica varias facetas como el descubrimiento del problema, el análisis de una solución y la codificación de la misma, junto con sus respectivas pruebas.

En los tiempos modernos el desarrollo de software se lo realiza en base a metodologías, las cuales plantean herramientas, técnicas, teorías, y buenas prácticas, con el objetivo de mitigar los problemas que puedan surgir, así como también pueden guiar a este proceso con ciertas pautas que son de gran utilidad durante el transcurso del desarrollo de una aplicación. En resumen, se puede afirmar que una metodología de desarrollo de software es una forma de realizar ciertas tareas con el fin de alcanzar un objetivo planeado.

Existen varios tipos de metodologías de desarrollo de software que utilizan y combinan varios modelos como el cascada, espiral, incremental, iterativo, prototipos, agiles, etc. Cada modelo plantea un camino que difiere entre los demás, lo cual implica la utilización de diferentes procesos y métodos. Cabe recalcar que el ámbito donde se aplique el desarrollo puede influir con la selección de una metodología, por ejemplo: en tipos de software como bancario, empresarial, educativo, seguro, hospitalario, etc.; así como en

tipos de usuarios, los cuales pueden variar con respecto al grado de involucramiento en el desarrollo del software, así como el nivel de aportación al mismo.

Las metodologías de desarrollo de software son cruciales para lograr un impacto favorable en proyectos de software, debido a que se toma muy en cuenta las fases de desarrollo como: análisis, diseño, construcción, pruebas, entrega y mantenimiento; las cuales se las realiza de acuerdo a técnicas y herramientas propias de la metodología.

En el presente proyecto por requerimiento de la SCPM se ha utilizado una metodología enfocada en el desarrollo de software como lo es UWE ya que ésta proporciona herramientas enfocadas en el análisis y diseño para facilitar la construcción y el mantenimiento de la aplicación.

Por parte de los desarrolladores se escogió una metodología ágil como XP, ya que al inicio del proyecto los requerimientos no estaban completamente definidos y esta metodología es recomendada para proyectos medianos con ambientes cambiantes permitiendo analizar, diseñar y programar en pequeñas interacciones durante todo el periodo de desarrollo.

Se implementó XP junto con UWE para el desarrollo de la aplicación ya que ambas no interfieren entre sí. Cabe mencionar que XP al ser una metodología ágil para la gestión de las interacciones del proyecto puede utilizar las herramientas que expone UWE sin afectar con el avance del mismo. Con otras metodologías como SCRUM o RUP no es posible realizar una combinación ya que estas utilizan sus propios artefactos los cuales no fueron solicitados por parte de la SCPM y perderían relevancia.

Las dos metodologías mencionadas se explican en los siguientes puntos.

1.5.1. Metodología UWE (UML Web Engineering).

UWE significa UML Web Engineering o Ingeniera Web basada en UML, lo cual es una combinación de elementos propios de la Ingeniería Web y del estándar UML.

La Ingeniería Web es sin duda una rama de la Ingeniería de Software[3], la cual es una disciplina que se aplica en la construcción de aplicaciones Web. Esta Ingeniería surge debido al gran crecimiento que ha tenido la Web[4] durante varios años, lo cual es un tema de interés social y tecnológico. También hay que recalcar que "este tipo de Ingeniería difiere entre un sitio Web y un aplicativo, ya que no se especializa en la construcción de sitios Web, es decir, hace énfasis en la construcción de aplicaciones Web" (López J. , 2010). Un aspecto muy importante de este campo es el diseño y la usabilidad debido a que la accesibilidad que posee una aplicación web es muy amplia, por tal motivo varios usuarios pueden acceder con facilidad a la aplicación por medio de un navegador.

UML[5] es un lenguaje de modelado el cual es un estándar utilizado para el diseño de un sistema informático donde se utilizan herramientas que sirven para comunicar principalmente la funcionalidad, el flujo de procesos y la estructura de aplicaciones de cualquier índole. Este tema se lo analiza más detenidamente en el **Marco Teórico**.

UWE es una metodología de desarrollo de software que se enfoca en la implementación de aplicaciones Web, haciendo uso de varios artefactos propios de UML principalmente

[3] Disciplina de la Ingeniería que comprende varios aspectos en la construcción de software (Sommerville, 2006).
[4] WWW (World Wide Web) mejor conocido como internet, el cual es una inmensa red mundial de información.
[5] Lenguaje Unificado de Modelado propio de OMG (Object Managment Group) utilizado para definir la arquitectura de aplicaciones informáticas (Object Management Group, 2015).

aplicados en la Web. Cabe recalcar también que "UWE es un proceso del desarrollo para aplicaciones Web enfocado sobre el diseño sistemático, la personalización y la generación semiautomática de escenarios que guíen el proceso de desarrollo de una aplicación Web" (Quiroga, 2015). UWE hace énfasis en la presentación de la aplicación y en la interacción de la misma con el usuario, sin dejar de lado la parte de la Ingeniería, la cual brinda un enfoque más técnico en la construcción de aplicaciones Web.

UWE posee varias etapas o fases dentro del ciclo de vida de desarrollo, las cuales son enfocadas principalmente en el análisis y diseño (Quiroga, 2015).

Tabla 1.
Fases de la Metodología UWE

Análisis de requisitos	En este punto se representa los requisitos funcionales de la aplicación mediante diagramas de casos de uso[6].
Modelo de contenido	Permite plasmar los conceptos a detalle en la aplicación, los cuales pueden ser representados en un diagrama de clases[7].
Modelo de navegación	Específica los objetos que pueden ser visualizados en la aplicación por medio de estructuras de acceso como consultas, menús e índices. Este modelo se lo representa con diagramas de navegación[8].
Modelo de presentación:	Representa interfaces de usuario a través de vistas de iteración propias de UML, la cuales se las define mediante diagramas de presentación[9].
Modelo de proceso	Define los procesos que se ejecutan dentro de la aplicación junto con la interacción de los mismos. Estos procesos pueden ser representados por medio de diagramas de estructura de procesos y flujo de procesos[10].

Nota. Descripción de las fases que conforman la metodología UWE
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

[6] Describen los requisitos funcionales de un sistema informático de manera visual.
[7] Representan una vista estática de un sistema informático sin interacción con el usuario final.
[8] Definen una relación entre diferentes componentes visuales principalmente los enlaces entre páginas, menús y vistas de acceso.
[9] Mediante interfaces de usuario se define la estructura visual de la aplicación, por lo cual estas se representan como vistas abstractas propias de UML.
[10] Establece los subprocesos y la interacción entre los mismos, los cuales forman parte de un proceso principal; como también se toma en cuenta el flujo del mismo desde su inicio hasta su fin.

Durante la etapa de análisis y diseño UWE define el uso de fases conformadas por modelos, las cuales se describen en la Tabla 1.

Para la construcción se realiza la codificación del sistema como tal, lo cual consiste en traducir en líneas de código los respectivos modelos de análisis y diseño mencionados anteriormente.

Por ultimo en la etapa de pruebas se cerciora que el sistema cumpla con lo esperado, así como el correcto rendimiento del mismo. Las pruebas proporcionan calidad en el producto a entregar, garantizando que el mismo es apto para ser implantado en producción, y utilizado por parte del usuario final.

Las herramientas que se utilizarán durante el proceso de desarrollo de software de acuerdo a la metodología UWE se explican a continuación.

1.5.1.1. Diagrama de casos de uso.

Este tipo de diagramas representarán una vista general del sistema junto con la interacción del usuario. Dicho en otras palabras, los diagramas de casos de uso representan de forma visual el comportamiento del sistema informático. Cada diagrama debe poseer un caso de uso donde se explica a detalle la funcionalidad del sistema como tal.

1.5.1.2. Diagrama de clases.

Este tipo de diagramas son estáticos debido a que describen la estructura del sistema sin tomar en cuenta la participación del usuario. Las clases descritas poseen métodos y atributos que formarán parte de las características y tareas a realizarse de cada clase.

1.5.1.3. Diagrama de navegación.

Este tipo de diagrama permite representar los objetos y la estructura de navegación de la aplicación. Es de gran utilidad para la construcción de aplicaciones web, ya que describe el enlace que existe entre las distintas páginas que pueden existir dentro del contenido de una aplicación.

1.5.1.4. Diagrama de presentación.

Proporciona una visión abstracta de una interfaz en la cual el usuario interactúa con la aplicación. Este diagrama estructura los objetos que se muestran en pantalla como texto, imágenes, enlaces, entre otros.

1.5.1.5. Diagrama de estructura de proceso.

Puede ser representado por un diagrama de clases donde se describen las relaciones entre las diferentes clases principales de la aplicación.

1.5.1.6. Diagrama de flujo de proceso.

Este tipo de diagrama puede ser relejado por un diagrama de actividades donde se incluyen actividades, actores responsables de las mismas y elementos de flujo de control.

1.5.2. Metodología XP.

Para gestionar el proyecto de desarrollo del sistema de gestión vehicular para la SCPM se ha optado por utilizar Extreme Programing (XP), que es una metodología ágil, debido a que esta se fundamenta en lo que son buenas prácticas para el cumplimiento de los requerimientos del cliente y su completa satisfacción en un tiempo relativamente corto.

En la actualidad el reciente crecimiento de las organizaciones y del mercado que lo rodea hace indispensable para una organización pública o privada mantenerse a la vanguardia de las tecnologías, esto para mantener una ventaja competitiva o tan sola para continuar ofreciendo sus servicios o productos. Es por esto que se requiere que los proyectos de desarrollo vayan a la par de este crecimiento y se puedan adaptar a cambios repentinos, esto genera la necesidad de una metodología de trabajo que pueda lidiar con los cambios, asegurar la calidad del producto final manteniendo el presupuesto del proyecto acordado con el cliente.

> XP está encaminada a la programación de software, la comunicación clara, y un equipo de trabajo que desarrolle el software con el menor coste y defectos posibles, así como con la mayor productividad y retorno de la inversión. Además, el cliente adquiere más control sobre el trabajo y un mayor protagonismo durante el desarrollo del producto final. Promueve la simplicidad en el diseño del proyecto así como las iteraciones continuas entre el equipo de trabajo y el cliente y la reutilización del código desarrollado (López F. , 2015).

En XP se trabaja estrechamente con el cliente, se hace pequeñas iteraciones cada dos semanas, donde no existe más documentación que el historial de usuarios en donde se registra los cambios requeridos por el cliente[11]. Cada versión contiene las modificaciones necesarias según como el cliente vaya retroalimentando al proyecto, por eso es necesaria la disponibilidad del cliente durante el desarrollo (Ganán, 2010).

[11] Guía Comparativa de Metodologías Ágiles

Las iteraciones en la metodología XP son cortas, ya que entre más rápido se le entreguen los desarrollos al cliente mucha más retroalimentación se va a obtener, lo cual significa una mejor calidad del producto a largo plazo[12]. Existe un nivel de análisis inicial orientado a programar las iteraciones de desarrollo y cada iteración incluye, diseño, codificación y pruebas.

1.6. Marco institucional

1.6.1. La Superintendencia de Control del Poder de Mercado (SCPM).

La SCPM es una entidad de control, con personalidad jurídica de derecho público, la cual posee autonomía administrativa, financiera, presupuestaria y organizativa. También tiene como finalidad asegurar que se cumplan las normas de eficacia y eficiencia que rigen el mercado del Ecuador. Cabe mencionar que tiene la potestad de sancionar la competencia desleal, acuerdos anticompetitivos, y prácticas monopólicas con el fin de crear un sistema económico gobernable y equitativo. Todo lo mencionado consta dentro de la Ley Orgánica de Regulación y Control del Poder de Mercado[13]; así como del control y regulación de las operaciones de concentración económica, de acuerdo con lo establecido en la Ley de la materia (Páez, 2012).

El concepto de Superintendencia dentro del Estado Ecuatoriano es de vital conocimiento de acuerdo a la siguiente información.

> Son organismos técnicos de vigilancia, auditoría, intervención y control de las actividades económicas, sociales y ambientales, y de los servicios que

[12] Ciclo de Vida de Desarrollo de Software

[13] Ley de protección de consumidores, empresarios y productores, de los abusos derivados de la alta concentración económica y las prácticas monopólicas (Páez 2012).

> prestan las entidades públicas y privadas, con el propósito de que estas actividades y servicios se sujeten al ordenamiento jurídico y atiendan al interés general (Páez, 2012).

La Superintendencia de Control del Poder de Mercado se creó mediante la Ley Orgánica de Regulación y Control del Poder de Mercado, publicada en el Registro Oficial Suplemento No. 555 de 13 de octubre de 2011, donde se detalla que "es un organismo de control y sanción con personalidad jurídica, patrimonio propio y autonomía administrativa, presupuestaria y organizativa" (Páez, 2012).

Los objetivos a alcanzar más relevantes para de la Superintendencia de Control del Poder de Mercado de acuerdo a la RESOLUCIÓN No. SCPM-DS-2012-001, se afirma lo siguiente.

> Investigar, corregir y de ser el caso sancionar los abusos del poder de mercado. Controlar y regular la concentración económica, a efecto de evitar prácticas monopólicas y oligopólicas contrarias al interés económico general, buscando la eficiencia en los mercados a favor de los operadores económicos y de manera particular el bienestar de los consumidores y usuarios. Garantizar a través del control el derecho a desarrollar actividades económicas y la libre concurrencia de los operadores económicos al mercado (Páez, 2012).

Es de gran interés conocer la institución donde se implementa el Sistema de Gestión Vehicular debido a que se amplía de manera significativa el contexto del caso de estudio. También es de gran relevancia conocer las teorías aplicadas en el desarrollo de este

proyecto con el objetivo sustentar el trabajo realizado por parte de los autores, las cuales se pueden apreciar en el siguiente apartado.

1.7. Marco teórico

El Software de Gestión Vehicular para instituciones del Estado involucra elementos teóricos básicos. Para esta investigación las líneas teóricas relevantes están relacionadas con:

- Programación Orientada a Objetos
- Lenguaje Unificado de Modelado
- Modelo Vista Controlador
- Bases de Datos Relacionales
- Gestión de Activos

1.7.1. Programación Orientada a Objetos.

Adentrándose en la definición de la Programación Orientada a Objetos, su historia, características, etc.; se dice que la misma es un paradigma de programación[14].

Durante el transcurso de la historia del desarrollo de software, se puede mencionar que existieron varios modelos, los cuales fueron la base de los primeros lenguajes de programación de alto nivel; dos de los principales son la Programación Orientada a Procesos o *Process Oriented Programming* (POP) y la Programación Orientada a Datos o *Data Oriented Programming* (DOP), donde cada uno planteo su propio enfoque. La POP brindaba estructuras enfocadas directamente a los procesos, donde el manejo de

[14] Escritura de código fuente propio de un sistema informático.

información no fue tan relevante, lo cual atrajo problemas con el pasar de los años; algunos de los lenguajes de programación que acogieron este modelo fueron C y Pascal. La DOP propone mayor atención en el manejo de los datos, donde nacieron gestores que se enfocaban netamente en la administración de datos de manera integrada, de igual forma surgieron novedades con respecto a los procesos ya que en este modelo fueron sencillos, y en algunos casos inexistentes (Álvarez Sánchez, 2015).

Es en este punto donde nace la Programación Orientada a Objetos u *Object Oriented Programming* (OOP) donde se acoge la POP y la DOP, consiguiendo un equilibrio entre ellas donde se aborda la relevancia en el trato de los procesos y de los datos, evitando los problemas de acoplamiento entre ambos paradigmas tal como se afirma que "la programación Orientada a objetos (POO) es una forma especial de programar, más cercana a como expresaríamos las cosas en la vida real que otros tipos de programación" (Álvarez, Qué es la programacion orientada a objetos, 2001).

En si la OOP es un paradigma que representa una abstracción de la vida real, lo cual da un alcance a la solución informática de un problema, donde es representada en código orientado a objetos, donde un objeto puede poseer propiedades, atributos, métodos y demás. La OOP es una nueva forma de pensar. Tambien hay que mencionar que "la programación orientada a objetos es una filosofía, un modelo de programación, con su teoría y su metodología" (Izquierdo, 2014). La OOP plantea una forma distinta de enfocar los problemas, haciendo énfasis en una ideología enfocada más a lo práctico, incorporando aspectos técnicos que son propios de este paradigma como el encapsulamiento, herencia, polimorfismo y abstracción.

La OOP propone un cambio en la concepción del desarrollo de software donde se introduce un nivel de abstracción, mejorando el código fuente de una aplicación o solución informática. Los conceptos de clase[15] y objeto[16] proporcionan cierta abstracción del mundo real enfocado en el sustantivo (sujeto u objeto) y no en el verbo (acción). La información representada en datos se encapsula dentro de una clase, donde el acceso se lo controla independientemente de la representación final de los mismos, como también el mantenimiento es reducido en cuanto a complejidad, permitiendo una evolución acelerada en el desarrollo. Por medio de las propiedades de la OOP se consigue simplificar la construcción de sistemas informáticos optimizando a gran medida el código fuente (Prieto Ramos, Sosa Sánchez, & Rodríguez Echeverría, 2004). La Figura 2 muestra las propiedades de la OOP las cuales se describen a continuación.

1.7.1.1. Abstracción.

Esta propiedad se encarga de ocultar los detalles de un objeto, mostrando solamente lo más relevante. Se menciona tambien que "es una generalización conceptual de un determinado conjunto de objetos y de sus atributos y propiedades, dejando en un segundo término los detalles concretos de cada objeto" (Devis Botella, 2010). Las propiedades de un objeto pueden ser abstraídos, permitiendo visualizar solamente las características más relevantes de un conjunto de comportamientos que se ejecutan por debajo de un proceso.

[15] Representa a un objeto específico dentro del campo de la programación.
[16] Entidad de la vida real que posee ciertas características.

1.7.1.2. Encapsulación.

Los datos y procedimientos que se manejan de forma integrada y se encapsulan dentro de un objeto cuyo acceso se lo maneja de manera controlada.

1.7.1.3. Polimorfismo.

Permite descomponer un sistema en un conjunto de módulos[17] independientes que pueden ser consolidados al final, permitiendo acceder a distintas funciones a través de una misma interfaz[18].

1.7.1.4. Herencia.

Incorpora las propiedades de un objeto primario hacia uno o varios que vendrían a ser sus predecesores, los cuales pueden utilizar dichas propiedades de manera independiente. La herencia es un aspecto importante en la OOP, ya que permite reutilizar el código, reduciendo el tiempo de desarrollo en gran medida.

La OOP, como se ha mencionado anteriormente, es una filosofía o paradigma de gran uso en el desarrollo de aplicaciones, la cual ha sido implementada y soportada por varios lenguajes de programación de alto nivel como Java, C#, PHP, Pyton, Perl, Ruby, y muchos más[19]; donde el desarrollador puede apegarse a este modelo sin restricciones, permitiendo incorporar a la vez el conocimiento de desarrollo y de la OOP.

---

[17] Subgrupo o división de lo general a lo específico utilizado en la descomposición de sistemas informáticos.
[18] Conjunto de métodos que posee un objeto dentro de la OOP.
[19] Varios lenguajes de programación que son utilizados en gran medida en la actualidad.

El análisis y diseño dentro del desarrollo de software es un punto crucial, donde se utilizan de igual forma métodos y modelos, uno de ellos se lo explica a continuación.

1.7.2. Lenguaje Unificado de Modelado.

El Lenguaje Unificado de Modelado o *Unifed Modeling Language* (UML), es un lenguaje que permite representar de forma gráfica el diseño de un sistema informático como tal. Para comprender de mejor forma este lenguaje primero se debe analizar su historia con una breve reseña.

El lenguaje de símbolos e imágenes se ha llevado a cabo desde tiempos remotos desde hace 75.000 años (Kimmel, 2008). En varias culturas como egipcias, mayas, indias, entre otras, se representaban mediante símbolos una serie de sucesos, mensajes, prevenciones, etc.; lo que da a concluir sin duda que dichos símbolos fueron un lenguaje de comunicación. No se analizará a fondo las culturas y lenguajes que han transcurrido durante la historia del ser humano, debido al contexto de investigación donde se está realizando este trabajo, pero es de gran utilidad tomar en cuenta este tema debido a que en los tiempos modernos se aplica nuevamente la representación de imágenes y símbolos como una forma de comunicar algo, y ni más ni menos que en el campo del desarrollo de sistemas informáticos, que resulta de cierta forma complejo tanto en el análisis, la gestión, como también en la construcción de los mismos.

UML como cualquier lenguaje cotidiano de la vida real fue creado por la necesidad de representar el software previo a su construcción. Debido a que la Programación Orientada a Objetos OOP tuvo su auge entre 1970 y 1980, existió una gran incógnita que fue la adecuación de técnicas de diseño basadas en este paradigma. Existieron métodos de diseño

bastante similares pero no se adecuaban entre si debido a la existencia de varias diferencias entre los mismos. Un estándar era necesario para impedir la incompatibilidad entre analistas y reducir los conflictos que existían en aquella época (Roig Vásquez, 1997).

A finales de 1980 y principios de 1990, Ivar Jacobson, James Rumbaugh y Grady Booch, investigadores de ciencias informáticas[20], dieron el primer paso en la definición de ciertos elementos de análisis y diseño de software; donde cada investigador uso símbolos independientes junto con reglas diferentes. A mediados de la década de 1990, se unieron los elementos de modelado de Rumbaugh conocido como Object Modeling Technique o Técnica de Modelado de Objetos, el de Booch conocido como método de Booch, y también Jacobson con su modelo Objectory and Use Cases. Es aquí cuando se presenció el nacimiento del Lenguaje Unificado de Modelado, mejor conocido como UML. En 1997 se estandarizo UML con su primera versión, y finalmente en el 2004 surgió su segunda versión, la cual se utiliza hasta la actualidad (Kimmel, 2008).

Después de analizar muy brevemente el recorrido que ha tenido UML, se lo puede describir de una forma más amplia. Existen varias definiciones de varios autores de libros, de los cuales se mencionan los siguientes.

Hay que mencionar que "UML es un lenguaje tanto como Pascal, C# (Csharp), el alemán, el inglés y el latín; y el UML posiblemente es uno de los lenguajes más recientes inventados por la humanidad" (Kimmel, 2008).

[20] Rama de las ciencias exactas que estudia el hardware, el software y las comunicaciones informáticas.

Tambien se dice que “The UML is a standard diagramming notation. As useful as it is to learn notation, there are more critical object-oriented things to learn; specialy how to think in objects-how to design object-oriented systems” (Larman, 2001).[21]

Roig afirma lo siguiente “Decimos que el UML es un lenguaje de modelado y no un método” (Roig Vásquez, 1997).

Entonces se concluye que UML es un lenguaje estandarizado el cual utiliza nomenclaturas y símbolos unificados que permiten representar principalmente el funcionamiento, la interacción y el flujo de tareas propios de un sistema informático. A diferencia de los métodos que consisten en procesos y funciones, UML comunica de manera visual al usuario una esquematización de un posible producto a desarrollar en un tiempo determinado, donde un cambio pudiera ocurrir sin poner en desventaja a una implementación de software avanzada.

El diseño de software es un tema delicado, pero de poco interés en varios campos de desarrollo de software. Un claro ejemplo de aquello es el siguiente. En el año 1994 se llevó a cabo un estudio por parte de The Standish Gruop[22] donde se analizó 50.000 proyectos, de los cuales el 16% fueron exitosos, y de ese porcentaje solo el 61% cubrió toda la funcionalidad esperada. Los proyectos restantes tuvieron varias complicaciones como retrasos, costos no previstos, y cancelación total de los mismos. En la actualidad este margen de éxito se ha elevado pero no de forma considerable (Álvarez Sánchez, 2015).

[21] Notación de símbolos expresados en diagramas que se adecua en la Programación Orientada a Objetos.
[22] Organización de consultoría e investigación que se centra en el rendimiento de proyectos de software a nivel mundial (The Standish Group, 2015).

Entonces se plantea una incógnita: ¿Cómo UML puede contribuir con el éxito de los proyectos de desarrollo de software?

UML permite modelar un sistema informático de cualquier índole con una gran variedad de diagramas, donde se puede manifestar distintas tareas a realizarse, esquematización de procesos y flujo de información. Si bien es cierto, todo lo mencionado se lo puede llevar a cabo de manera textual y llegar a acuerdos con el usuario final, pero es en este punto donde ocurre uno de los principales problemas.

Muchas veces el usuario que utilizará el software a desarrollarse no entiende de conceptos en temas como Programación Orientada a Objetos, Modelo Vista Controlador, UML, etc.; por lo cual es complicado para un desarrollador explicar la parte técnica, pero no la parte funcional de un sistema, donde el usuario puede aportar en gran medida. Cuando el usuario aparentemente está claro en todo lo requiere en la etapa de análisis de requerimientos, y posteriormente el desarrollo se encuentra en una etapa avanzada donde el usuario puede apreciar el software, pues es aquí donde un cambio por parte del usuario influiría mucho en tiempo, costo y calidad, dando cabida a un gran problema para que un proyecto de desarrollo no cumpla con lo esperado y fracase.

En cambio con UML se puede representar el software de forma gráfica, muchas veces didáctica, dependiendo del nivel de complejidad, donde el usuario puede opinar y aportar con la solución planteada, llenando vacíos y cubriendo funcionalidad no esperada por parte de los arquitectos y analistas de software.

Con el uso de UML en proyectos de software se busca mitigar los posibles problemas que pueden surgir en el análisis y levantamiento de información, tomando medidas correctivas

en el diseño directamente, tratando de agilitar el proceso de construcción de la mayoría de lo posible, manejando los cambios que puedan ocurrir de una manera adecuada.

UML facilita la documentación formal en el desarrollo de un proyecto de software, a su vez delimita el alcance real del proyecto con buenas practicas por medio de estándares, y también permite entregar el software en el tiempo previsto junto con todas las características esperadas por el usuario final.

1.7.3. Modelo Vista Controlador (MVC).

El mundo del desarrollo del software está sometido continuamente a cambios, como que de un proceso evolutivo se tratase. Actualmente, se puede ver cómo ha ido cambiando desde el código de máquina y lenguaje ensamblador a través de conceptos como programación estructurada, programación orientada a objetos y ahora con el creciente desarrollo de aplicaciones web y móviles (Prieto Ramos, Sosa Sánchez, & Rodríguez Echeverría, 2004). Tambien se menciona que “MVC es un patrón de diseño que considera dividir una aplicación en tres módulos claramente identificables y con funcionalidad bien definida: El Modelo, las Vistas y el Controlador” (Álvarez, Qué es MVC, 2014).

1.7.3.1. El modelo.

Esta capa es la que se encarga de la manipulación de los datos, es por este motivo que se implementan las funciones necesarias para acceder a la información y manipularla. Es en esta capa donde los modelos tendrán todos los métodos que permiten acceder a la base de datos y realizarán la transaccionalidad correspondiente.

Cabe mencionar que cuando se trabaja con MCV lo habitual es utilizar otras librerías adicionales en el IDE[23] de desarrollo, con el fin de permitir trabajar con abstracción de bases de datos y persistencia en objetos. Por ello, en vez de usar directamente sentencias SQL[24], que suelen depender del motor de base de datos con el que se esté trabajando, se utiliza un dialecto que permita la abstracción de datos basado en clases y objetos (Álvarez, Qué es MVC, 2014).

1.7.3.2. Las vistas.

Las vistas, contienen el código escrito que al ser compilado generará la interfaz con la que interactuará el usuario final, en nuestro caso una página web. En las vistas nada más se aprecia el código HTML y ASP que nos permiten mostrar la salida. Cabe mencionar que "en la vista generalmente se trabaja con los datos, sin embargo, no se realiza un acceso directo a éstos. Las vistas requerirán los datos a los modelos y estos generarán una salida, que será desplegada al usuario" (Álvarez, Qué es MVC, 2014).

1.7.3.3. El controlador.

El controlador realiza la función de enlazar las vistas y los modelos. El controlador es un objeto que se encarga de dirigir el flujo del control de la aplicación debido a mensajes externos, como datos introducidos por el usuario u opciones del menú seleccionadas por él. A partir de estos mensajes, el controlador se encarga de modificar el modelo o de abrir

[23] Interfaz de usuario que permite desarrollar aplicaciones en un lenguaje específico.
[24] Motor de base de datos desarrollado por Microsoft.

y cerrar vistas. El controlador tiene acceso al modelo y a las vistas, pero las vistas y el modelo no conocen de la existencia del controlador (Bascón Pantoja, 2004).

El desarrollo del Sistema de Gestión Vehicular se lo realiza utilizando MVC, ya que permitirá separar la aplicación en diferentes capas permitiendo una mejor escalabilidad y mantenimiento.

1.7.4. Bases de datos relacionales.

Para entender las bases de datos relacionales es indispensable empezar definiendo que una base de datos es un conjunto de información almacenada de diferente temática que permiten ser manipulados, recuperados y categorizados de diferentes maneras dependiendo de la necesidad para que puedan ser utilizados por diferentes usuarios o aplicaciones.

La forma en la que se almacena la información ha ido cambiando continuamente desde el uso de registros físicos como son los libros, tarjetas perforadas, cintas magnéticas hasta lo que se tiene hoy en día como grandes almacenes de datos que contienen información de todo el mundo (Costal, 2014).

Uno de estos avances ha sido las bases de datos relacionales ya que surgieron a la par de la programación orientada a objetos porque permiten representar las entidades del mundo real (Universidad del Azuay, 2015).

Edgar Frank Codd definió las bases del modelo relacional a finales de los 60. Trabajaba para IBM la cual tardó un poco en implementar sus bases. Pocos años después el modelo

se empezó a implementar cada vez más, hasta ser el modelo de bases de datos más popular (Sánchez, 2004).

Tabla 2.
Objetivos del modelo de Codd

Independencia física	La forma de almacenar los datos, no debe influir en su manipulación lógica.
Independencia lógica	Las aplicaciones que utilizan la base de datos no deben ser modificadas por que se modifiquen elementos de la base de datos.
Flexibilidad	El gestor brinda ambientes distintos de acuerdo a cada tipo de sistema.
Uniformidad	La base de datos y la información se encuentran sincronizadas sin importar las conexiones que posean.

Nota. Descripción de los objetivos del modelo de Codd
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El modelo de base de datos se puede resumir en lo siguiente.

> El modelo relacional intenta representar con una tabla a un tipo de objetos de la vida real, como puedan ser Empleados, Clientes, etc., e incluso considera las relaciones entre estos objetos como objetos en sí mismos. Para ello, designa cada tipo de objetos por un conjunto de atributos que son los que darán la «forma particular» a cada objeto (Costal, 2014).

Se escogió este modelo de base de datos ya que permite una abstracción de la realidad y se acopla con el paradigma de programación orientada a objetos. Siendo el tipo de base de datos más utilizado actualmente y el más difundido resulta muy conveniente el utilizarlo además de que uno de los requerimientos por parte de la SCPM es que se almacene la información en SQL Server, el cual soporta bases de datos relacionales.

1.7.5. Gestión de activos.

Actualmente se puede apreciar una inexperiencia en el campo empresarial con respecto a la gestión de activos fijos y aún más en el caso de las instituciones públicas. Por este motivo es necesario recalcar lo siguiente.

> Los activos fijos tienen una importancia en los negocios, ya que si se posee la información correcta de los mismos, se puede conocer el pasado, vigilar el presente y programar el futuro de las inversiones del negocio, tanto a corto como a largo plazo. Para lograr esto, es necesario considerar las necesidades propias de la empresa y evaluar las políticas con las que cuentan para la planificación o gestión de sus activos (Aguilar, 2015).

Contar con una buena gestión administrativa de los activos fijos, permite que los mismos se conviertan en una de las mejores inversiones de la empresa, impactando en los siguientes aspectos.

Tabla 3.
Administración de activos

Financiero	Debido a que se debe reflejar en las cuentas del estado de resultado el valor de las amortizaciones, ingresos o gastos que estos generen.
Fiscal	Permite generar una oportunidad de optimización fiscal, ya que ayuda a tomar los criterios más convenientes según las normativas impuestas.
Costos	Permite reducir el valor de inversiones (terrenos, edificios, maquinarias, entre otros) haciendo cargos que afecten al estado de resultados a través del tiempo.

Nota. Descripción de aspectos importantes en la administración de activos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Es necesario mencionar también que “la problemática de los activos físicos del Estado Ecuatoriano debe ser planteada desde distintos puntos de vista, en razón de la importancia

económica, la complejidad y la disparidad de situaciones que presenta la administración actual de estos activos" (Reinberg, 1986).

En el acuerdo ministerial 130[25] se expide un instructivo para el control de activos donde se describe las medidas que se deben tomar en cuenta, desde la toma física de inventarios y la forma de hacerlos. Aunque el acuerdo ministerial haya sido expedido en el año de 1986 aún está vigente. Es por este motivo que es necesario llevar un control exhaustivo de los activos del Estado (Jefatura de Gabinete de Ministros, 2010).

De acuerdo a lo mencionado anteriormente, el proyecto a desarrollar ayudará a la gestión de activos en cuanto al control de vehículos de la SCPM, ya que permitirá un control más eficiente en la toma de decisiones para la optimización de recursos.

[25] Documento normativo para el control y uso de activos fijos pertenecientes a instituciones públicas del Estado. Expedido en el año 2010 por la Jefatura de Gabinete de Ministros bajo la coordinación de Mónica Zorrilla Subsecretaria de Gestión y Empleo Público.

CAPÍTULO 2

ANÁLISIS Y DISEÑO

En este capítulo se trata a fondo las fases de análisis y diseño del Sistema de Gestión de Transporte para la Superintendencia de Control del Poder de Mercado.

UWE es la metodología de desarrollo de software que se está utilizando para el presente trabajo, la cual indica que se debe implementar ciertos modelos durante el transcurso del desarrollo de análisis y diseño.

El análisis de requisitos de acuerdo con UWE, es la etapa donde se cubre toda la fase de análisis dentro del desarrollo de software. Los modelos de contenido, navegación, presentación y proceso; acaparan en cambio la fase diseño de software para su posterior codificación.

El análisis de requisitos es una de las partes más relevantes en la construcción de un sistema informático ya que permite identificar la problemática y los procesos que requieren ser optimizados, los cuales ayudan a definir las partes de una solución integrada, tomando en cuenta también el criterio del usuario final que será quien utilice la aplicación.

2.1. Análisis de requisitos

Los requisitos de un sistema son proporcionados por los involucrados que entienden la problemática, como también la mayor parte de los procesos que forman parte de la misma, en este caso suelen ser el usuario final o el cliente. En esta etapa la interacción con los involucrados es crucial debido a que se debe recopilar la mayor parte de información con el objetivo de identificar las variables y puntos clave que identifican la raíz del problema

o necesidad, para su posterior optimización. Una vez identificado lo que se desea mejorar mediante una solución informática, se debe documentar lo que el sistema va a realizar en cuanto a funcionalidad. Por parte de la SCPM se obtuvo un acta de requisitos funcionales y no funcionales, la cual se encuentra en el **Anexo 1**. A continuación se explica un breve resumen los mismos.

- **Requisitos funcionales:** describen lo que debe realizar la aplicación como tal. Estos requisitos se ven reflejados por medio de casos de uso.
- **Requisitos no funcionales:** se hace hincapié en el rendimiento, fiabilidad, seguridad y mantenibilidad que debe poseer la aplicación.

El análisis de requisitos incorpora casos de uso, los cuales deben ser descritos a detalle junto con un diagrama de casos de uso el cual interpreta el caso de uso de forma gráfica.

Los casos de uso que son utilizados para describir la funcionalidad del Sistema de Gestión de Transporte son los siguientes.

- Caso de uso solicitud de movilización
- Caso de uso registro de movilización de vehículos
- Caso de uso gestión de vehículos

A continuación se describe cada caso de uso con su respectiva explicación a detalle y junto con su diagrama.

2.1.1. Caso de uso solicitud de movilización.

2.1.1.1. Descripción de caso de uso solicitud de movilización.

Nombre: Solicitud de movilización.

Actores: Empleado (Iniciador), Administrador Financiero y Administrador de Movilización.

Postcondición: Validar datos de entrada.

Propósito: Registrar una solicitud de movilización para posteriormente ser autorizada o negada.

Resumen: Un Empleado ingresa información en un formulario de movilización, el cual se almacena en la bandeja de solicitudes pendientes del Administrador de Movilización o del Administrador Financiero. El Administrador Financiero preautoriza o niega solicitudes pendientes con comisión de servicio, En el caso de que se preautorice una solicitud con comisión de servicio, el Administrador Financiero puede adjuntar un archivo opcional de preautorización. El Administrador de Movilización autoriza o niega solicitudes pendientes sin comisión de servicio, y solicitudes con comisión de servicio que han sido preautorizadas por el Administrador Financiero. En el caso de autorizar la solicitud se asignará un vehículo y un conductor para su posterior movilización, y en el caso de solicitudes con comisión de servicio, el Administrador de Movilización puede adjuntar un archivo opcional de autorización.

En la Tabla 4 se describe el curso normal que tomará el caso de uso solicitud de movilización durante el proceso de generación de solicitud por parte del Empleado.

Tabla 4.
Curso normal de una solicitud de movilización

Curso normal de caso de uso solicitud de movilización			
1	El empleado llena los datos del formulario de movilización. (Solicitud normal sin comisión de servicio)	2	Validar datos del formulario

2	El Empleado envía el formulario hacia la bandeja del Administrador de Movilización.	3	Enviar notificación de nueva solicitud hacia el Administrador de Movilización.
4	El Administrador de Movilización autoriza la solicitud del Empleado asignando un vehículo y conductor respectivamente disponible. (Solicitud normal sin comisión de servicio)	5	Enviar notificación de autorización de solicitud hacia el Empleado.

Nota. Descripción del curso normal del caso de uso solicitud de movilización
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Los cursos alternos que pueden suscitarse en un caso de uso se los debe detallar de igual manera, tal como se muestra en la Tabla 5, donde existe un curso alterno en el orden 3 dentro del curso normal del caso de uso solicitud de movilización.

Tabla 5.
Cursos alternos de una solicitud de movilización

Cursos alternos de caso de uso solicitud de movilización	
1.1	El empleado llena los datos del formulario de movilización especificando que la solicitud es de comisión de servicio.
1.2	El Empleado envía el formulario hacia la bandeja del Administrador Financiero. Enviar notificación de nueva solicitud con comisión de servicio hacia el Administrador Financiero.
1.3	El Administrador Financiero preautoriza la solicitud del Empleado adjuntando o no un archivo opcional de preautorización. Enviar notificación de preautorización de solicitud hacia el Empleado.
1.3.1	El Administrador Financiero niega una la solicitud con comisión de servicio. Se envía una notificación de negación de movilización hacia el Empleado.
4.1	El Administrador de Movilización autoriza la solicitud con comisión de servicio del Empleado asignando un vehículo y conductor respectivamente disponible, y ajuntando o no un archivo opcional de autorización. Enviar notificación de autorización de solicitud hacia el Empleado.
4.2	El Administrador de Movilización niega una la solicitud. Se envía una notificación de negación de movilización hacia el Empleado.

Nota. Descripción de los cursos alternos del caso de uso solicitud de movilización
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Diagrama de caso de uso de solicitud de movilización

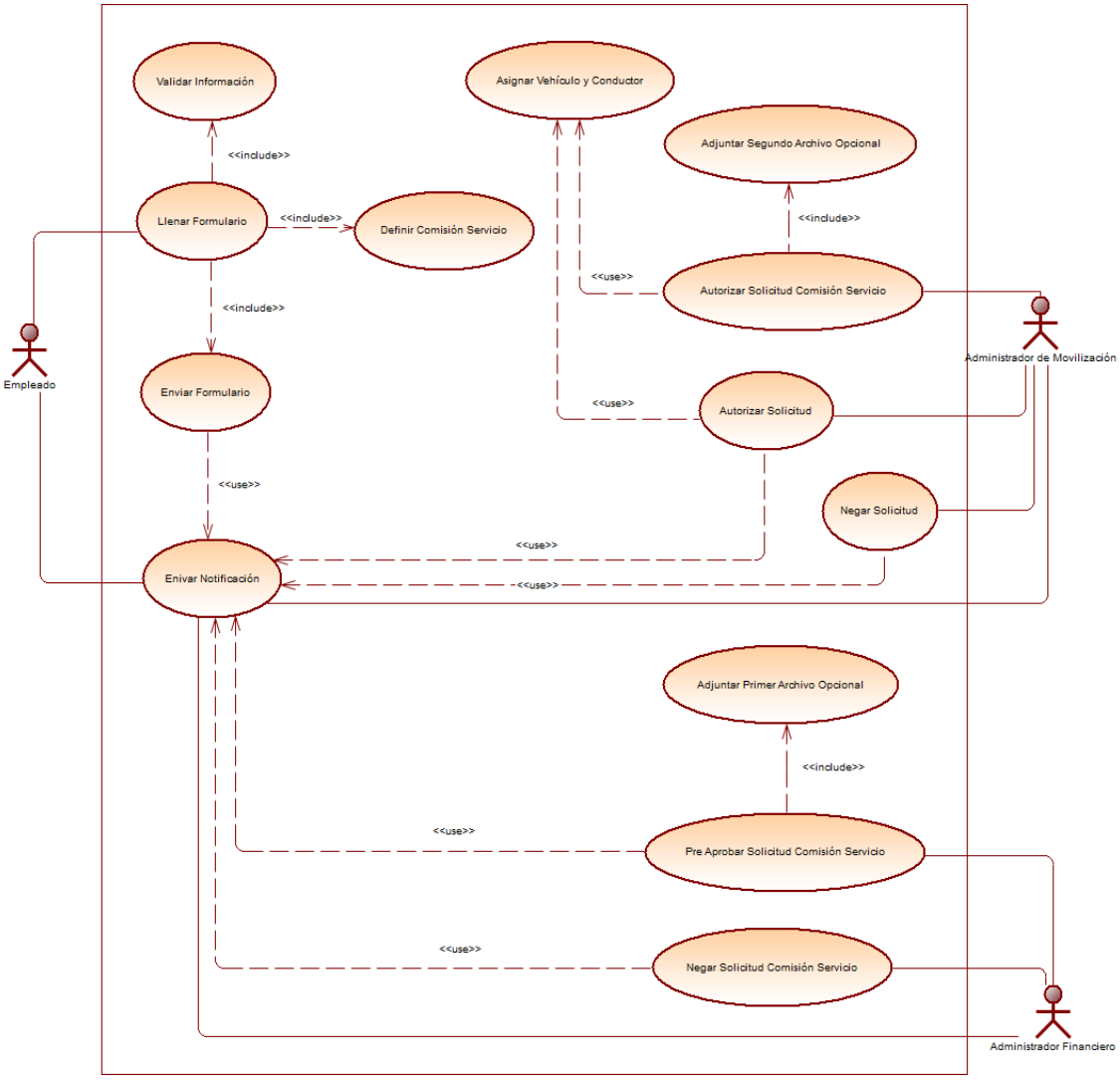


Figura 1. Diagrama de caso de uso representando el curso de solicitudes de movilización
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.1.2. Caso de uso registro de movilización de vehículos.

2.1.2.1. Descripción de caso de uso registro de movilización de vehículos.

Nombre: Registro de Movilización de Vehículos.

Actores: Personal de Seguridad (Iniciador).

Propósito: Registrar vehículos que entran y salen de la SCPM.

Resumen: El Personal de Seguridad de la SCPM se encarga de registrar el kilometraje de ingreso y de salida de los vehículos usados por la SCPM, los cuales se pueden encontrar autorizados para movilizarse, como también pueden no poseer autorización. El Personal de Seguridad también registra vehículos ajenos a la institución que ingresan y salen de la misma. Por último el Personal de Seguridad se encarga de registrar el arribo de los vehículos de transporte diario de los empleados de la institución.

En la Tabla 6 se describe el curso normal que tomará el caso de uso registro de movilización de vehículos durante el proceso de generación de un nuevo registro de vehículo por parte del Personal de Seguridad.

Tabla 6.
Curso normal del registro de movilización de vehículos

Curso normal de caso de uso registro de movilización de vehículos	
1	El Personal de Seguridad registra un vehículo autorizado de uso de la SCPM que ingresa o sale de la institución.

Nota. Descripción del curso normal del caso de uso registro de movilización de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Los cursos alternos que pueden ocurrir dentro del caso de uso registro de movilización de vehículos, se describen en la Tabla 7 donde se puede apreciar 3 posibles alternativas que parten del orden 1 en el curso normal descrito en la Tabla 10.

Tabla 7.
Cursos alternos del registro de movilización de vehículos

Cursos alternos de caso de uso registro de movilización de vehículos	
1.1	El Personal de Seguridad registra un vehículo no autorizado de uso de la SCPM que ingresa o sale de la institución.
1.2	El Personal de Seguridad registra un vehículo externo que ingresa o sale de la institución.
1.3	El Personal de Seguridad registra el arribo de un vehículo de transporte diario de los Empleados.

Nota. Descripción de los cursos alternos del caso de uso registro de movilización de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Diagrama de caso de uso de registro de movilización de vehículos

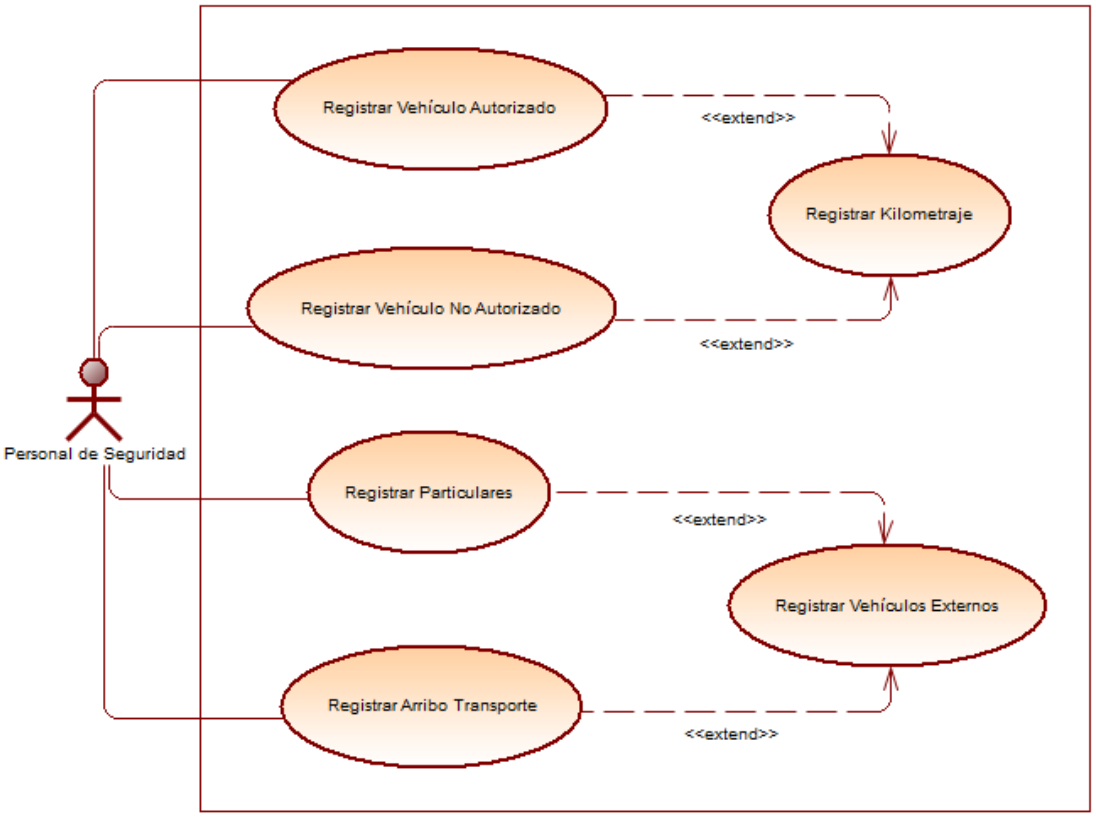


Figura 2. Diagrama de caso de uso representando el curso de registro de movilización de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.1.3. Caso de uso gestión de vehículos.

2.1.3.1. Descripción de caso de uso gestión de vehículos.

Nombre: Gestión de Vehículos.

Actores: Súper Administrador (Iniciador).

Propósito: Gestionar la información de vehículos en el Gestor de Transporte.

Resumen: El Súper Administrador del Gestor de Transporte es el encargado de crear y modificar los registros de vehículos en el Gestor de Transporte, los cuales pueden variar

de acuerdo al Estado del vehículo y otras características como el tipo de vehículo, tipo de propietario y marcas.

Diagrama de caso uso de gestión de vehículos

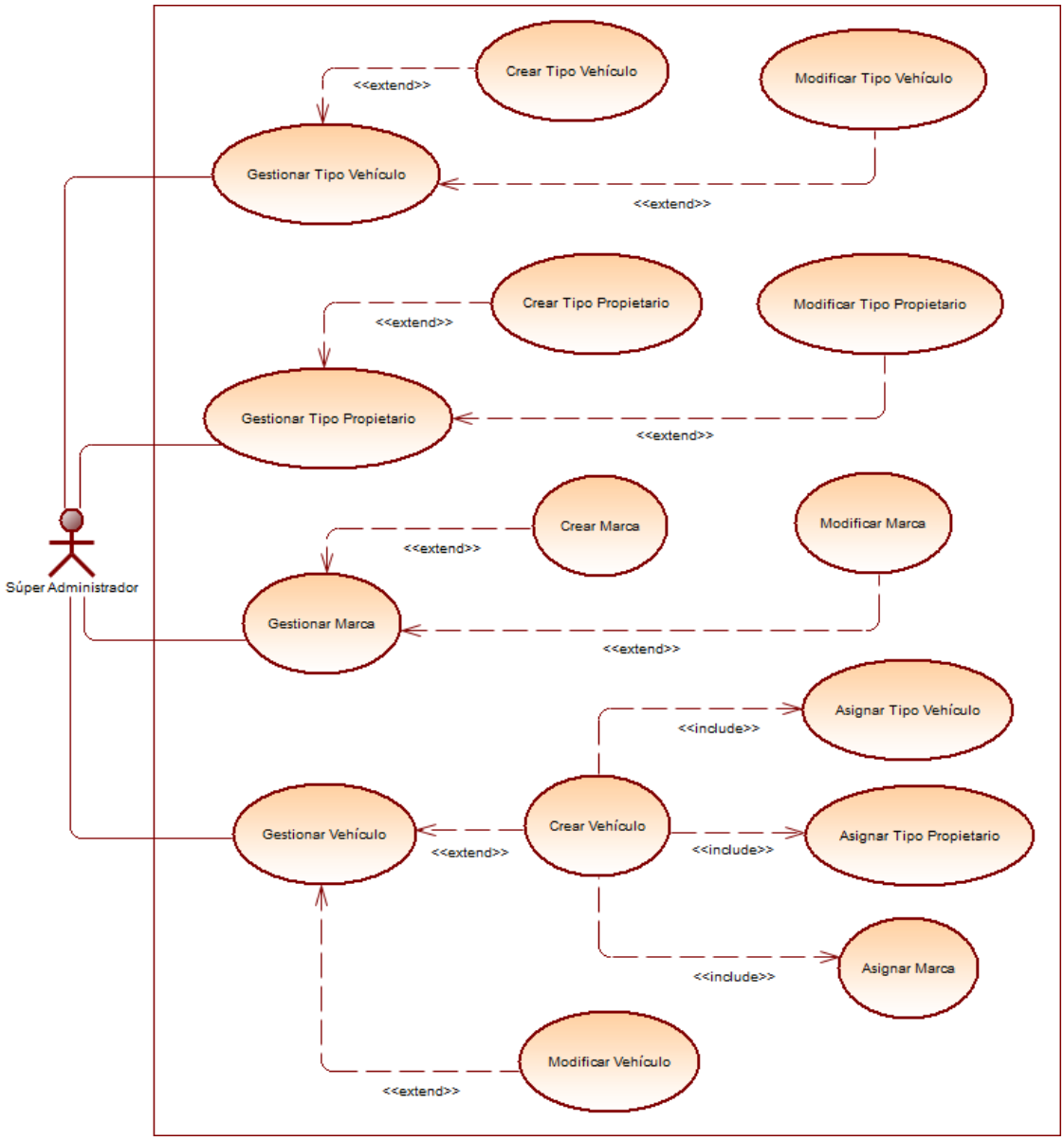


Figura 3. Diagrama de caso representando el curso de gestión de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

En la Tabla 8 se describe el curso normal que tomará el caso de uso gestión de vehículos durante el proceso gestión de vehículos nuevos y existentes por parte del Súper Administrador.

Tabla 8.
Curso normal de gestión de vehículos

Curso normal de caso de uso gestión de vehículos	
1	El Súper Administrador crea y modifica Tipos de Vehículos.
2	El Súper Administrador crea y modifica Tipos de Propietario.
3	El Súper Administrador crea y modifica Marcas de Vehículos.
4	El Súper Administrador crea y modifica Vehículos.

Nota. Descripción del curso normal del caso de uso gestión de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.2. Modelo de contenido

Este modelo permite especificar el contenido de la aplicación por medio de la descripción de los objetos que interactúan entre sí de ser el caso, y que describen el problema a solucionar.

Los objetos son descritos mediante un diagrama de clase donde cada clase posee atributos que equivalen a las características de un objeto, de igual forma una clase contiene métodos que describen ciertas tareas propias de un objeto.

2.2.1. Diagrama de clases del Sistema de Gestión de Transporte.

En la Figura 4 se puede apreciar el diagrama de clases para esta solución, el cual se lo interpreta de la siguiente manera.

Diagrama de clases

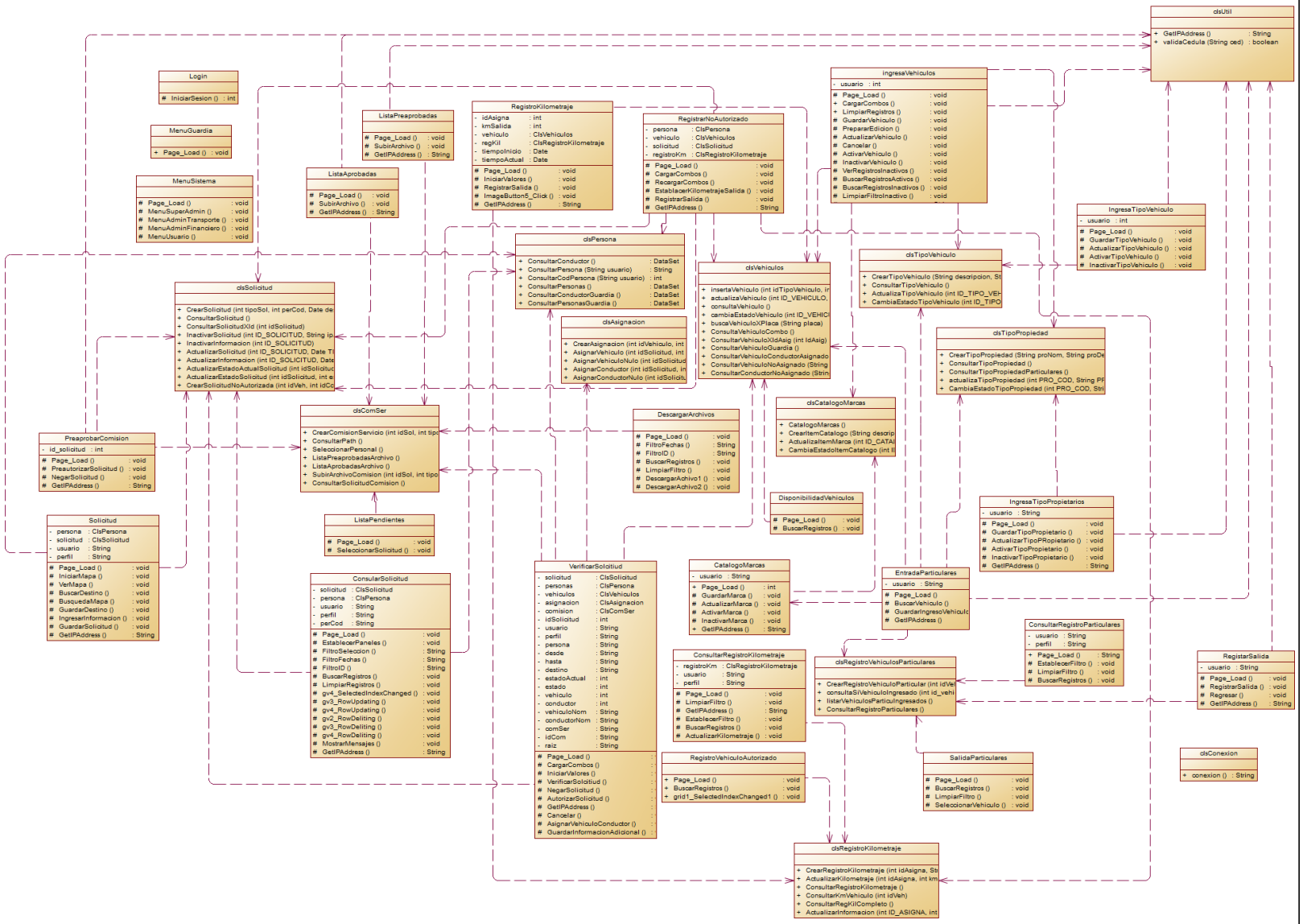


Figura 4. Diagrama de clases que representa todas las clases del Sistema de Gestión de Transporte
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.2.2. Clases de vistas y controladores.

Estas clases son las encargadas de manejar las pantallas que se le muestra al usuario y de realizar llamadas a las clases que manejan las llamadas a la base de datos.

Tabla 9.
Clases de vistas y controladores

Clase	**Descripción**
Login	Se encarga de realizar el logeo a la aplicación validando el usuario y la contraseña.
MenuGuardia	Crea el menú para para que el usuario tenga acceso a las pantallas del guardia de seguridad que se encargará de registrar la entrada y salida de vehículos.
MenuSistema	Crea el menú para que el usuario tenga acceso a las pantallas del sistema dependiendo del tipo de usuario que se está logeando.
ListaPreaprobadas	Se encarga de mostrar en pantalla la lista de solicitudes pre aprobadas.
ListaAprobadas	Se encarga de mostrar en pantalla la lista de solicitudes aprobadas.
PreaprobarComision	Clase encargada de registrar la preaprobación de las solicitudes con comisión.
Solicitud	Se encarga de crear y registrar las solicitudes de movilización del usuario.
ListaPendientes	Se encarga de listar en pantalla todas las solicitudes pendientes con comisión de servicios que requieren archivo.
ConsultarSolicitud	Clase que lista en pantalla todas las solicitudes y permite filtrarlas por aprobadas, negadas, pendientes, etc.
RegistroKilometraje	Registra los kilometrajes de entrado o de salida de un vehículo.
RegistrarNoAutorizado	Realiza el registro de la salida de un vehículo no autorizado.
IngresaVehiculos	Se encarga de registrar un nuevo vehículo ya sea particular, perteneciente a la super intendencia o alquilado. También lista en pantalla los vehículos activos y los inactivos.
IngresaTipoVehiculo	Controla la pantalla que permite gestionar el catálogo de tipos de vehículos y además listar los activos e inactivos.
IngresaTipoPropietarios	Controla la pantalla que permite gestionar el catálogo de tipos de propietarios y además listar los activos e inactivos.

DescargarArchivos	Permite que el usuario descargar los archivos de comisión de servicio que se subieron con la solicitud.
DisponibilidadVehiculos	Muestra en pantalla los vehículos y conductores asignados a una solicitud, así como los vehículos y conductores disponibles en una determinada fecha.
Asignación	Controla la pantalla que permite al usuario asignar un conductor o un vehículo a una determinada solicitud, además de que si la solicitud requiere comisión de servicio permite subir un archivo.
CatalogoMarcas	Controla la pantalla que permite gestionar el catálogo de marcas de vehículos, además de listar los activos e inactivos.
EntradaParticulares	Permite al guardia registrar la entrada de los vehículos particulares a las instalaciones de la super intendencia.
ConsultarRegistroKilometraje	Permite al usuario consultar y editar los registros de kilometraje tanto de entrada como de salida de un determinado vehículo o solicitud.
RegistarSalida	Permite al guardia registrar la salida de los vehículos particulares de las instalaciones de la Superintendencia.
RegistroVehiculoAutorizado	Permite listar y filtrar en pantalla las solicitudes autorizadas.
SalidaParticulares	Permite al guardia listar y filtrar en pantalla los vehículos particulares que se encuentran actualmente dentro de las instalaciones de la Superintendencia para posteriormente registrar su salida.

Nota. Descripción de las clases de vistas y controladores
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.2.3. Clases del modelo.

Estas clases son utilizadas por las clases que controlan las pantallas para realizar llamadas a los procedimientos almacenados de la base de datos.

Tabla 10.
Clases de Modelo

Clase	**Descripción**
clsSolicitud	Clase encargada de llamar a los procedimientos almacenados para gestionar las solicitudes de movilización.
clsComSer	Clase encargada de la llamada de procedimientos almacenados para gestionar las solicitudes que requieren comisión de servicio.

clsUtil	Clase que tiene las funciones de obtener la dirección IP y la de validar cédula del usuario.
clsAsignacion	Se encarga de registrar en la base de datos las asignaciones de vehículo y conductor que se realizan a una solicitud por parte del administrador.
clsVehiculo	Se encarga de listar, actualizar y crear información de los vehículos utilizando procedimientos almacenados.
clsTipoVehiculo	Se encarga de gestionar el catálogo de los tipos de vehículos.
clsTipoPropiedad	Se encarga de gestionar el catálogo de los tipos de propietarios de los vehículos.
clsCatalogoMarcas	Gestiona el catálogo de marcas del vehículo.
clsRegistroVehiculosParticulares	Permite registrar o consultar los vehículos particulares que ingresan a las instalaciones de la SCPM.
clsRegistroKilometraje	Permite gestionar la información de los kilometrajes de los vehículos de uso de la super intendencia.
clsConexion	Clase utilizada para obtener la cadena de conexión a la base de datos.
clsPersona	Se encarga de consultar en la base de datos la información del personal como pueden ser guardias, usuarios o conductores.

Nota. Descripción de las clases del modelo de datos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Una vez completado el modelo de contenido de la aplicación se debe continuar la fase de diseño mediante el modelo de navegación, el cual especifica el contenido de navegación de una aplicación que en este caso es web.

2.3. Modelo de navegación

La navegación de una aplicación web es crucial dentro de la definición de la estructura visual, la cual forma parte de la interacción entre el usuario y la aplicación. Los diagramas que se han propuesto en base a los distintos perfiles que maneja la aplicación se describen a continuación.

2.3.1. Navegación del perfil administrador de movilización.

El usuario administrador podrá ver todas las solicitudes y realizar un filtro por pendientes, aprobadas o negadas. Con las solicitudes pendientes el administrador podrá asignarle un conductor o un vehículo, editar esta información y finalmente aprobarla o negarla.

Diagrama de navegación del perfil administrador de movilización

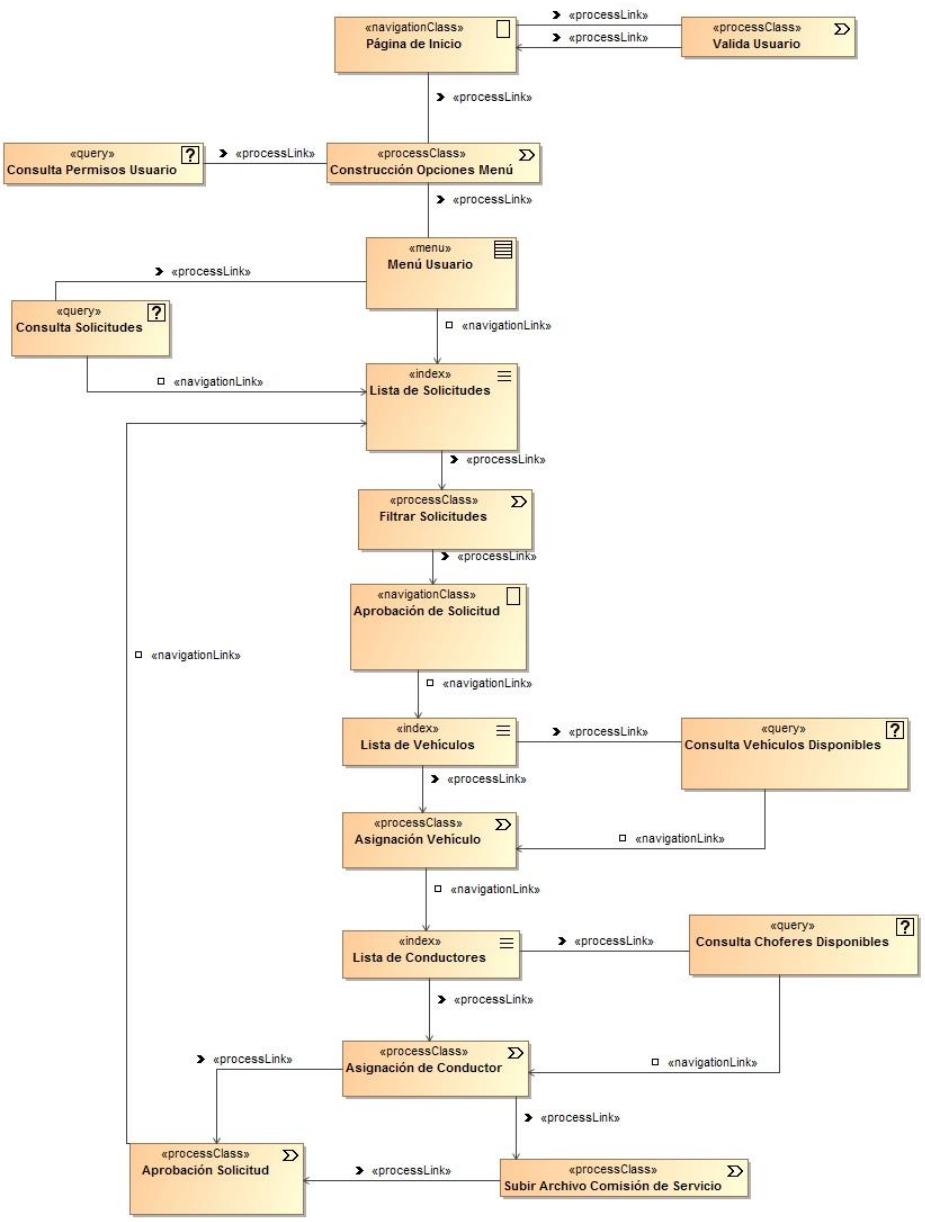


Figura 5. Diagrama que representa la navegación del perfil administrador de movilización
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

La figura 5 se explica el diagrama de navegación para el perfil de usuario administrador, donde al usuario se le mostrará la página de inicio en donde deberá iniciar sesión ingresando su usuario y contraseña. El sistema procederá a validar si las credenciales ingresadas son correctas para posteriormente consultar el perfil de usuario que tiene asignado para posteriormente crear un menú con las opciones habilitadas para ese perfil que en este caso es de administrador.

Al administrador se le mostrará una lista con todas las solicitudes la cual podrá filtrar por solicitudes negadas, aprobadas, pendientes y estas a su vez se las podrá filtrar por un rango de fechas.

Al ubicarse en la lista de solicitudes pendientes, podrá seleccionar una lo cual desplegará la información de la solicitud y podrá asignarle un conductor o un vehículo y en el caso de que esta solicitud sea por comisión de servicio, el administrador podrá adjuntar un documento opcional para finalmente aprobar la solicitud o negarla según considere conveniente.

2.3.2. Navegación del perfil empleado.

La figura 6 muestra el diagrama de navegación para el perfil de usuario Empleado. El usuario empleado podrá ver todas las solicitudes que haya creado con su respectivo estado.

También podrá seleccionar una solicitud pendiente para actualizar los datos ingresados o inactivarla. Al usuario se le mostrará la página de inicio en donde deberá iniciar sesión ingresando su usuario y contraseña. El sistema procederá a validar si las credenciales ingresadas son correctas para posteriormente consultar el perfil de usuario que tiene

asignado para posteriormente crear un menú con las opciones habilitadas para ese perfil que en este caso es de Empleado.

Diagrama de navegación del perfil empleado

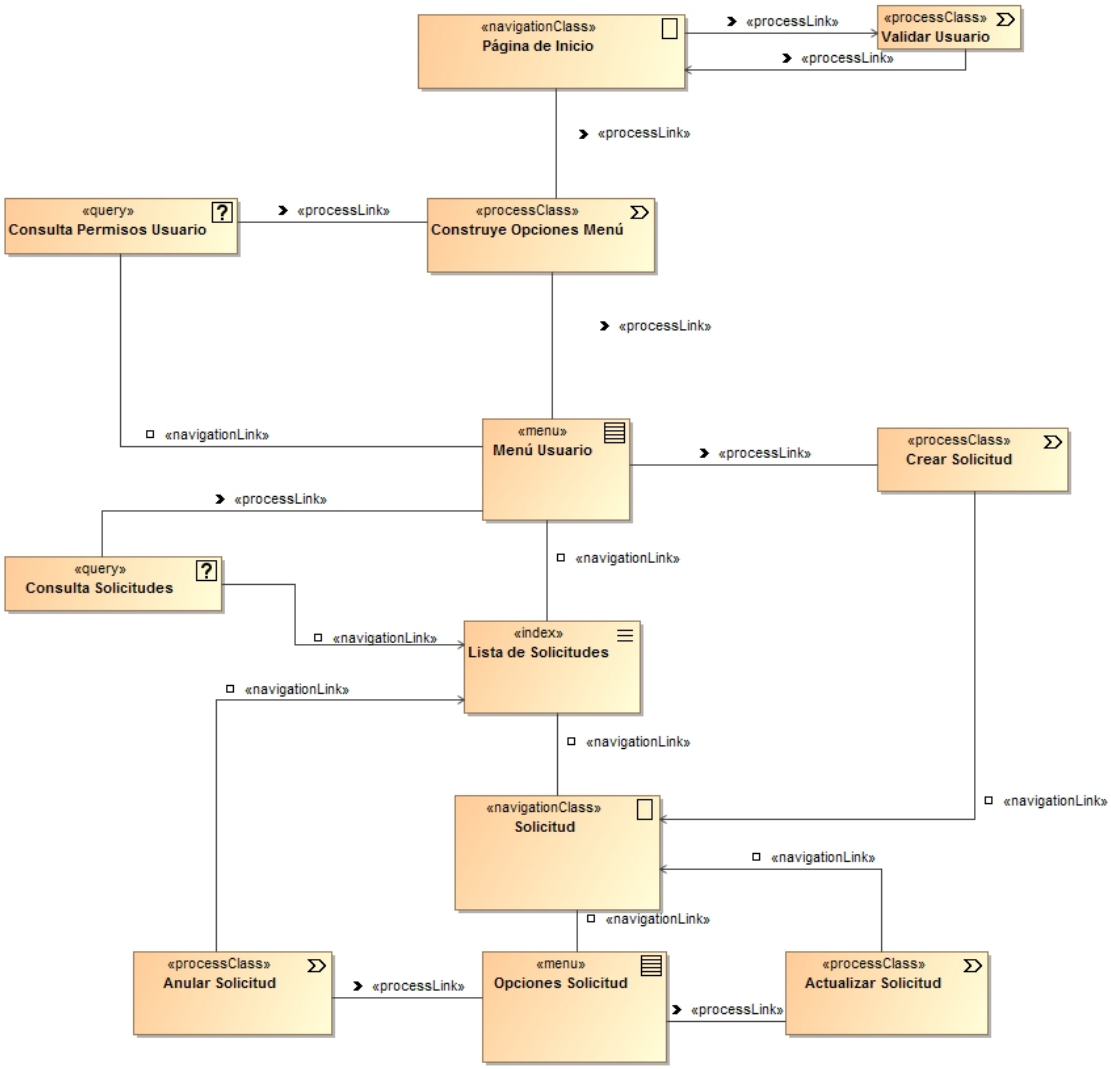


Figura 6. Diagrama que representa la navegación del perfil empleado
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Al usuario empleado se le mostrará todas las solicitudes que haya creado las cuales podrá filtrar por su estado siendo estas pendientes, aprobadas o negadas.

También podrá crear una nueva solicitud o selecciona una solicitud pendiente y editarla o negarla.

2.3.3. Navegación del perfil personal de seguridad.

La figura 7 se aprecia el diagrama de navegación para el perfil de usuario Guardia de Seguridad.

Diagrama de navegación del personal de seguridad

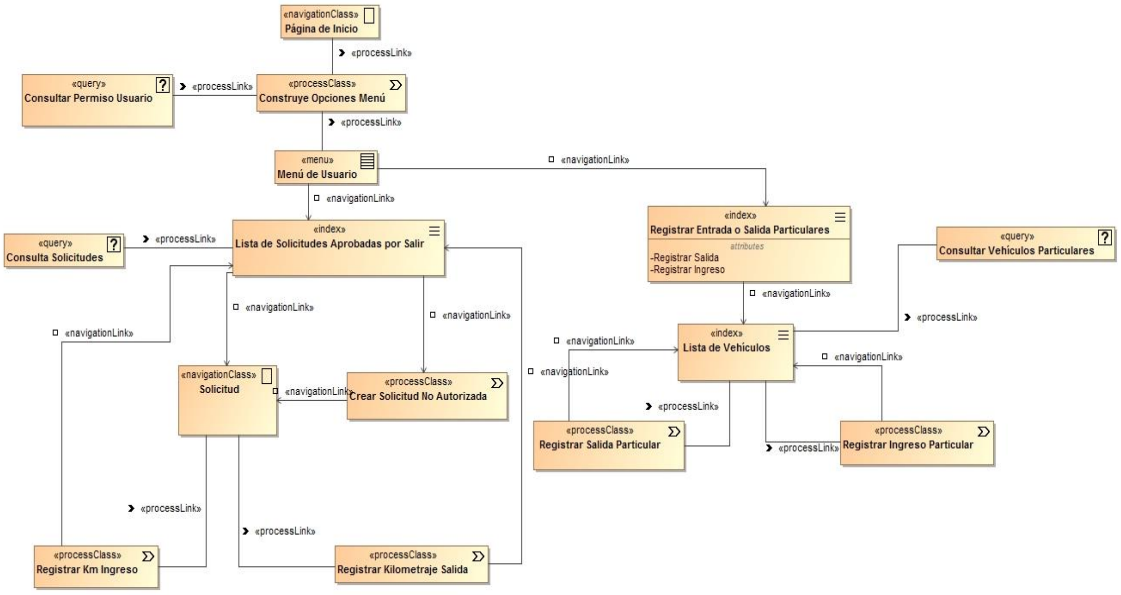


Figura 7. Diagrama que representa la navegación del perfil personal de seguridad
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El usuario Guardia de Seguridad podrá ver todas las solicitudes aprobadas que tienen programadas las salidas para la fecha actual. También podrá registrar el ingreso y salida de vehículos particulares así como el kilometraje de ingreso y el de salida de los vehículos pertenecientes a la SCPM.

Al usuario Guardia de Seguridad se le mostrará dos opciones en el menú. La primera le mostrará una lista de solicitudes aprobadas que tienen programadas la salida para ese

mismo día y la segunda en donde podrá registrar los vehículos particulares que ingresen a la SCPM.

Al ubicarse en la lista de solicitudes aprobadas pendientes de salida el guardia de seguridad podrá seleccionar una solicitud y registrar el kilometraje de ingreso o el de salida. Al registrar el kilometraje de ingreso de una solicitud esta saldrá de la lista.

También en esta misma pantalla el guardia de seguridad podrá registrar la salida de vehículos no autorizados.

El guardia de seguridad podrá registrar los vehículos particulares que ingresen a las instalaciones de la SCPM y una vez ingresados podrá registrar su salida.

2.3.4. Navegación del perfil administrador financiero.

La figura 8 describe el diagrama de navegación para el perfil de usuario del Administrador financiero.

Al usuario se le mostrará la página de inicio en donde deberá iniciar sesión ingresando su usuario y contraseña. El sistema procederá a validar si las credenciales ingresadas son correctas para posteriormente consultar el perfil de usuario que tiene asignado para posteriormente crear un menú con las opciones habilitadas para ese perfil que en este caso es de Administrador Financiero.

El usuario Administrador financiero podrá ver todas las solicitudes pendientes que hayan sido creadas por comisión de servicio. A estas solicitudes podrá adjuntarles un documento opcional y pre aprobarlas o negarlas según lo considera conveniente.

Diagrama de navegación del perfil administrador financiero

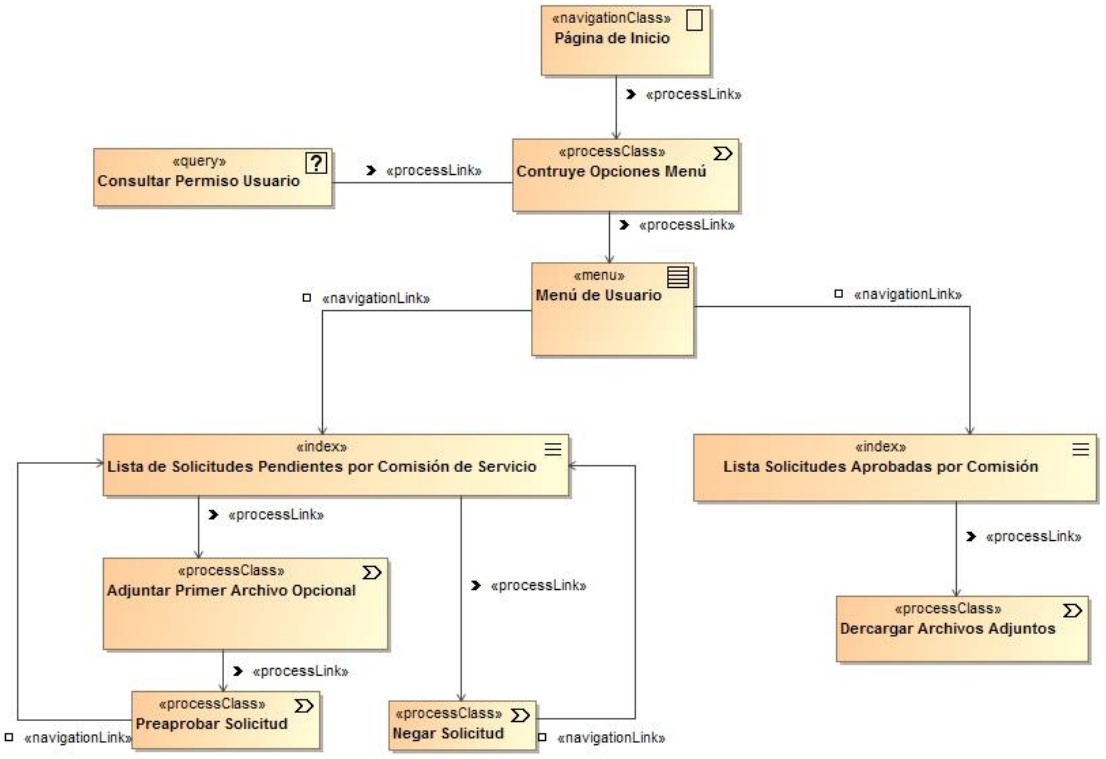


Figura 8. Diagrama que representa la navegación del perfil administrador financiero
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

También podrá ver todas las solicitudes por comisión de servicio aprobadas y podrá descargar los archivos adjuntos que tengan asociadas.

2.3.5. Navegación del perfil súper administrador.

En la figura 9 se observa el diagrama de navegación para el perfil de Súper Administrador el cual posee acceso total del sistema de gestión de transporte.

La navegación permite conocer de qué manera se encuentran enlazadas las páginas de una aplicación web con la finalidad de estructurar el contenido visual de acuerdo al análisis de requisitos y el modelo de contenido descritos anteriormente.

El siguiente paso en la fase de diseño de acuerdo con la metodología UWE, es la definición del modelo de presentación, el cual contendrá las interfaces visuales de usuario que pueden ser consideradas una entrada del sistema, en las cuales el usuario final puede interactuar directamente con la aplicación.

Diagrama de navegación del perfil súper administrador

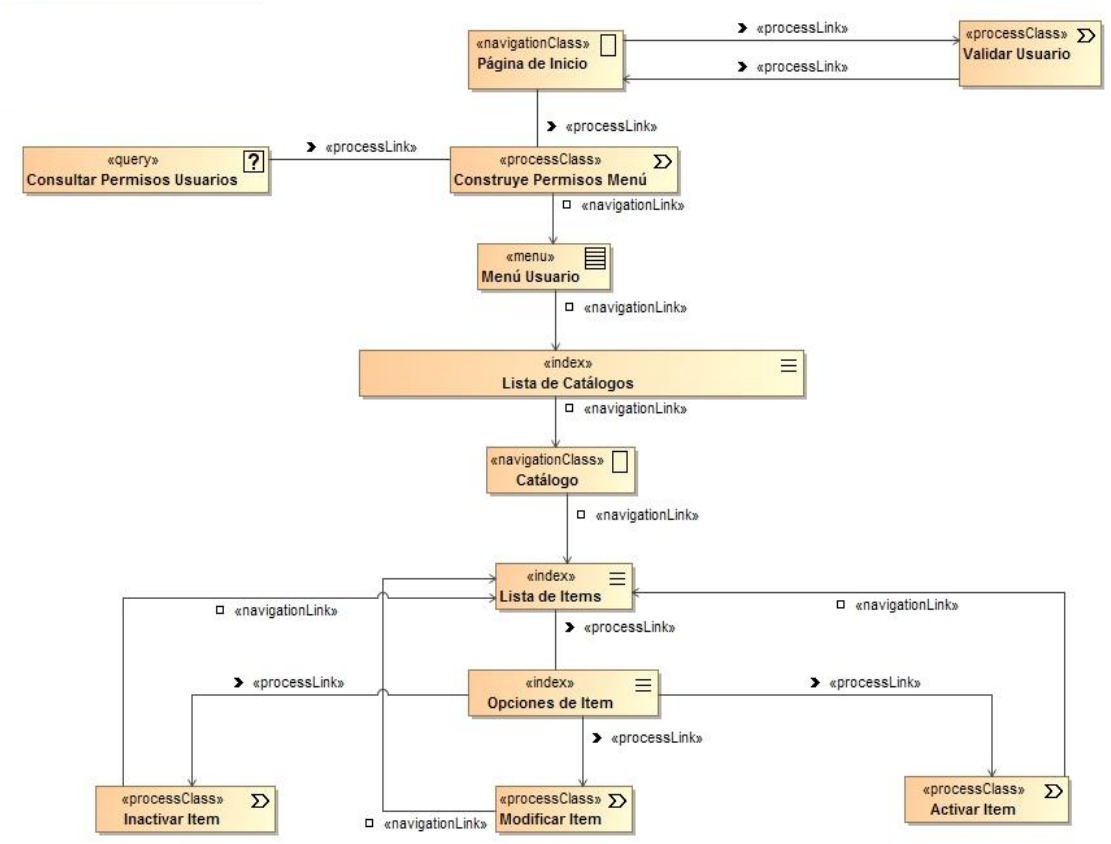


Figura 9. Diagrama que representa la navegación del perfil súper administrador
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4. Modelo de presentación

Las interfaces de usuario son de gran relevancia dentro de una aplicación web debido a que las mismas permiten que el usuario final utilice el sistema de manera visual. UWE hace énfasis en esta sección planteando un modelo de presentación debido a que las aplicaciones web pueden ser accedidas con facilidad mediante redes internas,

institucionales, e incluso desde la web, a través de un navegador de internet. Por este motivo se debe estructurar de manera adecuada las interfaces del sistema mediante diagramas de presentación.

2.4.1. Diagramas de Presentación.

Cabe mencionar que los diagramas de presentación establecen los elementos **<<presentationAlternatives>>** como ventanas emergentes o pop-up[1] que se muestran en la página web, y **<<customComponent>>** como lista de datos seleccionable mejor conocida como **combobox**[2] o **dropdownlist**[3], los cuales son utilizados para definir interfaces de usuario. También todas las páginas de la aplicación poseen elementos en común, los cuales se encuentran definidos inicialmente en la parte superior de cada página y que corresponden a una página maestra[4]. Los elementos de la página maestra son una imagen como encabezado, un cuadro de texto que muestra la hora actual, y botones que permiten dirigir al usuario a la página de inicio y también cerrar la sesión actual.

2.4.1.1. Vehículos.

La figura 10 muestra la página de vehículos donde se gestiona la información referente a los vehículos. Inicialmente se cuenta con un formulario de ingreso de datos donde se especifica la información de vehículos que se va a crear, como placa, marca, modelo, año, número de ocupantes, combustible, tipo de vehículo y tipo de propietario. También se

[1] Elemento que representa una ventana que se muestra sobre una página web, la cual puede ser activada o no por el usuario.
[2] Se utiliza para desplegar datos en una lista dinámica donde el usuario puede seleccionar una opción como dato de entrada.
[3] El mismo concepto de combobox.
[4] Página web de la cual otras páginas pueden heredan sus propiedades y elementos.

puede apreciar en la parte inferior los registros creados que se encuentren activos, inactivos y un formulario de búsqueda de registros, los cuales se muestran si se accionan los botones de **Ver Activos** y **Ver Inactivos**.

Diagrama de presentación de vehículos

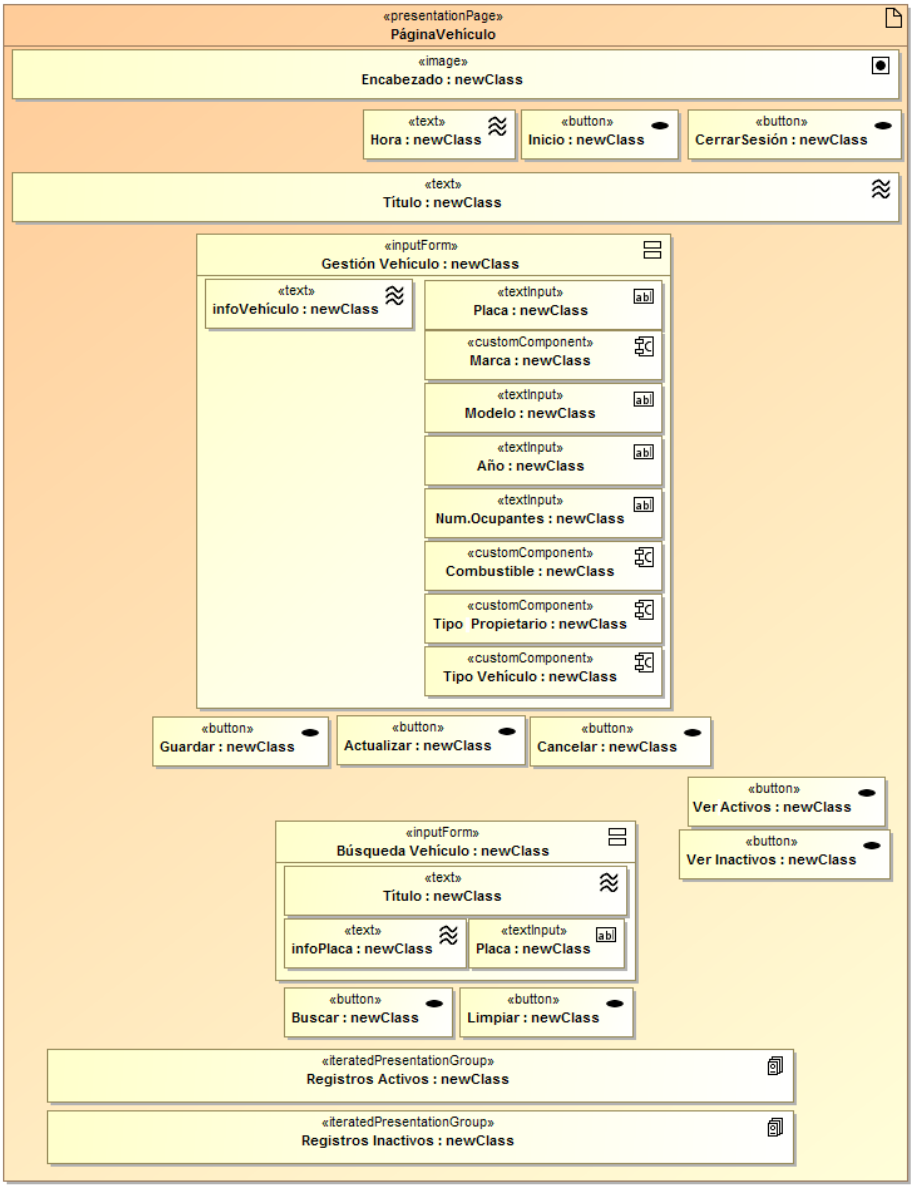


Figura 10. Diagrama que representa la interfaz de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.2. Solicitud de movilización.

En la figura 11 se observa la página de solicitudes de movilización donde se ingresan las solicitudes de movilización de los empleados. Inicialmente se cuenta con un formulario de ingreso de datos donde se especifica la información de una solicitud que se va a crear, como fecha y hora desde, fecha y hora hasta, destino y comisión de servicio. En el formulario de ingreso existe el botón **Ver Mapa**, el cual permite visualizar una ventana emergente donde se puede hacer una búsqueda de destino por medio de un mapa. Por último el botón **Guardar** permite registrar los datos del formulario y el botón **Regresar** direcciona al usuario a otra página del sistema.

Diagrama de presentación de solicitud de movilización

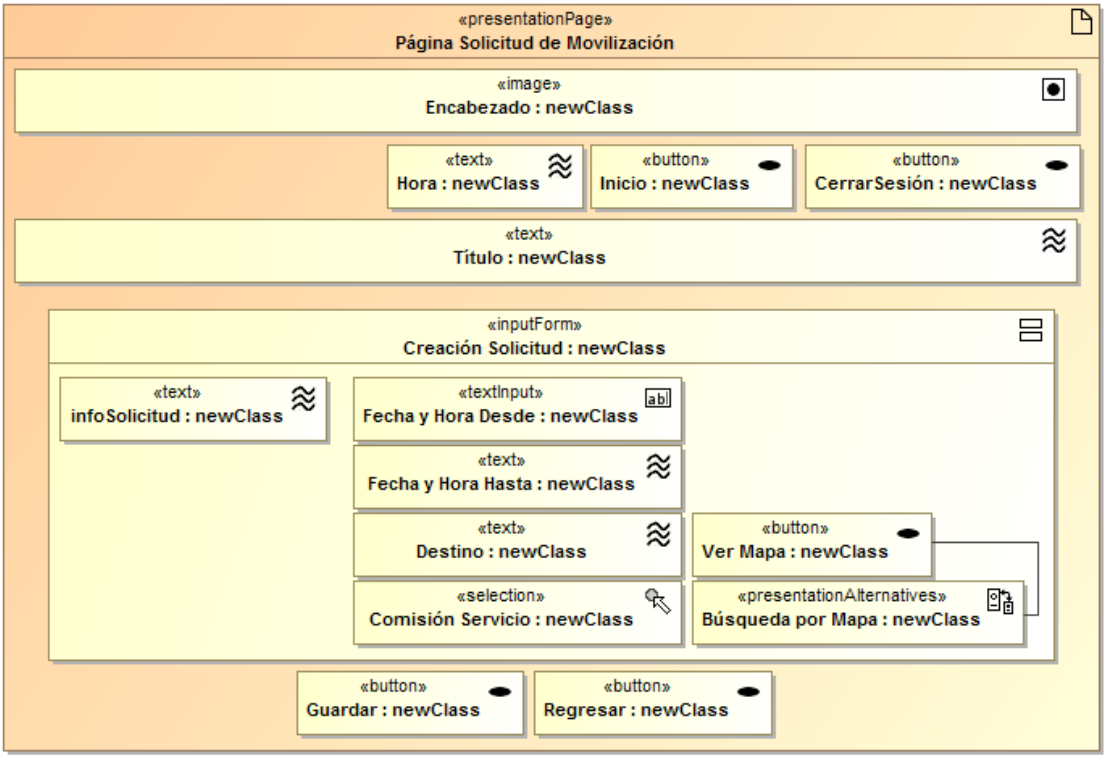


Figura 11. Diagrama que representa la interfaz de solicitud de movilización
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.3. Preautorización de solicitud.

En la figura 12 se explica la página de preautorización de solicitudes donde se preautorizan las solicitudes de movilización que se encuentre con comisión de servicio. Inicialmente se cuenta con un formulario de ingreso de datos donde se especifica cierta información a una solicitud seleccionada como si requiere un archivo adjunto, subir archivo y observación. El botón **Preautorizar** permite registrar los datos del formulario y el botón **Negar** cambia el estado del solicitud a negado.

Diagrama de presentación de preautorización de solicitud

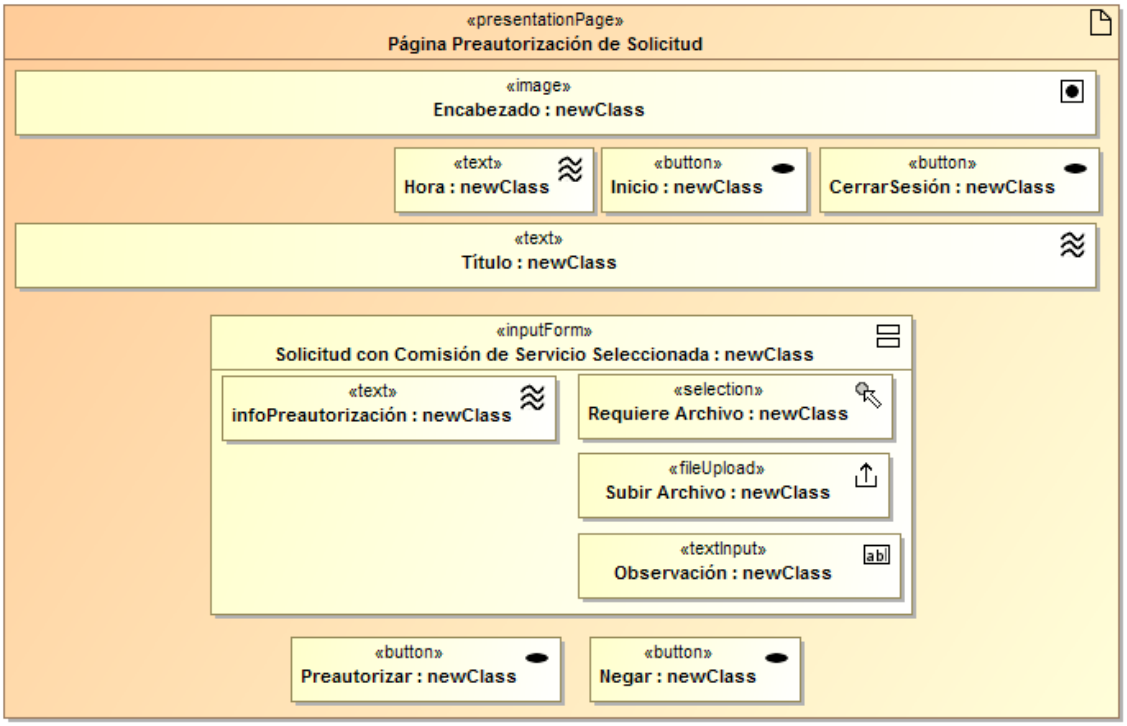


Figura 12. Diagrama que representa la interfaz de preautorización de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.4. Asignación.

En la figura 13 se aprecia la página de asignación en la cual se puede verificar, asignar un vehículo y conductor, autorizar y negar solicitudes pendientes sin comisión de servicio y

solicitudes preautorizadas con comisión de servicio. Inicialmente se cuenta con un formulario donde se muestra la información de una solicitud que ha sido seleccionada, y adicionalmente existe el botón **Inf.Adicional** que se muestra solo para solicitudes con comisión de servicio que han sido preautorizadas, el cual permite mostrar una ventana emergente donde se especifica si se requiere adjuntar un archivo, subir archivo, observación, y botones para guardar la información y cancelar la operación. Los botones **Verificar**, **Autorizar**, **Negar**, **Asignar** y **Regresar** se mostraran de acuerdo a la interacción que exista entre el usuario y la página. Por último el formulario Asignación que parece en la parte inferior puede ser mostrado con el botón **Asignar**. Este formulario permite asignar un vehículo y un conductor a la solicitud seleccionada.

Diagrama de presentación de asignación

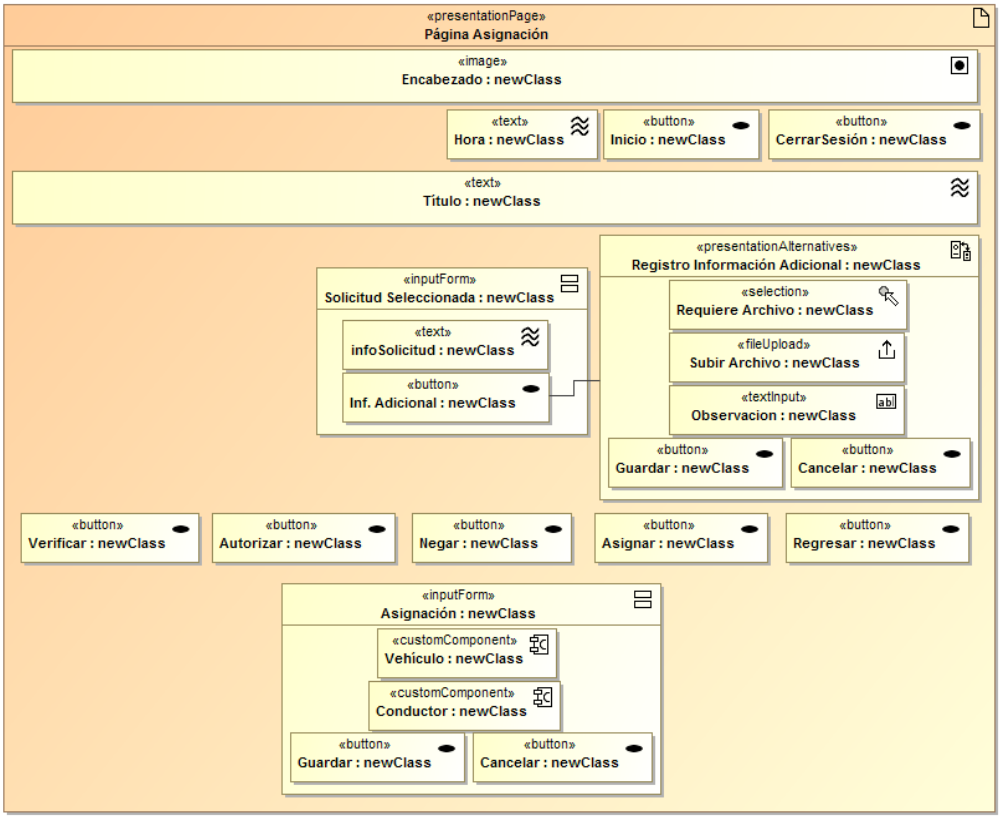


Figura 13. Diagrama que representa la interfaz de asignación
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.5. Solicitudes que requieren archivo adjunto.

En la figura 14 se expone la página de solicitudes que requieren archivo. Estas solicitudes pueden ser aquellas que son de comisión de servicio o que se especificó que requieren archivo adjunto tanto en la página de preautorizar solicitud y asignación. Inicialmente se cuenta con una tabla de información donde se visualiza las solicitudes con comisión de servicio que requieren un archivo adjunto y no cuentan con uno. La tabla cuenta con información de la solicitud y con la opción de subir el archivo faltante.

Diagrama de presentación de solicitudes que requieren archivo adjunto

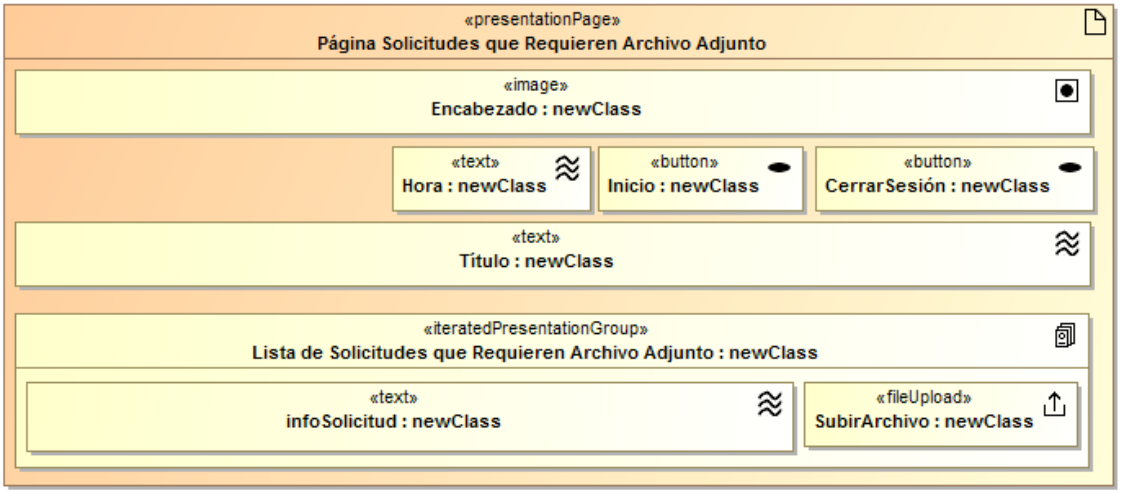


Figura 14. Diagrama que representa la interfaz de solicitudes que requieren archivo adjunto
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.6. Registro de ingreso de vehículo particular.

En la figura 15 se presenta la página de registro de ingreso de vehículo particular, donde se define un formulario de búsqueda por placa con la finalidad de encontrar un vehículo con registros anteriores y no ingresar nuevamente toda la información del mismo. Si existe el vehículo la información se verá reflejada en el formulario Registro Vehículo Particular, caso contrario se debe ingresar toda la información del formulario como nombre y cedula

del conductor, placa, marca, modelo, año, número de ocupantes, combustible, tipo de vehículo, tipo de propietario y observación. Por último el botón **Guardar** permite registra el ingreso del vehículo particular.

Diagrama de presentación de registro de ingreso de vehículo particular

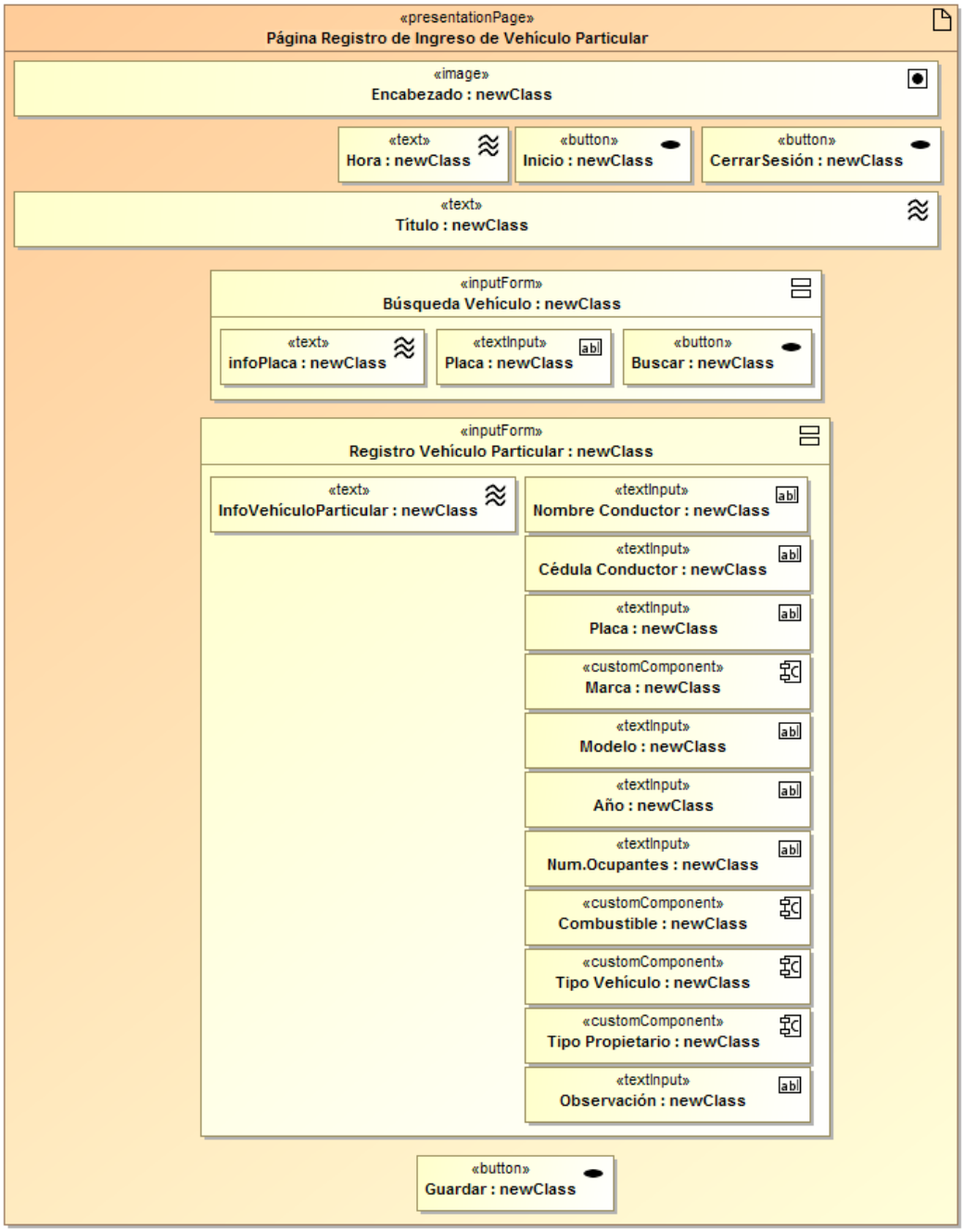


Figura 15. Diagrama que representa la interfaz de registro de ingreso de vehículo particular
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.7. Registro de salida de vehículo particular.

La figura 16 muestra la página de registro de salida de vehículo particular, donde se define un formulario con información del vehículo seleccionado y una observación. El botón **Guardar** permite registrar la salida del vehículo particular, y el botón **Cancelar** anula el proceso.

Diagrama de presentación de registro de salida de vehículo particular

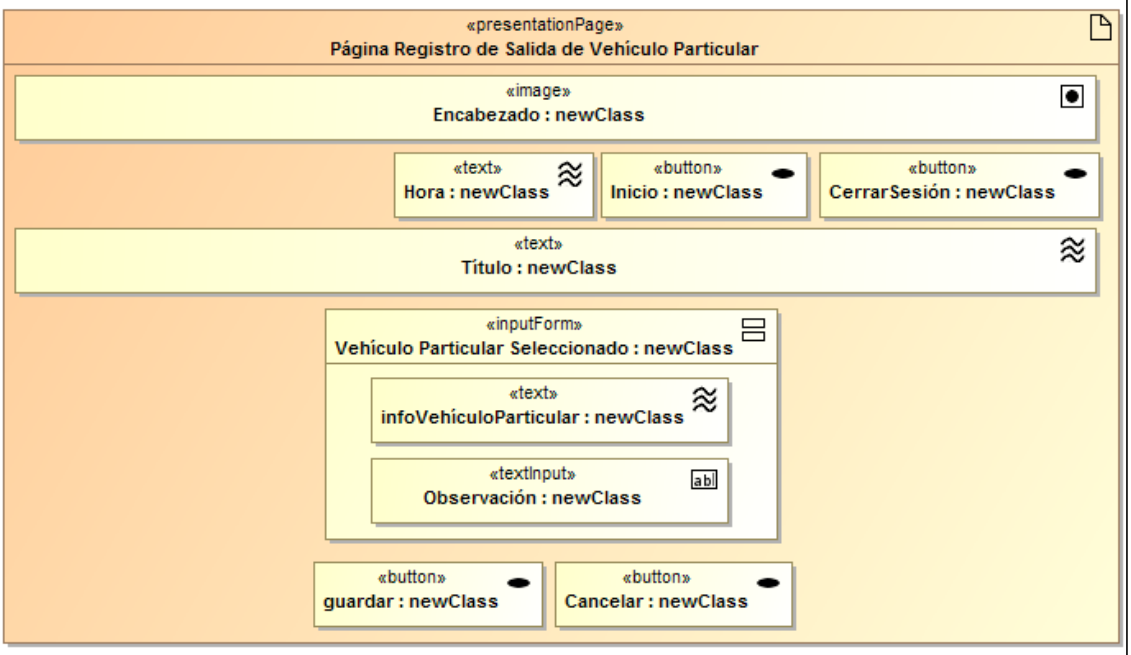


Figura 16. Diagrama que representa la interfaz de registro de salida de vehículo particular
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.8. Registro de entrada y salida de vehículo institucional.

En la figura 17 se observa la página de registro de entrada y salida de vehículos institucionales, donde se define un formulario con información del vehículo institucional seleccionado y una observación. El botón **Guardar** permite registrar la entrada o la salida del vehículo institucional, y el botón **Regresar** direcciona al usuario otra página del sitio.

Diagrama de presentación de registro de entrada y salida de vehículo institucional

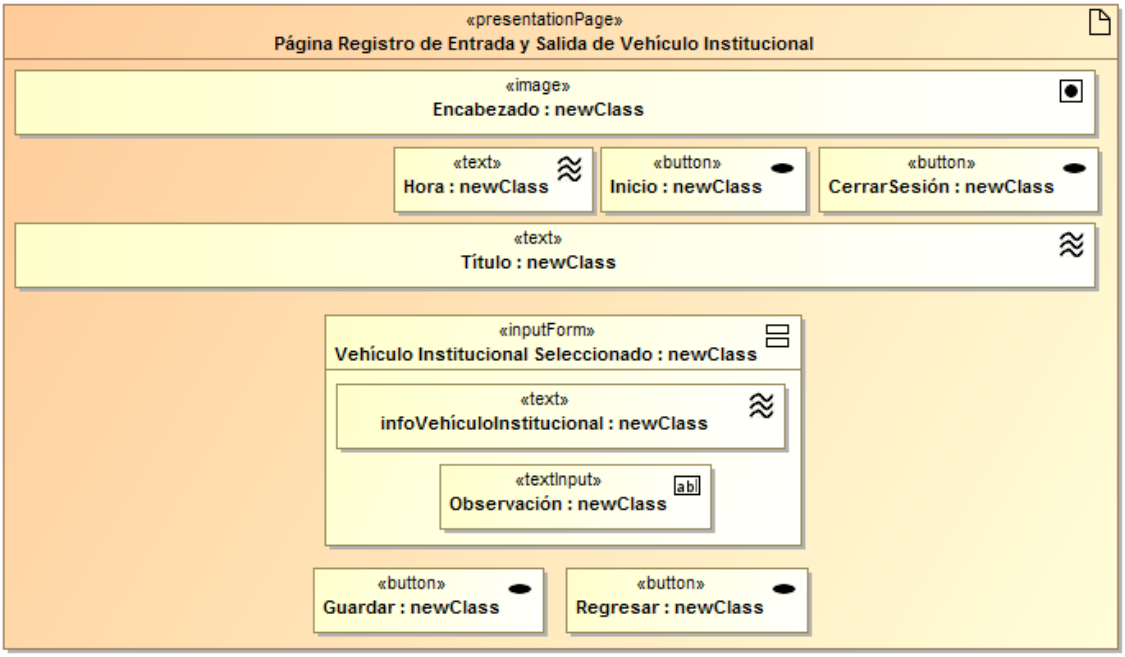


Figura 17. Diagrama que representa la interfaz de registro de entrada y salida de vehículo institucional
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.4.1.9. Registro de salida no autorizada de vehículo institucional.

En la figura 18 se explica la página de registro de salida no autorizada de vehículos institucionales. En el formulario Registra Salida no Autorizada se selecciona el vehículo institucional, el conductor, el usuario, y adicionalmente se ingresa una observación. El botón **Guardar** permite registrar la salida no autorizada del vehículo institucional, y el botón **Regresar** direcciona al usuario otra página del sitio.

El modelo de presentación permite definir por medio de interfaces abstractas propias de UML, como va a ser visualizada la aplicación, donde el usuario final o cliente puede verificar estas interfaces, las cuales podrían requerir cambios de acuerdo a sugerencias y nuevas especificaciones provenientes de los mismos.

Diagrama de presentación de registro de salida no autorizada de vehículo institucional

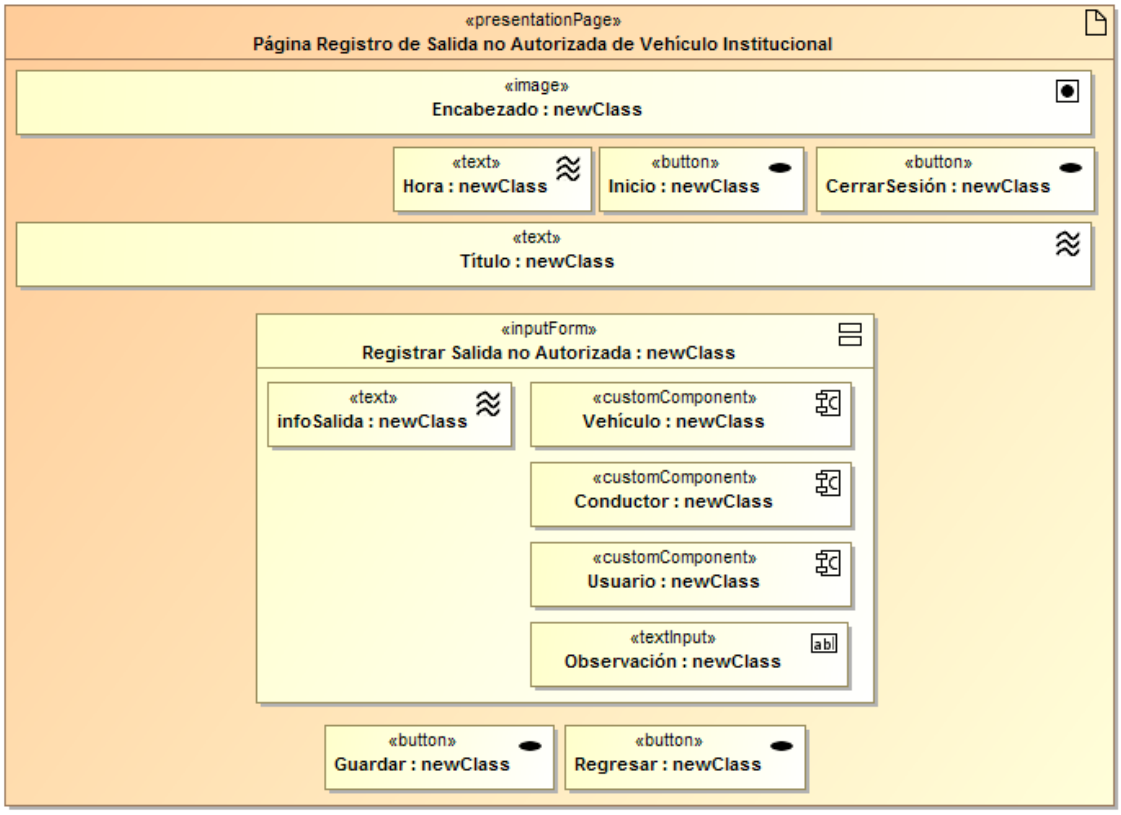


Figura 18. Diagrama que representa la interfaz de registro de salida no autorizada de vehículo institucional
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Para finalizar la etapa de diseño se debe describir un modelo de proceso propio de UWE, el cual define y estructura los procesos como tal junto con el flujo de los mismos.

2.5. Modelo de proceso

En un sistema informático los procesos son la esencia del funcionamiento e integración del mismo, los cuales son encargados de manejar la información y transformarla de acuerdo al tipo de tarea o subrutina que se esté ejecutando dentro una aplicación.

En el modelo de proceso se estructura los procesos y se define el flujo de los mismos con el objetivo de especificar las tareas que interactúan entre sí al momento de ejecutar una determinada acción dentro del sistema.

En este modelo se utiliza los diagramas de estructura de proceso y de flujo de proceso. En el diagrama de estructura de proceso se define los procedimientos que formarán parte de un proceso como tal, en cambio en el diagrama de flujo de proceso se da seguimiento a un proceso, describiendo los caminos posibles que pueden surgir desde el inicio hasta el fin de un proceso.

2.5.1. Diagramas de estructura de proceso.

En estos diagramas se definen clases de procesos por medio de rectángulos, los cuales se caracterizan por su descripción en la parte de superior como <<**processClass**>>.

2.5.1.1. Gestión de vehículo.

La Figura 19 muestra el diagrama de estructura de proceso de gestión de vehículo, el cual representa la gestión de la información referente a los vehículos institucionales, particulares y de transporte.

Diagrama de estructura de proceso de gestión de vehículo

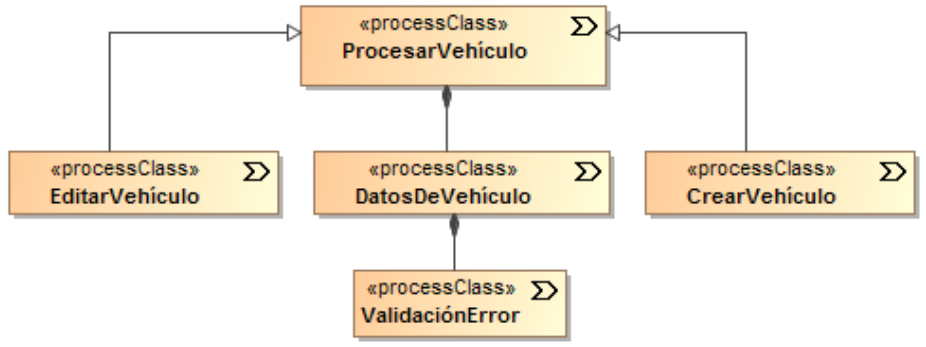


Figura 19. Representación gráfica de la estructura de proceso de gestión de vehículo
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

La clase de proceso principal es **ProcesarVehículo** la cual incluye otra clase de proceso que es **DatosDeVehículo** que representa toda la información de vehículos dentro de la

aplicación. La clase de proceso **DatosDeVehículo** incluye la clase de proceso **ValidaciónError**, la cual permite validar los datos de entrada de vehículos antes de ser procesados. **ProcesarVehículo** hereda sus propiedades a otras 2 clases de proceso que son **EditarVehículo** y **CrearVehículo**, las cuales permiten editar e ingresar nueva información de vehículos en los registros del sistema.

2.5.1.2. Gestión de solicitud.

En la Figura 20 se observa el diagrama de estructura de proceso de gestión de solicitud, en el cual se representa la gestión de la información referente a las solicitudes de movilización que serán manejadas en la aplicación por el personal de la institución.

La clase de proceso principal es **ProcesarSolicitud** la cual incluye otra clase de proceso que es **DatosDeSolicitud** que representa toda la información de solicitudes de movilización dentro de la aplicación. La clase de proceso **DatosDeSolicitud** incluye la clase de proceso **ValidaciónError**, la cual permite validar los datos de entrada de una solicitud antes de ser procesados. **ProcesarSolicitud** hereda sus propiedades a otras 2 clases de proceso que son **EditarSolicitud** y **CrearSolicitud**, las cuales permiten editar e ingresar nueva información de solicitudes de movilización en los registros del sistema.

Diagrama de estructura de proceso de gestión de solicitud

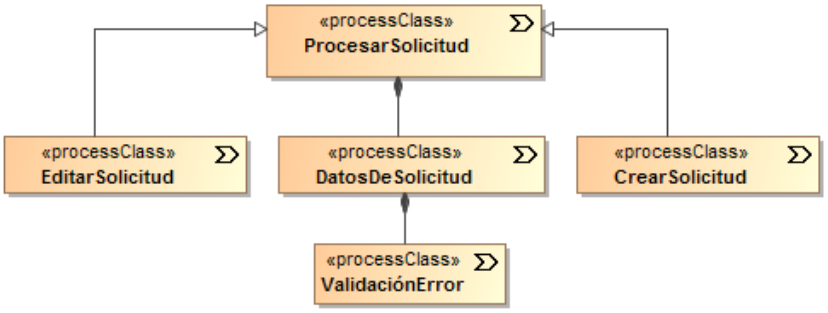


Figura 20. Representación gráfica de la estructura de proceso de gestión de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.5.1.3. Preautorización de solicitud.

En la Figura 21 se explica el diagrama de estructura de proceso de preautorización de solicitud, el cual representa la preautorización y negación de solicitudes de movilización con comisión de servicio. Cabe mencionar que este proceso lo realiza el Administrador Financiero desde la aplicación.

Diagrama de estructura de proceso de preautorización de solicitud

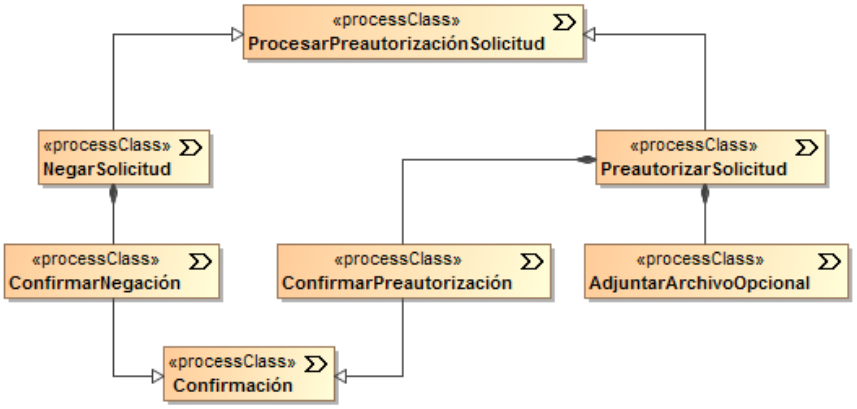


Figura 21. Representación gráfica de la estructura de proceso de preautorización de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

La clase de proceso principal **ProcesarPreautorizaciónSolicitud** se desglosa en **NegarSolicitud** y **PreautorizarSolicitud**, las cuales describen la preautorización y negación de una solicitud de movilización con comisión de servicio por parte del Administrador Financiero. **PreautorizarSolicitud** incluye a la clase de proceso **AdjuntarArchivoOpcional**, donde el Administrador Financiero puede adjuntar de manera opcional un archivo de preautorización. La clase de proceso Confirmación se desglosa en las clases de proceso **ConfirmarNegación** y **ConfirmarPreautorización** que son incluidas en **NegarSolicitud** y **PreautorizarSolicitud** respectivamente.

2.5.1.4. Autorización de solicitud.

En la Figura 22 se aprecia el diagrama de estructura de proceso de autorización de solicitud, el cual representa la aprobación y negación que puede tener una solicitud de movilización, incluidas aquellas con comisión de servicio. El Administrador de Movilización es quien realiza este proceso desde la aplicación.

Diagrama de estructura de proceso de autorización de solicitud

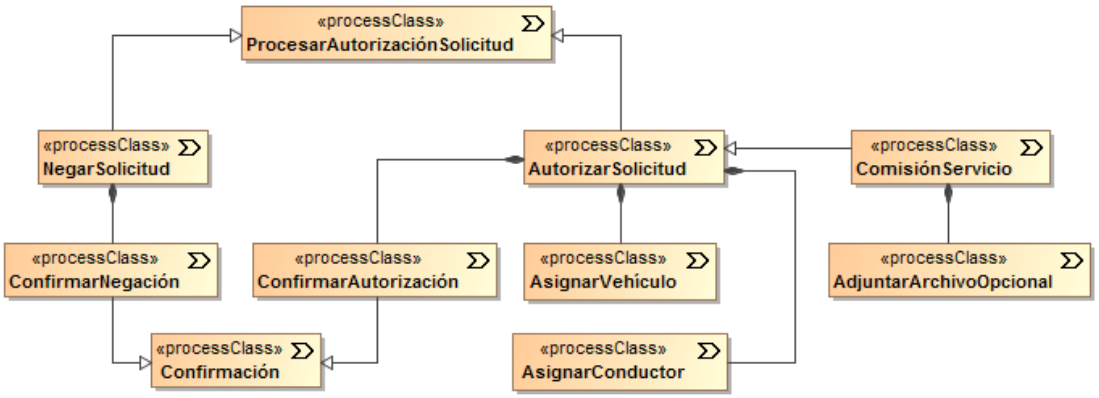


Figura 22. Representación gráfica de la estructura de proceso de autorización de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

La clase de proceso principal **ProcesarAutorizaciónSolicitud** se desglosa en **NegarSolicitud** y **AutorizarSolicitud**, las cuales describen la autorización y negación de una solicitud de movilización por parte del Administrador de Movilización. **AutorizarSolicitud** incluye a las clases de proceso **AsignarVehículo** y **AsignarConductor**, permitiendo asignar un vehículo y un conductor a una solicitud para que sea aprobada. De existir una solicitud con comisión de servicio que se desee aprobar, la clase de procesos **ComisiónServicio** hereda las propiedades de **AutorizarSolicitud**, y además permite al Administrador de Movilización adjuntar un archivo opcional de autorización. La clase de proceso Confirmación se desglosa en las clases de proceso

ConfirmarNegación y **ConfirmarAautorización** que son incluidas en **NegarSolicitud** y **AutorizarSolicitud** respectivamente.

2.5.1.5. Registro de ingreso y salida de vehículos.

En la Figura 23 se expone el diagrama de estructura de proceso de registro de ingreso y salida de vehículos, el cual describe la gestión de registros de vehículos institucionales, de transporte y particular. El Personal de Seguridad es el encargado de realizar este proceso desde la aplicación.

La clase de proceso principal **ProcesarIngresoSalidaVehículos** se divide en **ProcesarVehículoInstitucional**, **ProcesarVehículoTransporte** y **ProcesarVehículoParticular**.

Diagrama de estructura de proceso de registro de ingreso y salida de vehículos

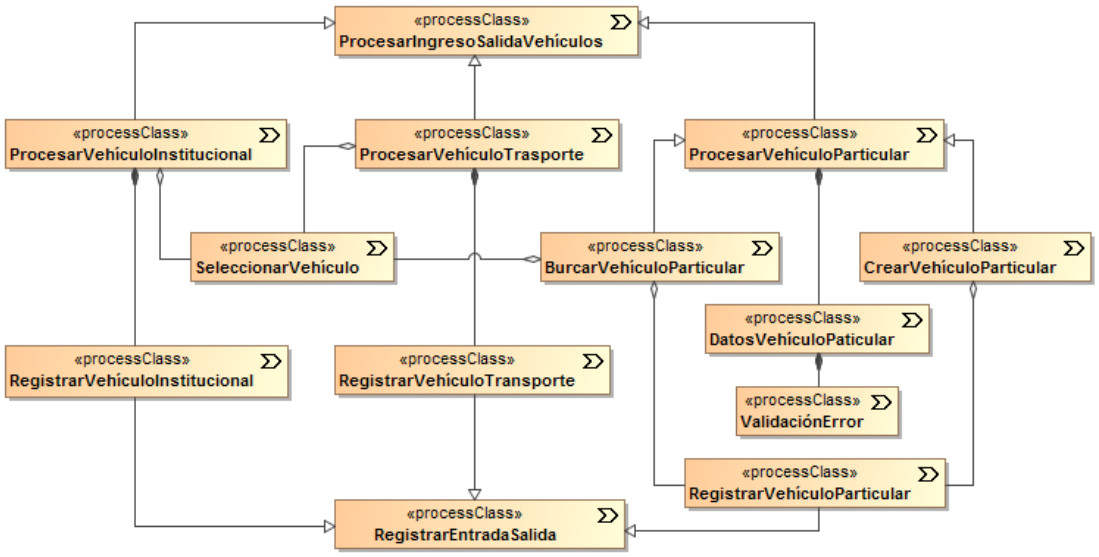


Figura 23. Representación gráfica de la estructura de proceso de registro de ingreso y salida de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

ProcesarVehículoInstitucional y **ProcesarVehículoTransporte** incluyen a la clase de proceso **SeleccionarVehículo**, permitiendo seleccionar un vehículo institucional o de transporte de manera respectiva. **ProcesarVehículoInstitucional** incluye a la clase de proceso **RegistrarVehículoInstitucional**, siendo esta misma un desglose de **RegistrarEntradaSalida**, la cual representa de manera general el registro de entrada y salida de vehículos en la institución. De igual manera **ProcesarVehículoTransporte** incluye a la clase de proceso **RegistrarVehículoTransporte**, la cual también es un desglose de **RegistrarEntradaSalida**.

La clase de proceso **ProcesarVehículoParticular** incluye a **DatosVehículoParticular** que describe la información propia de vehículos particulares que ingresan y salen de la institución, los cuales son utilizados si se registra un nuevo vehículo particular. Cabe mencionar que **DatosVehículoParticular** incluye a la clase de proceso **ValidaciónError**, donde se valida la información ingresada antes de ser almacenada. **ProcesarVehículoParticular** hereda sus características a **BuscarVehículoParticular** y **CrearVehículoParticular**.

La clase de proceso **BuscarVehículoParticular** incluye a **SeleccionarVehículo**, permitiendo seleccionar un vehículo particular registrado anteriormente. **BuscarVehículoParticular** y **CrearVehículoParticular** incluyen a la clase de proceso **RegistrarVehículoParticular**, la cual es otro desglose de **RegistrarEntradaSalida**.

Para hacer más énfasis en el modelo de proceso es importante definir el flujo de los procesos que se encuentran estructurados y documentados en los diagramas de estructura de proceso.

2.5.2. Diagramas de flujo de proceso.

Cabe recalcar que en este tipo de diagrama se describe el flujo de proceso que se lo realiza por medio de acciones del usuario o del sistema, las cuales se las diferencia por su notación <<**userAction**>> y <<**systemAction**>> descritas en la parte superior de cada elemento de acción.

2.5.2.1. Creación de vehículo.

La Figura 24 muestra el diagrama de flujo de proceso de creación de vehículo, en el cual se detalla el proceso de creación de un vehículo en la aplicación.

Diagrama de flujo de proceso de creación de vehículo

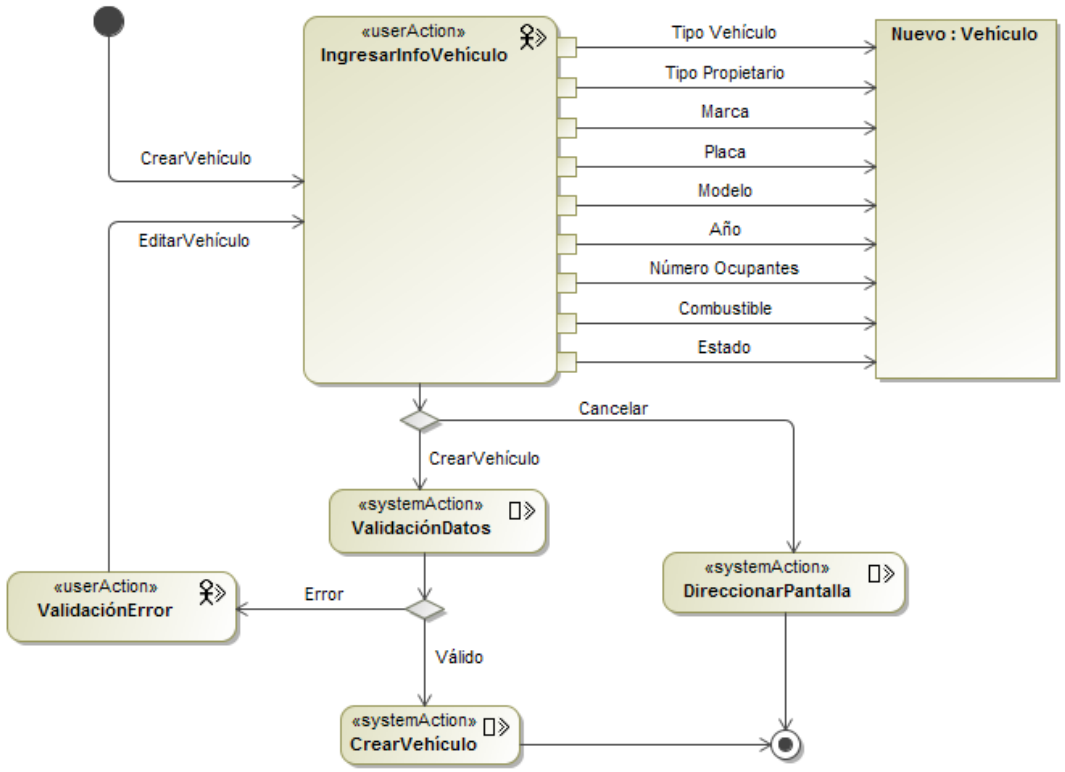


Figura 24. Representación gráfica que describe el flujo de proceso de creación de vehículo
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Se inicia el proceso con la acción de usuario **IngresarInfoVehículo** en donde se ingresa los datos que posee un vehículo como son Tipo Vehículo, Tipo Propietario, Marca, Placa, Modelo, Año, Número Ocupantes, Combustible y Estado. Posteriormente se decide si se va a crear un vehículo (**CrearVehículo**) o si se desea cancelar la tarea (**Cancelar**). En el caso de elegir la opción **Cancelar**, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo de proceso.

Una vez elegida la opción **CrearVehículo**, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **IngresarInfoVehículo** donde el flujo inicia nuevamente a partir de ese punto. Cuando los datos de entrada sean validados y no posean errores, el sistema se encargará de almacenarlos en el repositorio de la aplicación por medio de la acción **CrearVehículo**, la cual da por terminado el flujo de proceso.

2.5.2.2. Creación de solicitud.

En la Figura 25 se observa el diagrama de flujo de proceso de creación de solicitud, en el cual se detalla el proceso de creación de una solicitud de movilización en la aplicación.

Se inicia el proceso con la acción de usuario **IngresarInfoSolicitud** en donde se ingresa los datos que posee una solicitud de movilización como son Fecha y Hora Inicio, Fecha y Hora Fin, Destino, Comisión Servicio y Estado. Posteriormente se decide si se va a crear una solicitud (**CrearSolicitud**) o si se desea cancelar la tarea (**Cancelar**). En el caso de

elegir la opción **Cancelar**, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo de proceso.

Diagrama de flujo de proceso de creación de solicitud

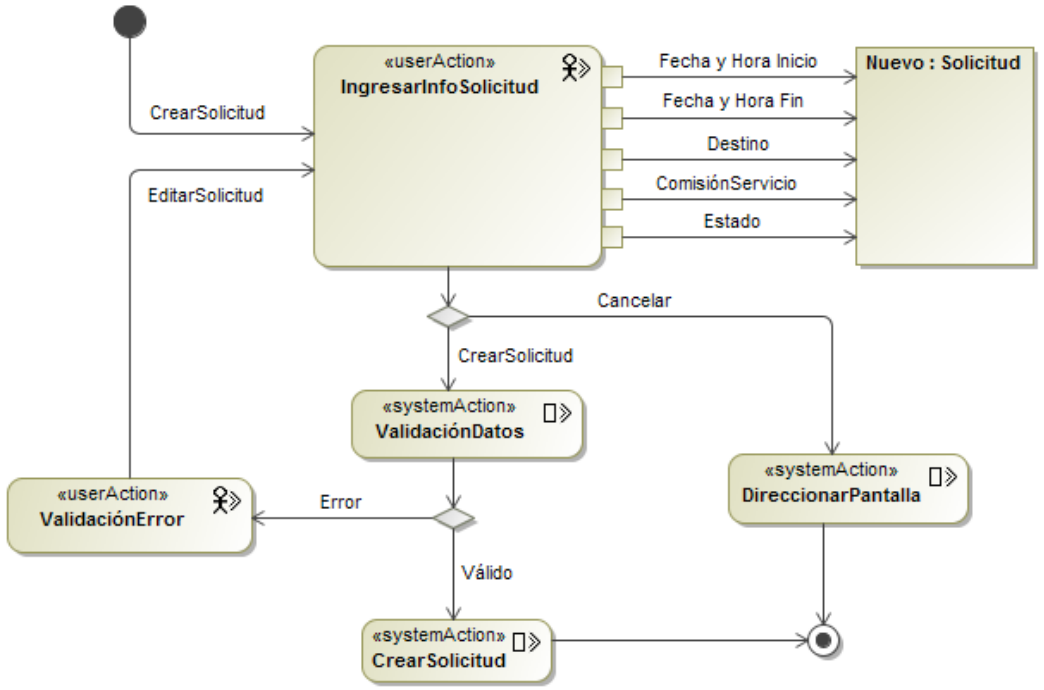


Figura 25. Representación gráfica que describe el flujo de proceso de creación de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Una vez elegida la opción **CrearSolicitud**, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **IngresarInfoSolicitud** donde el flujo inicia nuevamente a partir de ese punto. Cuando los datos de entrada sean validados y no posean errores, el sistema se encargará de almacenarlos en el repositorio de la aplicación por medio de la acción **CrearSolicitud**, la cual da por terminado el flujo de proceso.

2.5.2.3. Preautorización de solicitud.

En la Figura 26 se explica el diagrama de flujo de proceso de preautorización de solicitud detalla el proceso de preautorización de solitudes de movilización con comisión de servicio, el cual es realizado por el Administrador Financiero.

Diagrama de flujo de proceso de preautorización de solicitud

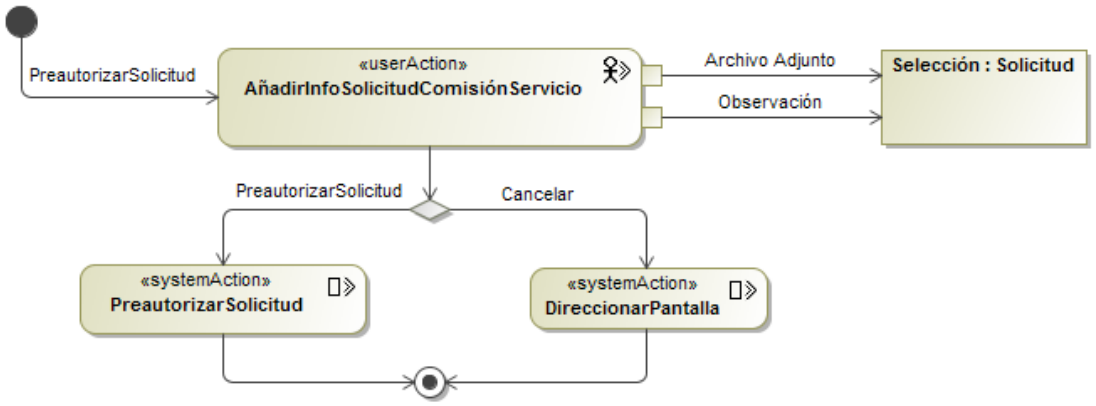


Figura 26. Representación gráfica que describe el flujo de proceso de preautorización de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.5.2.4. Autorización de solicitud.

En la Figura 27 se aprecia el diagrama de flujo de proceso de autorización de solicitud, el cual detalla el proceso de autorización de solitudes de movilización del personal de la institución que es realizado por el Administrador de Movilización.

Se inicia el proceso con la acción de usuario **AñadirInfoSolicitud** en donde se ingresa datos adicionales que permiten a una solicitud ser aprobada, que en este caso son Vehículo y Conductor. Si la solicitud de movilización es de comisión de servicio, el Administrador de Movilización puede adjuntar un archivo y una observación adicional a través de Archivo Adjunto y Observación. Posteriormente se decide si se va autorizar la solicitud (**AutorizarSolicitud**) o si se desea cancelar la tarea (**Cancelar**). En el caso de elegir la

opción **Cancelar**, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo de proceso. Una vez elegida la opción **AutorizarSolicitud**, el sistema se encarga de alterar el estado de la solicitud por medio de la acción **AutorizarSolicitud**, la cual da por terminado el flujo de proceso.

Diagrama de flujo de proceso de autorización de solicitud

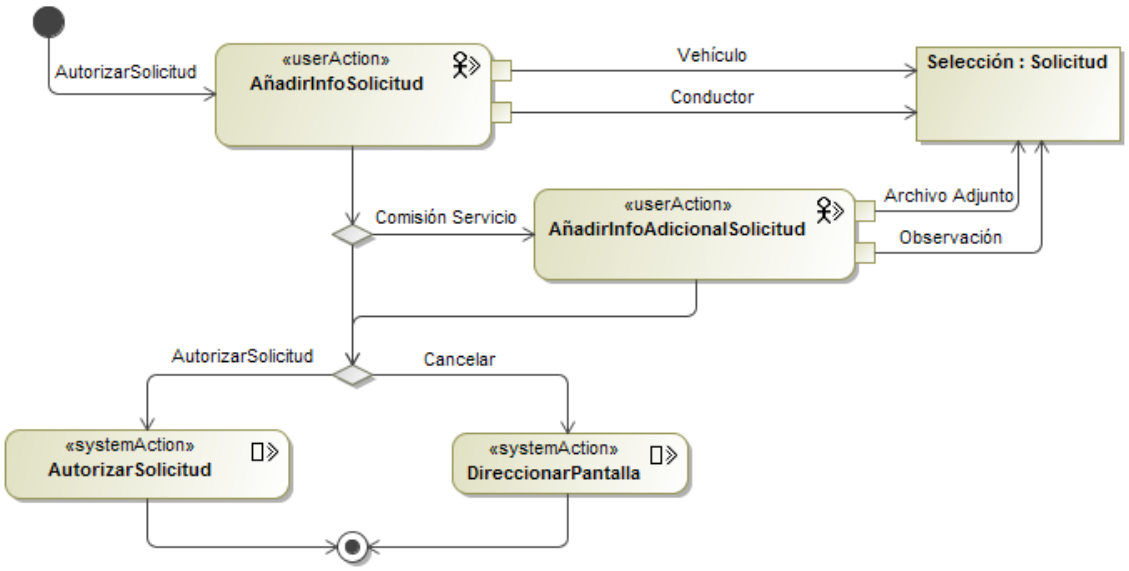


Figura 27. Representación gráfica que describe el flujo de proceso de autorización de solicitud
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

2.5.2.5. Registro de recorrido de vehículo institucional.

En la Figura 28 se expone el diagrama de flujo de proceso de registro de recorrido de vehículo institucional, el cual detalla el proceso de registro de vehículos institucionales desde el momento que salen de la institución hasta su retorno.

Se inicia el proceso con la acción de usuario **AñadirInfoRecorrido** en donde se ingresa los datos que forman parte de un recorrido como lo son Vehículo Institucional, Conductor, Usuario, Kilometraje Inicial, Kilometraje Final y Observación. Si un vehículo está

autorizado para movilizarse, solamente debe ingresarse el Kilometraje Inicial, Kilometraje Final y Observación; caso contrario se debe especificar adicionalmente el vehículo, el conductor y el usuario que requiere movilización. Posteriormente se decide si se va registrar el recorrido (**RegistrarRecorrido**) o si se desea cancelar la tarea (**Cancelar**). En el caso de elegir la opción **Cancelar**, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo de proceso.

Diagrama de flujo de proceso de registro de recorrido de vehículo institucional

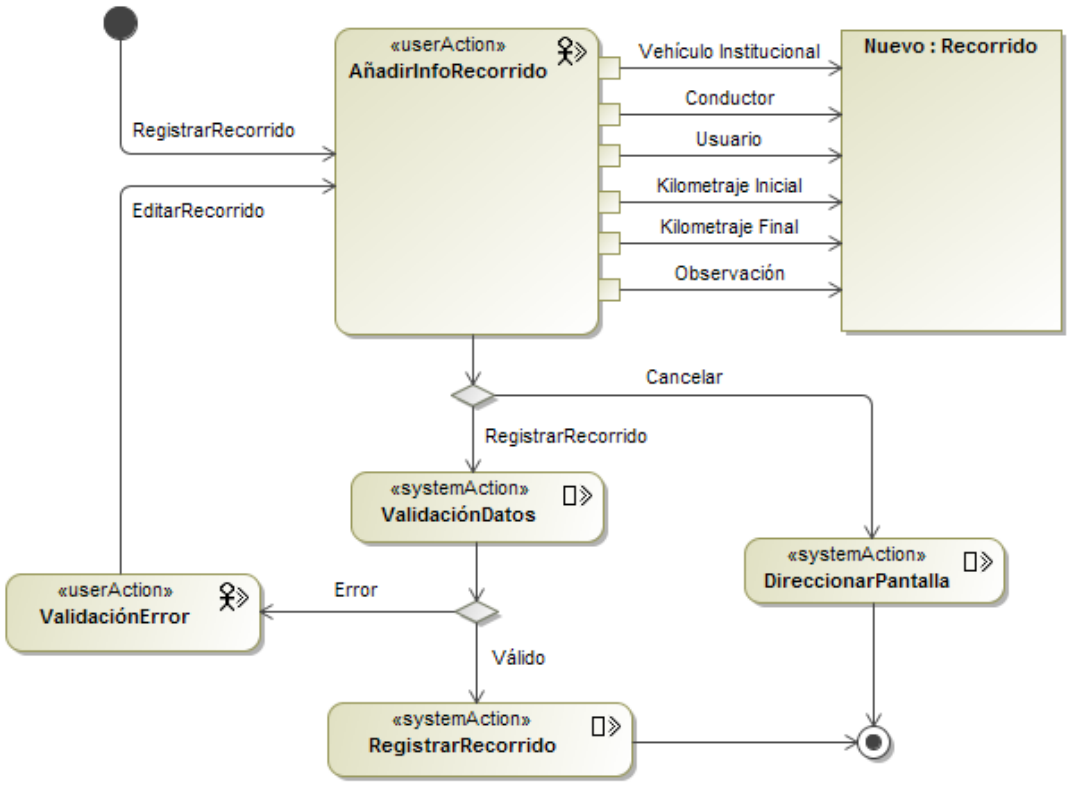


Figura 28. Representación gráfica que describe el flujo de proceso de registro de recorrido de vehículo institucional
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Una vez elegida la opción **RegistrarRecorrido**, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la

información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **AñadirInfoRecorrido** donde el flujo se inicia nuevamente a partir de ese punto.

Cuando los datos de entrada sean validados y no posean errores, el sistema se encargará de almacenarlos en el repositorio de la aplicación por medio de la acción **RegistrarRecorrido**, la cual da por terminado el flujo de proceso.

2.5.2.6. Registro de arribo de vehículo de transporte.

La Figura 29 muestra el diagrama de flujo de proceso de registro de arribo de vehículo de transporte, el cual detalla el proceso de registro de arribo de vehículos de transporte de los empleados de la institución.

Se inicia el proceso con la acción de usuario **AñadirInfoArribo** en donde se ingresa los datos que forman parte de un arribo como lo son Vehículo de Transporte, Fecha y Hora Arribo, y Observación. Posteriormente se decide si se va registrar el arribo (**RegistrarArribo**) o si se desea cancelar la tarea (**Cancelar**). En el caso de elegir la opción Cancelar, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo de proceso.

Una vez elegida la opción **RegistrarArribo**, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **AñadirInfoArribo** donde el flujo inicia nuevamente a partir de ese punto.

Diagrama de flujo de proceso de registro de arribo de vehículo de transporte

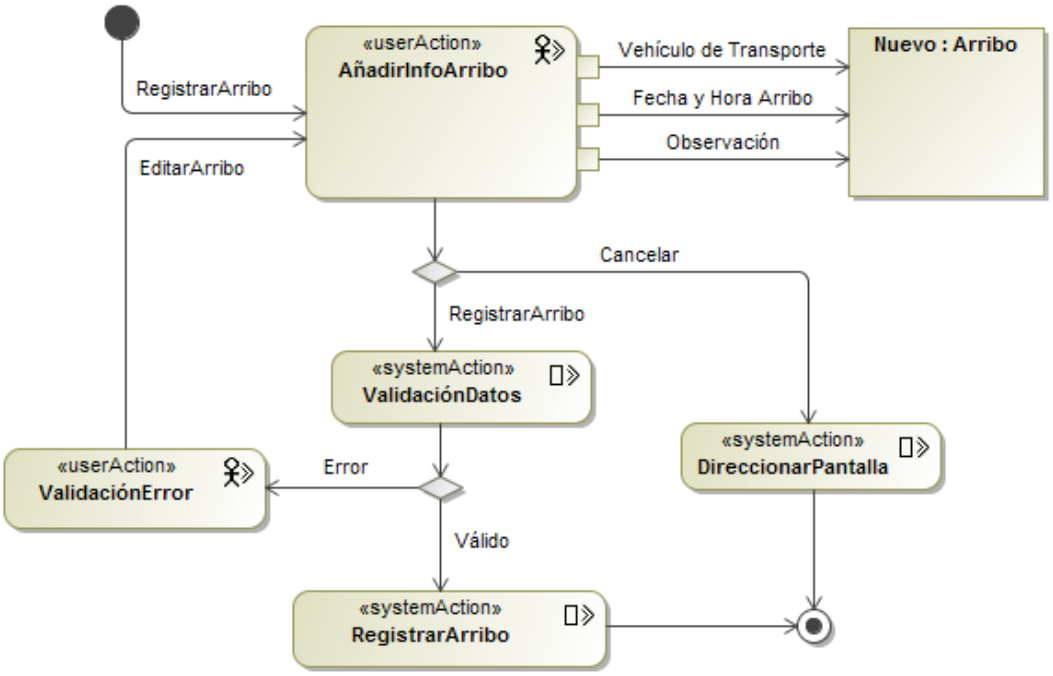


Figura 29. Representación gráfica que describe el flujo de proceso de registro de arribo de vehículo de transporte
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Cuando los datos de entrada sean validados y no posean errores, el sistema se encargará de almacenarlos en el repositorio de la aplicación por medio de la acción **RegistrarArribo**, la cual da por terminado el flujo de proceso.

2.5.2.7. Registro de historial de vehículo particular.

En la figura 30 se puede apreciar cómo se inicia el proceso con la acción de usuario **BuscarVehículoParticular** donde se ingresa el número de placa (**Placa**) para verificar si en los registros del sistema existe el vehículo buscado.

En el caso de no existir un vehículo se lo crea mediante **AñadirInfoVehículoParticular** donde se ingresa los datos que forman parte de un vehículo particular como lo son Nombre Conductor, Cédula Conductor, Placa, Marca, Modelo, Año, Número Ocupantes,

Combustible, Tipo Vehículo y Tipo Propietario. Posteriormente se decide si se va crear el nuevo vehículo (**CrearVehículoParticular**) o si se desea cancelar la tarea (**Cancelar**).

Diagrama de flujo de proceso de registro de historial de vehículo particular

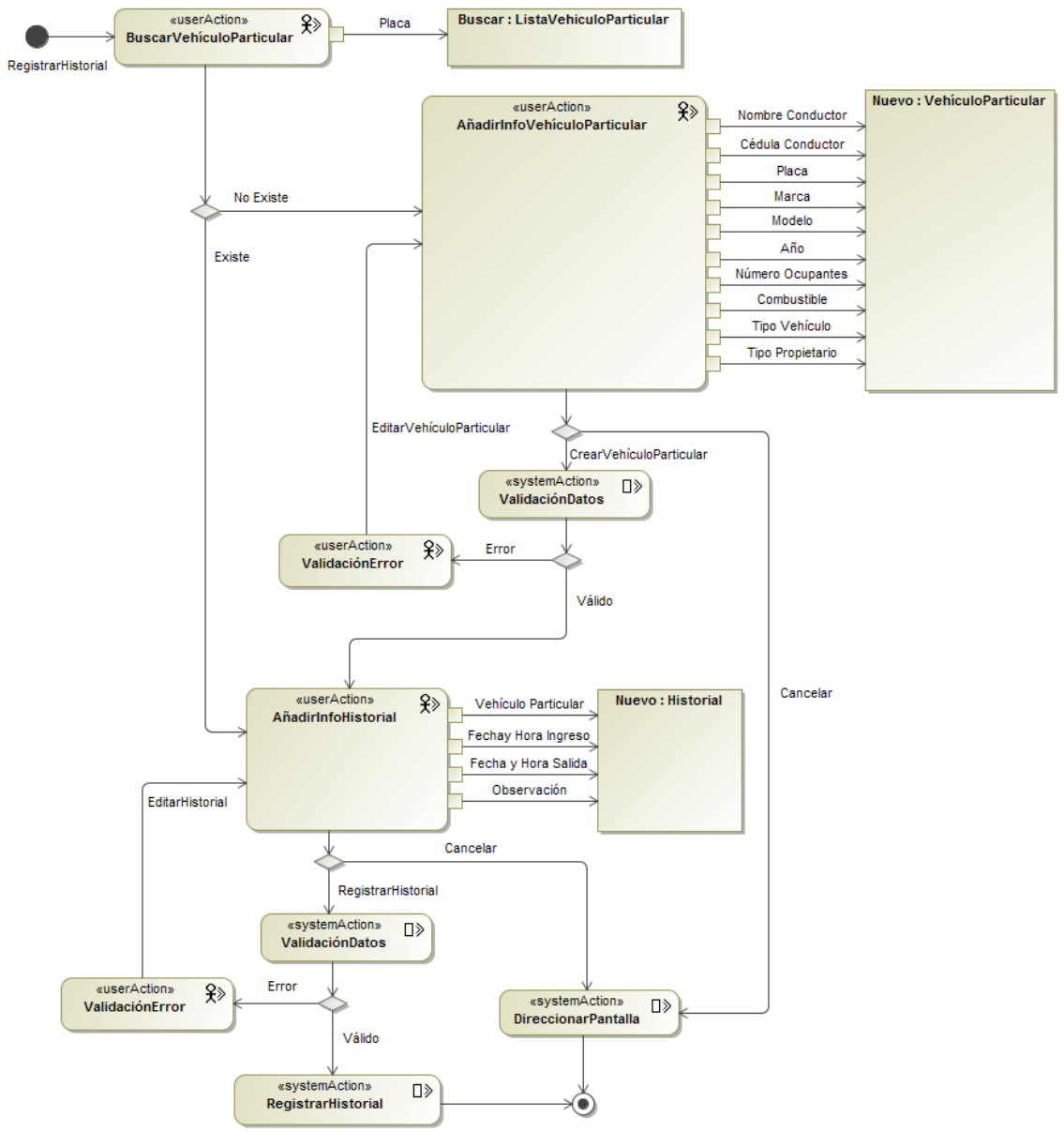


Figura 30. Representación gráfica que describe el flujo de proceso de registro de historial de vehículo particular
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

En el caso de elegir la opción Cancelar, el sistema realiza la acción **DireccionarPantalla**, la cual direcciona al usuario a otra página de la aplicación, dando por terminado el flujo

de proceso. Este subproceso se lo realiza de igual manera para las demás opciones de Cancelar que aparezcan durante el flujo.

Una vez elegida la opción **CrearVehículoParticular**, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **AñadirInfoVehículoParticular** donde el flujo se inicia nuevamente a partir de ese punto.

Cuando los datos de entrada de vehículo particular sean validados y no posean errores se procede a ingresar los datos del historial en la acción **AñadirInfoHistorial**, los cuales son Vehículo Particular, Fecha y Hora Ingreso, Fecha y Hora Salida y Observación. Cabe mencionar que en el caso de existir un vehículo particular registrado en el sistema, el flujo debe iniciar desde la acción de usuario **AñadirInfoHistorial**.

Continuando con el flujo se debe decidir si se va registrar el historial (**RegistrarHistorial**) o si se desea cancelar la tarea (**Cancelar**). Posteriormente de ser elegida la opción RegistrarHistorial, el sistema valida los datos de entrada a través de la acción **ValidaciónDatos**. Por medio de **Error** y **Válido** se decide si la información ingresada es correcta. En el caso de **Error** el usuario debe verificar nuevamente los datos de entrada por medio de la acción **ValidaciónError**, la cual direcciona a **AñadirInfoHistorial** donde el flujo se inicia nuevamente a partir de ese punto. Cuando los datos de entrada del historial sean validados y no posean errores, el sistema se encargará de almacenarlos en el repositorio de la aplicación por medio de la acción **RegistrarHistorial**, la cual da por terminado el flujo de proceso.

CAPÍTULO 3

CONSTRUCCIÓN Y PRUEBAS

En este capítulo se detalla las herramientas y recursos necesarios para realizar los procesos que forman parte de las fases de construcción y pruebas en el desarrollo de software.

Cabe mencionar que UWE es de gran utilidad para el análisis y diseño del software ya que en esta sección se recopila esa información con la finalidad de construir módulos y probarlos. Además de UWE se implantó la metodología XP donde se puede apreciar de mejor manera durante la construcción y pruebas, ya que cada entregable es una iteración la cual es probada y entregada al usuario siempre y cuando se cumpla con los requisitos planteados, caso contrario la iteración debe repetirse.

Metodología UWE y XP

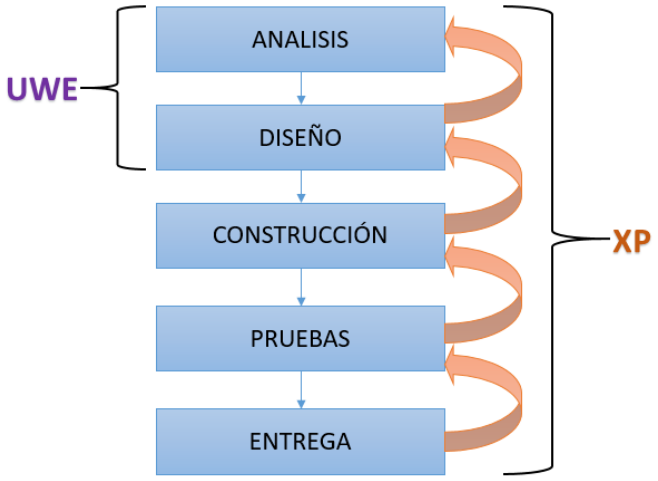


Figura 31. Relación entre metodología UWE y XP
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

En la figura 31 se puede apreciar cómo afecta los procesos iterativos de la metodología XP en las fases de análisis y diseño que son enfocadas más por UWE, es decir que una iteración interviene a todo el ciclo de desarrollo en general.

3.1. Herramientas

Las herramientas que se utilizaron para construir el software se mencionan a continuación.

- Lenguaje de programación C#
- Framework 3.5
- Motor de base de datos SQL Server 2008 R2
- IDE de desarrollo Visual Studio 2008 (construcción de aplicación web)
- IDE de desarrollo Visual Studio 2015 (construcción de aplicación móvil)
- Reporting Service 2008 (implementación de reportes)

3.2. Diagrama de base de datos

En la figura 32 se presenta el diagrama de la base de datos, la cual describe las tablas que almacenará la información de la aplicación.

La base de datos se estructuró de acuerdo a la funcionalidad del sistema como tal. Las tablas de mayor relevancia son VEHICULO_TBL, SOLICITUD_TBL y ASIGNA_VEHICULO_TBL ya que en estas se reflejan la funcionalidad del sistema y almacenan la mayor parte de las transacciones realizadas.

Si se desea analizar a fondo cada tabla de la base de datos, se puede revisar el **Anexo 2** en el cual se encuentra el diccionario de datos que contiene la descripción de las tablas y sus respectivos campos detallando sus características.

Diagrama de bases de datos

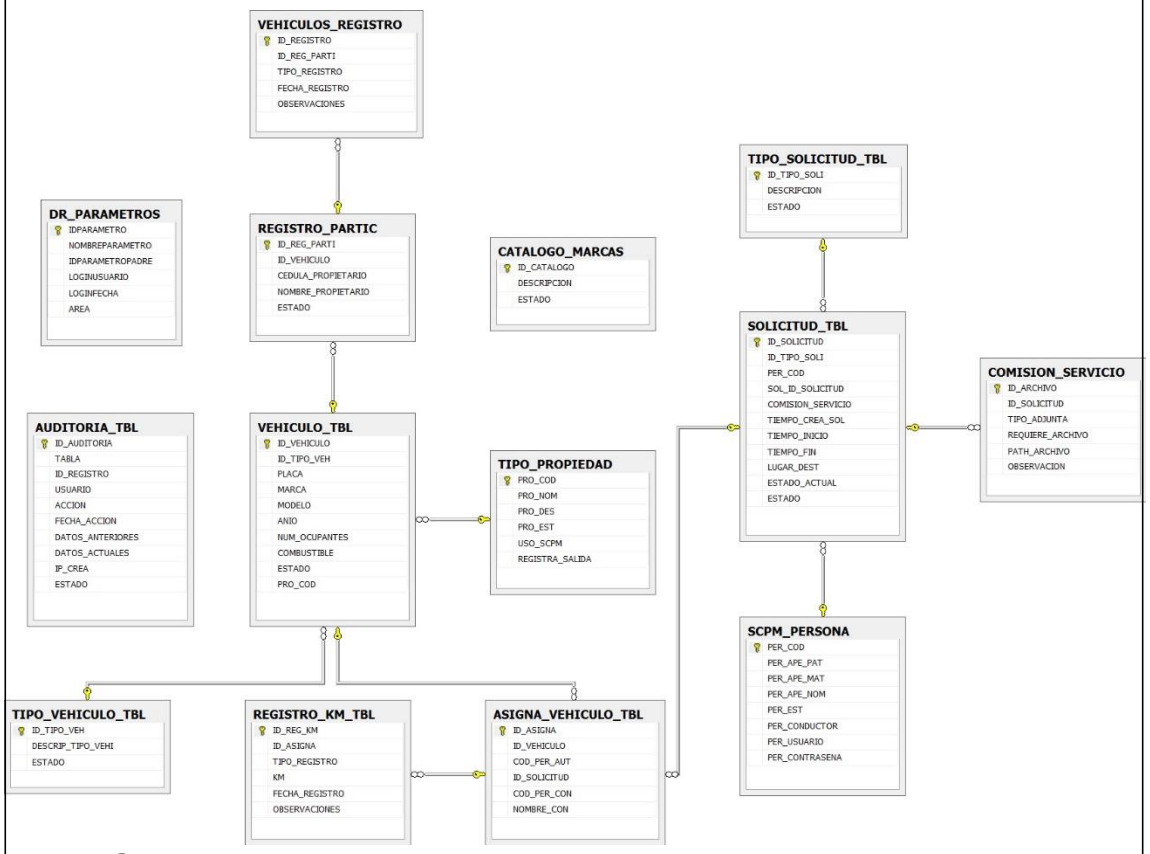


Figura 32. Diagrama de base de datos de la aplicación
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.3. Construcción MVC

Como se mencionó en el **Capítulo 1** en la sección **Marco Teórico**, el Modelo Vista Controlador (MVC) es una de las líneas teóricas fundamentales para la construcción de este tipo de software.

MVC define un modelo el cual divide al software en tres partes principales como son el Modelo, las Vistas y el Controlador, que respectivamente se relación con las capas de datos, presentación y negocio.

3.3.1. Capa de datos.

Esta capa define el Modelo de MVC el cual se relaciona con los datos que serán almacenados y extraídos del repositorio del aplicativo.

En la construcción de la aplicación web por medio del IDE planteado se estableció una solución que consta de 2 proyectos, los cuales se muestran en la figura 33.

Solución creada en IDE

Figura 33. Solución de 2 proyectos para la construcción de la aplicación web
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El proyecto **Base** contiene parte del código que permite ingresar, modificar y extraer información desde la aplicación. Como se muestra en la figura 34 el proyecto **Base** contiene diferentes archivos de tipo **.CS** los cuales contienen codificación que permite a la aplicación comunicarse con procedimientos almacenados que interactúan directamente con la base de datos.

Proyecto Base

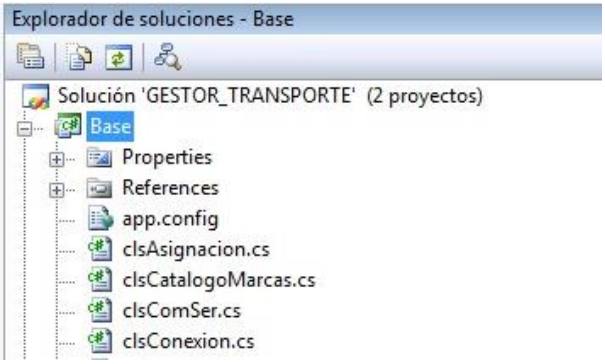


Figura 34. Estructura del proyecto Base
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.3.2. Capa de presentación.

Esta capa representa las Vistas en MVC, las cuales definen las interfaces de la aplicación. El proyecto **FrontEnd** contiene las interfaces de usuario que forman parte de la capa de presentación. En la figura 35 se puede apreciar los diferentes tipos de archivos que forman parte del proyecto entre los cuales se mencionan los más relevantes.

Directorio Css: contiene archivos de estilos que son parte del diseño de los formularios aspx.

Directorio Images: contiene imágenes que son parte del diseño de los formularios aspx.

Archivos de tipos aspx: son formulario de contenido web para usuario final.

Archivos de tipo master: páginas maestras que permiten heredar sus propiedades y diseño a formularios aspx.

Proyecto FontEnd

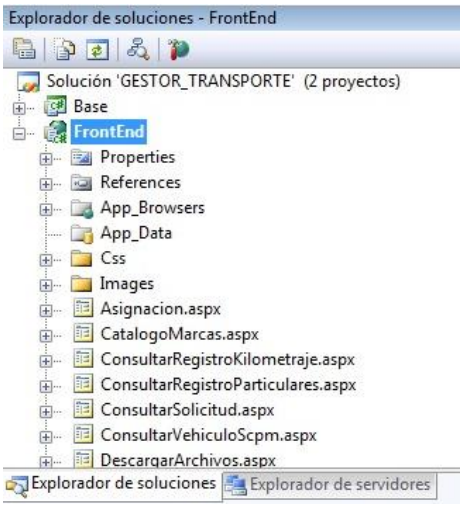


Figura 35. Estructura del proyecto FrontEnd
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.3.3. Capa de negocio.

La capa de negocio define las reglas de negocio de la cual está formada la aplicación. Esta capa contiene prácticamente la mayor parte del código generado. Cabe mencionar que la POO (Programación Orientada a Objetos) fue de gran utilidad para la construcción debido a que sus propiedades pueden ser utilizadas con normalidad mediante el lenguaje de programación C#.

La relación entre la capa de negocio y MVC es el Controlador, el cual interactúa entre el Modelo y las Vistas, dicho en otras palabras la capa de negocio trabaja entre las capas de datos y presentación.

En el proyecto **FrontEnd** a cada formulario aspx se ha relacionado 2 tipos de archivo cs que contiene código fuente que describe las diferentes restricciones y validaciones de la aplicación, como la interacción entre los componentes de contenido web.

Se ha tomado como ejemplo en la figura 36 al formulario *Asignacion.aspx* donde se puede apreciar la capa de negocio por medio de los archivos *Asignación.aspx.cs* y *Asignacion.aspx.designer.cs* respectivamente.

Capa de negocio

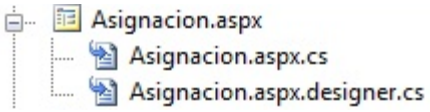


Figura 36. Archivos cs que forman parte de la capa de negocio
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Por medio *Asignacion.aspx.designer.cs* se establece la relación entre *Asignación.aspx.cs* y los elementos del formulario *Asignacion.aspx*. Los elementos relacionados pueden ser

cajas de texto, botones, paneles, etc; los cuales pueden manipulados en *Asignación.aspx.cs* con normalidad.

En el archivo *Asignación.aspx.cs* se define las reglas de negocio como lo son las distintas validaciones y restricciones que puede poseer el formulario *Asignación.aspx*.

Para interactuar con la capa de negocio es necesario incluir al proyecto **Base** en el archivo *<NombreFormulario>.aspx.cs* por medio de la siguiente línea de código: `using Base`.

Una vez incluido el proyecto **Base** es posible crear objetos de clases públicas y también utilizar los métodos y variables públicos a través de los objetos creados. En **Capitulo 2** en la sesión Modelo de Contenido se puede apreciar el diagrama de clases en el cual se establece la comunicación entre las clases de las capas de datos y negocio.

3.4. Construcción de aplicación móvil

La finalidad de construir una aplicación móvil que forme parte de una solución informática integrada, nace a partir de los módulos que serán utilizados por el personal de seguridad de la SCPM, debido a que dicho personal no utilizará un computador, es decir que harán uso de dispositivos móviles como tablets. Los módulos que forman parte de la aplicación móvil son el ingreso y salida de vehículos de uso de la SCPM y vehículos ajenos a la institución.

Se optó por construir la aplicación móvil en Android por medio de la herramienta Visual Studio 2015, la cual brinda varios beneficios para este tipo de aplicación. Visual Studio 2015 integra varias funcionalidades de la plataforma Xamarin, la cual permite crear aplicaciones móviles multiplataforma.

3.4.1. Proyecto WebViewApp.

Una vez instalado Visual Studio 2015 se procede a crear un proyecto Android de tipo WebViewApp. Este tipo de proyecto permite crear una aplicación móvil para Android, la cual podrá acceder a una página web específica de la aplicación web descrita en la sección anterior.

Proyecto APP

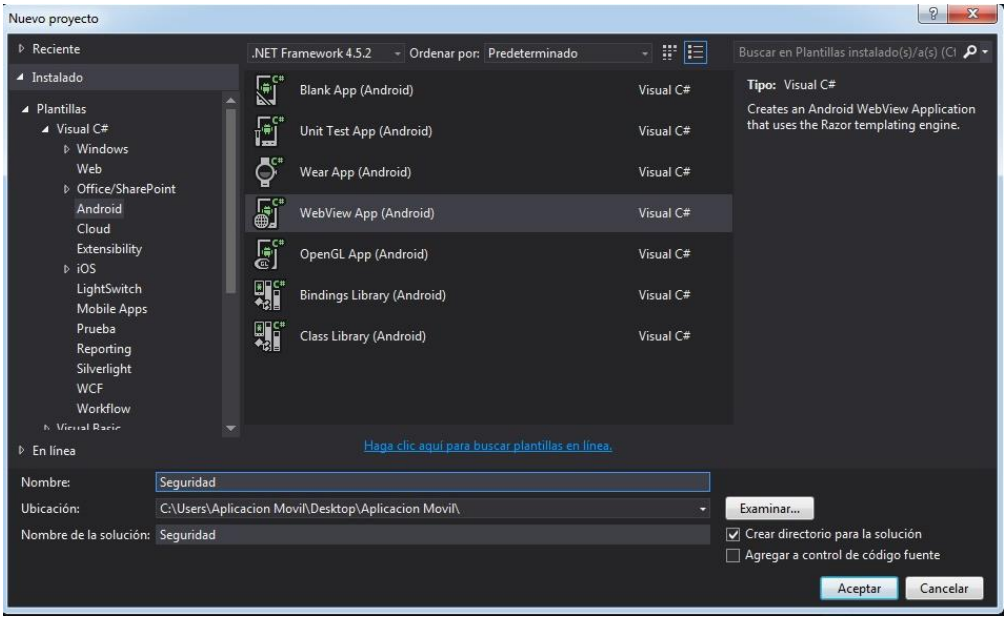


Figura 37. Creación de Proyecto Android WebViewApp
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

En la figura 38 se aprecia el código fuente del archivo Main.axml donde se define el alto y ancho de la interfaz y de la vista de contenido web WebView. Entre lo más importante se puede apreciar el id de la WebView el cual será utilizada por el archivo *MainActivity.cs* más adelante.

Archivo Main.axml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:minWidth="25px"
    android:minHeight="25px">
    <WebView
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:id="@+id/webView" />
</LinearLayout>
```

Figura 38. Código fuente del archivo Main.axml
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El archivo *MainActivity.cs* contiene código C# el cual será embebido en la aplicación móvil, el cual se muestra a continuación.

Archivo MainActivity.cs

```
namespace Seguridad
{
    [Activity(Label = "Seguridad", MainLauncher = true)]
    public class MainActivity : Activity
    {
        private WebView web_view;
        protected override void OnCreate(Bundle bundle)
        {
            base.OnCreate(bundle);

            SetContentView(Resource.Layout.Main);

            web_view = FindViewById<WebView>(Resource.Id.webView);
            web_view.Settings.JavaScriptEnabled = true;

            web_view.LoadUrl("http://<DireccionIPServidor>/Transporte/Requerimi
            entos/MenuGuardia.aspx");

            web_view.SetWebViewClient(new HelloWebViewClient());

        }
    }
}
```

Figura 39. Código fuente del archivo MainActivity.cs
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

La página web que se debe mostrar se la define a través del método **web_view.LoadUrl** donde se define el formulario *MenuGuardia.aspx* que es exclusivo para la aplicación móvil.

Antes de compilar el proyecto se debe establecer el modo Release para poder generar un archivo de tipo apk el cual será instalado en el dispositivo móvil.

Modo Release

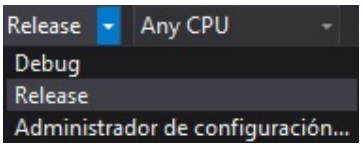


Figura 40. Modo Release previo a la compilación del proyecto
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

También se debe definir la versión mínima de Android la cual se ejecutara la aplicación móvil que en este caso es **Android_4.1**.

Versión Android

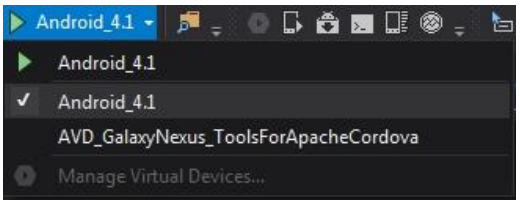


Figura 41. Definición de versión de Android previo a la compilación del proyecto
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Una vez definido el modo Release y la versión de Android se procede a compilar el proyecto, cuyo resultado se ve reflejado en el directorio bin\Debug de la raíz del árbol de directorio el cual puede ser verificado por medio del explorador de Windows como se muestra en la figura 42.

Archivos generados

Mono.Android.dll.mdb	01/10/2015 17:25	Archivo MDB	6.012 KB
mscorlib.dll.mdb	01/10/2015 17:25	Archivo MDB	1.349 KB
Seguridad.dll	01/10/2015 17:29	Extensión de la apl...	12 KB
Seguridad.dll.mdb	01/10/2015 17:29	Archivo MDB	3 KB
Seguridad	01/10/2015 17:29	Program Debug D...	28 KB
Seguridad.Seguridad.apk	01/10/2015 17:30	Archivo APK	5.561 KB
Seguridad.Seguridad-Signed.apk	01/10/2015 17:31	Archivo APK	5.490 KB
System.Core.dll.mdb	01/10/2015 17:25	Archivo MDB	353 KB
System.dll.mdb	01/10/2015 17:25	Archivo MDB	686 KB
System.Web.Services.dll.mdb	01/10/2015 17:25	Archivo MDB	102 KB
System.Xml.dll.mdb	01/10/2015 17:25	Archivo MDB	525 KB
System.Xml.Linq.dll.mdb	01/10/2015 17:25	Archivo MDB	42 KB

Figura 42. Generación de archivos en modo Release después de la compilación del proyecto
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El archivo *Seguridad.Seguridad-Signed.apk* que se genera después de compilar el proyecto es el instalador de la aplicación que debe ser almacenado en el dispositivo móvil que se requiera instalar. Una vez instalada la aplicación móvil se procede a ejecutarla como cualquier aplicativo de uso común tal como se muestra en la figura 43.

Aplicación móvil

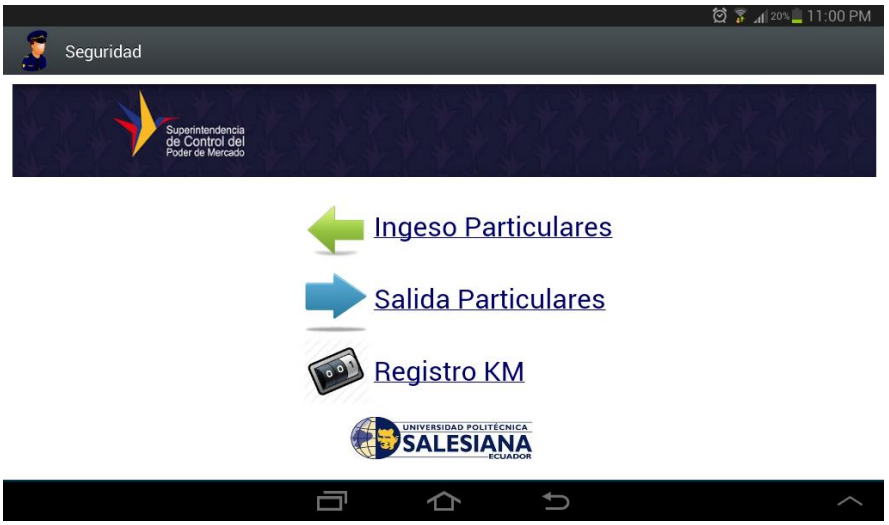


Figura 43. Aplicación móvil en funcionamiento
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.5. Construcción de reportes

Los reportes son parte fundamental de un sistema informático debido a la gran información que puede proporcionar a un usuario final. A través de los reportes se refleja los registros que son alimentados por un aplicativo, los cuales se encuentran almacenados en una base de datos.

Reporting Service 2008 es la herramienta que se ha escogido para administrar los reportes que son publicados en un servidor de informes de entorno web. Cabe mencionar que el editor de reportes que se utilizó es SQL Server Business Intelligence Development Studio, el cual permite crear informes y publicarlos en un servidor Reporting Service.

SQL Server Business Intelligence Development Studio debe ser ejecutado como administrador con la finalidad de poder publicar sin restricción los informes generados. Una vez ejecutada el editor se procede a crear un nuevo proyecto de servidor de informes como se muestra en la figura 44.

Proyecto de servidor de informes

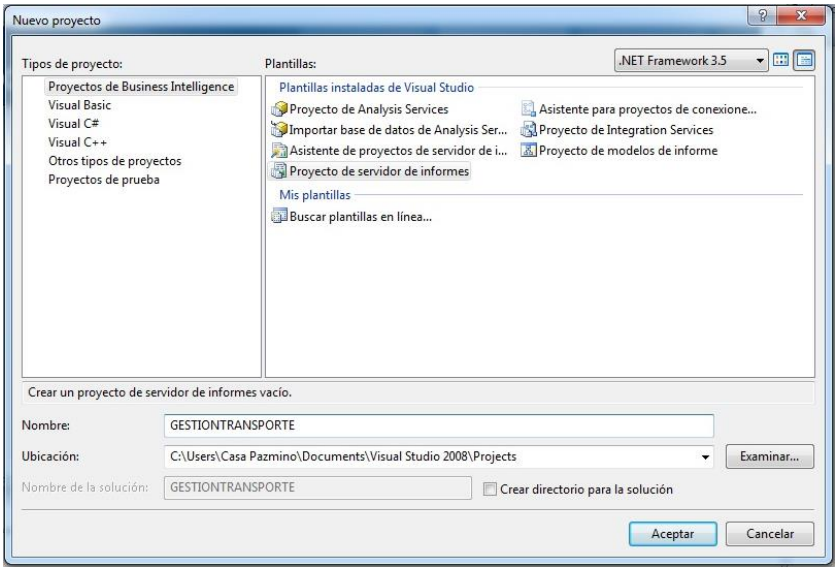


Figura 44. Creación de proyecto de servidor de informes
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Cuando se crea este tipo de proyecto se generan 3 directorios principales. El directorio *Informes* es donde se almacenaran los reportes o informes que serán editados y publicados.

Creación de Informes

Figura 45. Visualización de informes creados desde el IDE de desarrollo
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Como se muestra en la figura 45 se ha creado 4 informes los cuales se detallan en la siguiente tabla.

Tabla 11.
Informes generados

Informe	**Filtros de búsqueda**	**Descripción**
ReporteVehiculos	Placa, marca, modelo, combustible y estado.	Muestra los vehículos que han sido ingresados en el sistema.
ReporteSolicitudes	ID de solicitud, fecha de inicio, fecha de finalización, estado de solicitud, placa de vehículo asignado, nombre de conductor asignado y comisión de servicio.	Muestra las solicitudes que han sido ingresadas en el sistema.
ReporteVehiculosParticulares	Placa, cedula de conductor, fecha de entrada y fecha de salida	Muestra los vehículos ajenos a la institución que han ingresado a la misma.
ReporteRegistroKM	ID de solicitud, placa de vehículo, fecha de salida y fecha de retorno.	Muestra el registro de kilometraje de los vehículos usados por la institución.

Nota. Descripción de los informes generados
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.6. Implementación

3.6.1. Implementación de la aplicación web en Internet Infomation Services

En el servidor de aplicaciones IIS (Internet Information Services) propio de Microsoft, inicialmente se debe copiar los proyectos **Base** y **FrontEnd** dentro del directorio **C:\inetpub\wwwroot\Transporte**. Desde el administrador del IIS se procede puede observar los proyectos incorporados en el servidor, tal como se muestra en la siguiente figura.

Administrador de IIS

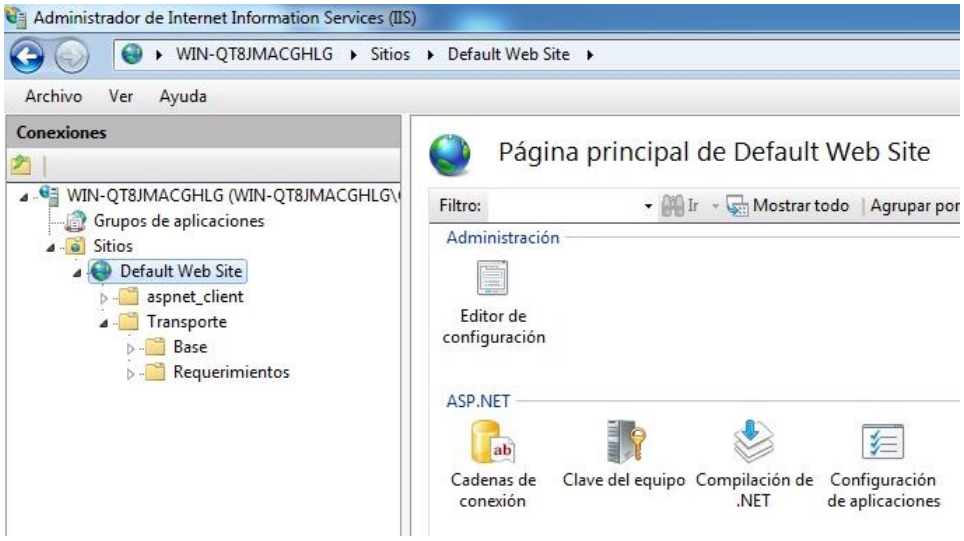


Figura 46. Proyecto reflejados dentro en IIS
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Después se debe convertir al proyecto **FrontEnd** en una aplicación web la cual contendrá el contenido que visualizará el usuario desde un navegador. En este caso como se muestra en la figura 47 aparece con el nombre **Requerimientos**.

Nueva aplicación web

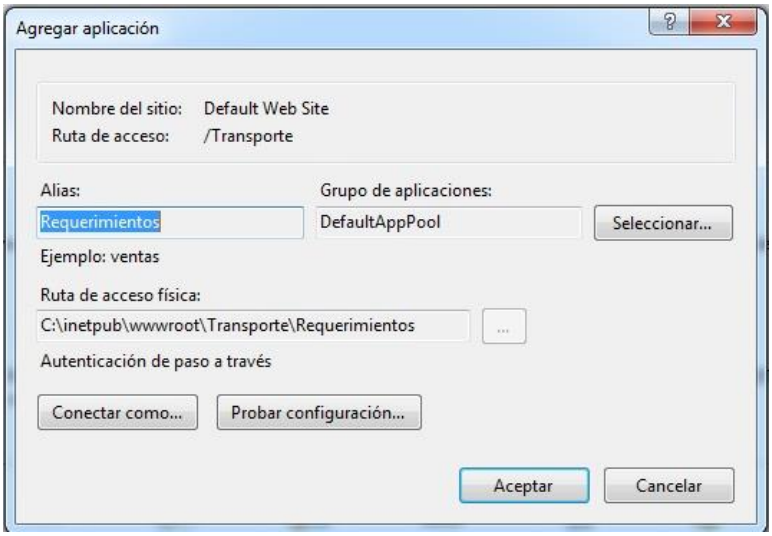


Figura 47. Agregación del proyecto como aplicación web
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

En la figura 48 se puede apreciar el árbol del servidor y como se ha creado una nueva aplicación web dentro de IIS.

Árbol de IIS

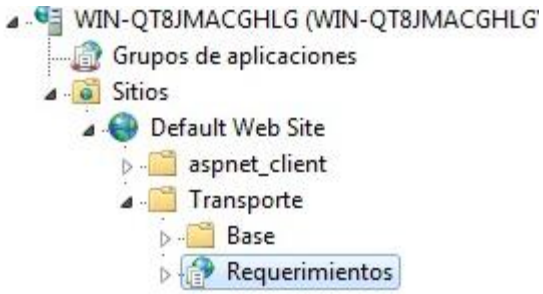


Figura 48. Desglose del árbol de directorio de IIS
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Finalmente desde otro ordenador se procede a abrir la aplicación web para constatar que se ha implementado completamente tal como se muestra en la figura 49.

Aplicación web desde el navegador

Figura 49. Aplicación web implementada completamente
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.6.2. Implementación de los reportes en Reporting Services

El Administrador de configuración de Reporting Services permite configurar el servidor de informes y sus características como cuentas de servicio, dirección URL de servicio web, base de datos, dirección URL del Administrador de informes, correo electrónico, claves de cifrado, entre otros.

Configuración del servidor Reporting Services

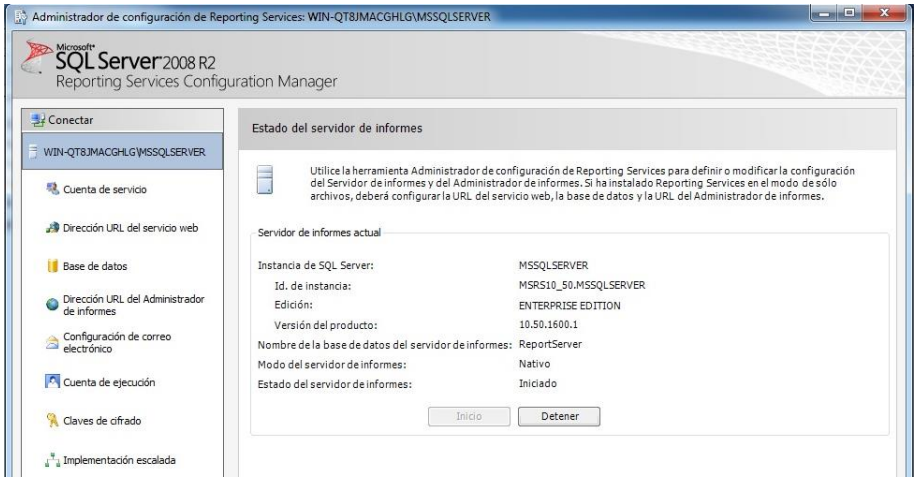


Figura 50. Pantalla de Administrador de configuración de Reporting Services
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Para poder acceder al servidor de informes se debe abrir una nueva página desde un navegador con la siguiente dirección **http://localhostReports/**. El siguiente paso es publicar los informes generados especificando la dirección del servidor en el proyecto de informes tal como se muestra en la figura 51.

URL del servidor de informes

Figura 51. Configuración de dirección URL del servidor de informes dentro del proyecto
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Finalmente se debe implementar el proyecto, lo cual permite publicar en el servidor todos los informes generados.

Publicación de informes

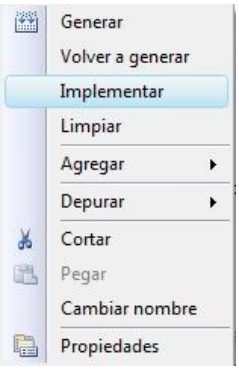


Figura 52. Implementación de los informes en el servidor
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Para poder visualizar los reportes se debe abrir un navegador con la dirección URL del servidor de informes tal como se muestra en las siguientes figuras.

Informes publicados

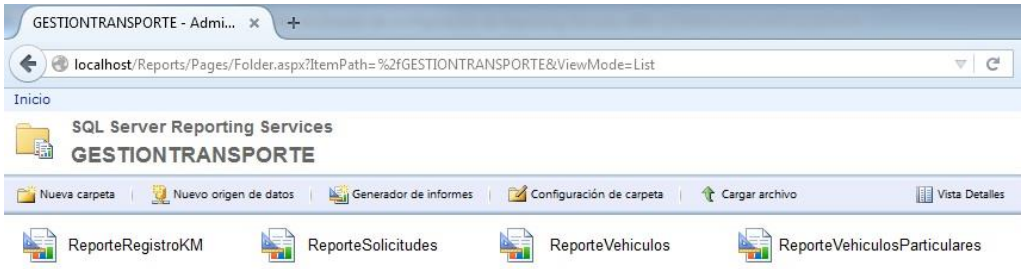
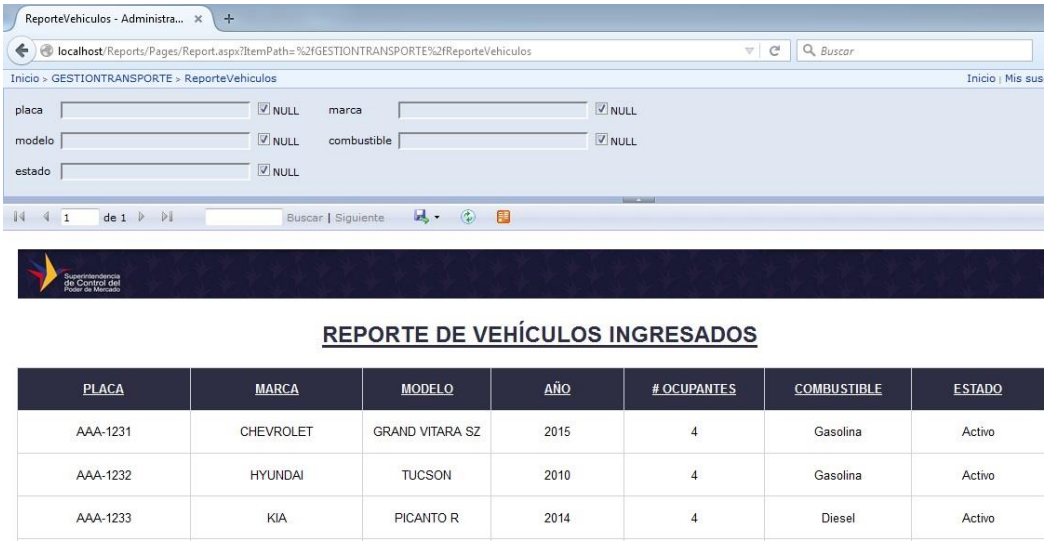


Figura 53. Informes publicados en el servidor
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Visualización de reportes



PLACA	MARCA	MODELO	AÑO	# OCUPANTES	COMBUSTIBLE	ESTADO
AAA-1231	CHEVROLET	GRAND VITARA SZ	2015	4	Gasolina	Activo
AAA-1232	HYUNDAI	TUCSON	2010	4	Gasolina	Activo
AAA-1233	KIA	PICANTO R	2014	4	Diesel	Activo

Figura 54. Visualización del reporte de vehículos
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

3.7. Pruebas

Esta sesión describe las pruebas que se ha realizado al sistema una vez finalizada la fase de construcción. Los tipos de pruebas que se ejecutaron son las siguientes: pruebas de unidad, de rendimiento, de carga y estrés.

3.7.1. Pruebas de unidad.

Este tipo de pruebas tienen como objetivo verificar la funcionalidad y estructura de cada componente de forma individual una vez que ha sido codificado.

En la siguiente tabla se presenta de forma general el resultado de las pruebas de unidad del sistema desarrollado.

Tabla 12.
Pruebas de unidad

N°	Funcionalidad	Número de Pruebas Realizadas	Porcentaje de pruebas exitosas
1	Gestión tipo de vehículo	4	100%
2	Gestión tipo de propietario	4	100%
3	Gestión marcas de vehículo	4	100%
4	Gestión vehículos	5	100%
5	Consulta de vehículos particulares	2	100%
6	Consulta historial de kilometraje	2	100%
7	Creación de solicitud	1	100%
8	Consulta de solicitudes	7	100%
9	Consulta solicitudes pendientes con comisión de servicio	1	100%
10	Subida de archivo adjunto del administrador financiero	1	100%
11	Subida de archivo adjunto del administrador de movilización	1	100%
12	Descarga de archivos	3	100%
13	Consulta de disponibilidad de vehículos	1	100%
14	Registro de ingreso de vehículos particulares	2	100%
15	Registro de salida de vehículos particulares	2	100%
16	Registro de kilometraje de entrada y salida de vehículos de uso institucional	3	100%
17	Registro de salida de vehículos no autorizados	2	100%

Nota. Resumen general de las pruebas de unidad del sistema
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Las pruebas de unidad se utilizan para probar los subprogramas, las subrutinas, los procedimientos individuales o las clases en un programa. Esto quiere decir que es mejor probar primero los bloques desarrollados más pequeños del programa para posteriormente tratar la interacción de estos como un conjunto. Si se requiere revisar a detalle los resultados de las pruebas de unidad, estas se pueden apreciar en el **Anexo 3** del presente trabajo.

3.7.2. Pruebas de rendimiento.

También conocidas como pruebas de caja negra, las cuales consisten en realizar pruebas para verificar que un conjunto de pequeñas partes del programa codificado funcionan juntos para cumplir una funcionalidad. Las pruebas de integración forman parte de la fase de pruebas de software, donde los módulos individuales de software son combinados y analizados como un conjunto.

En la siguiente tabla se expone de manera general el resultado de las pruebas de rendimiento del sistema desarrollado.

Tabla 13.
Pruebas de rendimiento

N°	Proceso	Número de Pruebas Realizadas	Porcentaje de pruebas exitosas
1	Registro de ingreso y salida de vehículos particulares	4	100%
2	Gestión de solicitudes de movilización	6	100%
3	Gestión de solicitudes de movilización con comisión de servicio	10	100%

Nota. Resumen general de las pruebas de rendimiento del sistema
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

De igual manera, si se es necesario observar el detalle del resultado de las pruebas de rendimiento, las mismas pueden ser apreciadas en el **Anexo 4** del presente trabajo.

3.7.3. Pruebas de carga y estrés

Este tipo de pruebas permiten realizar carga de información masiva con la finalidad de observar cómo reacciona el sistema ante ambientes extremos.

La herramienta de libre distribución que se ha escogido es JMeter ya que permite realizar pruebas de carga y estrés a sistemas de entorno web. Si se desea conocer más acerca de la herramienta el sitio oficial es **http://jmeter.apache.org/**.

Desde JMeter se debe configurar la carga que se realizrá al sistema. En la figura 55 se muestra que al sistema se lo cargará con 5000 usuarios durante un periodo de 5 segundos y de un intento por usuario.

Configuración de prueba de carga

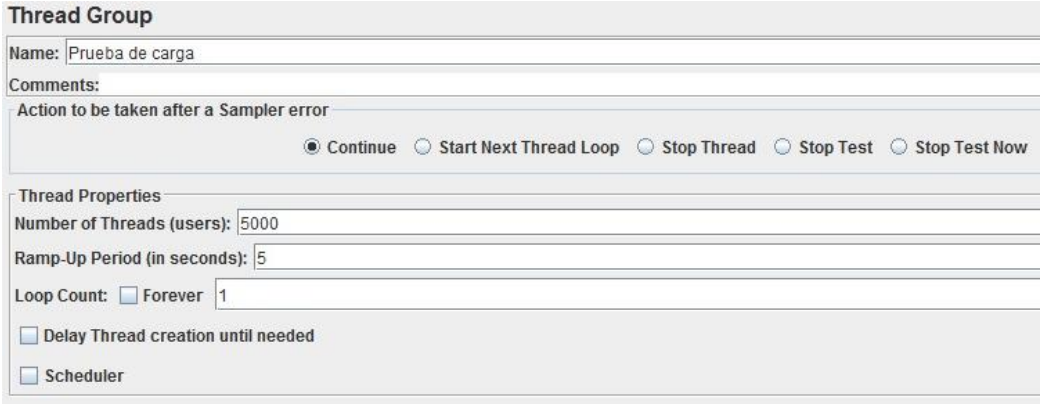


Figura 55. Configuración de carga al sistema
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Posteriormente se debe configurar una nueva petición HTTP hacia la aplicación web donde se define el sitio en específico como se muestra en la figura 56.

Petición HTTP

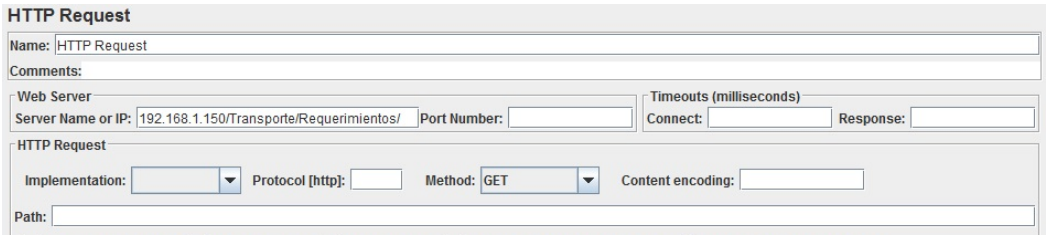


Figura 56. Petición HTTP hacia el sitio web
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

Finalmente se deben ejecutar las pruebas para poder obtener un informe grafico como se la figura 57.

Reporte grafico de pruebas de carga y estrés

Figura 57. Descripción del reporte grafico de pruebas de carga y estrés
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

El reporte generado permite observar el resultado de las pruebas realizadas las cuales fueron favorables para que el sistema sea apto y funcione correctamente en circunstancias normales. A continuación se describe el resultado las pruebas realizadas.

Tabla 14.
Resultado de pruebas de carga y estrés

Número de usuarios	5000
Rendimiento	7.697,044/minuto
Promedio	957 milisegundos
Media	721 milisegundos
Desviación	880 milisegundos

Nota. Resumen del resultado de las pruebas de carga y estrés
Elaborado por: Fabricio Pilicita & Jorge Zúñiga

CONCLUSIONES

- El desarrollo de software puede verse afectado por varias eventualidades ya que es un tema sensible de tratar entre las partes involucradas (usuario final y programador), por los distintos criterios y conocimientos que pueden existir en los mismos, los cuales influyen en retrasos en un proyecto de desarrollo como el inadecuado levantamiento de información, cambios de requerimientos en fases avanzadas, entre otros. Al utilizar la metodología UWE como herramienta para la definición de requerimientos y diseño del software, ésta ha permitido reducir demoras de manera significativa ya que se puede interactuar de mejor forma con el usuario final por medio de diagramas los cuales pueden ser interpretados sin mayor conocimiento informático, permitiendo estrechar la diferencia entre lo que requiere el usuario final y lo que entregará el equipo de desarrollo.
- Debido a la naturaleza del proyecto donde las reuniones con el usuario final se realizaron de manera informal, resultó de gran ayuda la gestión del proyecto por medio de la metodología XP, ya que la misma ofreció un manejo dinámico de los entregables desarrollados por medio de iteraciones, permitiendo que en cada reunión se encuentre un producto completo para su revisión, validación e implementación facilitando el manejo de cambios si el usuario solicitaba alguno.
- En la construcción de un sistema informático, las reglas de negocio son críticas al momento de ser plasmadas en el código fuente. Cabe recalcar que en este proyecto una de ellas fue la reserva de vehículos y conductores al momento de ser asignados a una solicitud de movilización, ya que en ese punto la información debía ser controlada de la mejor manera para no generar inconsistencias. Esto se logró al

construir primero la capa de datos y presentación, para posteriormente desarrollar la capa de negocio con sus respectivas validaciones y controles, lo cual permitió agilitar la implementación de nuevos cambios y mejoras durante el transcurso del proyecto.

- Al utilizar la herramienta JMeter para realizar pruebas de carga y estrés, permitió simular ambientes de producción complejos y evaluar la respuesta del sistema desarrollado en condiciones extremas. El resultado de estas pruebas fueron satisfactorias lo cual brinda la seguridad a los desarrolladores de que en un ambiente normal la aplicación funcionará sin inconvenientes.

RECOMENDACIONES

- Se recomienda utilizar la herramienta de desarrollo Visual Studio Comunity 2015 para la construcción de la aplicación móvil, ya que permite adaptar ciertas páginas de una aplicación web construida en .NET, en interfaces móviles basadas en Android. Esto también permite que diferentes tecnologías interoperen entre si, logrando que el código fuente sea reutilizable y flexible al momento de implementar este tipo de solución basada en distintos lenguajes de programación.
- Se recomienda utilizar el modelo MVC (Modelo Vista Controlador) ya que permite al equipo de desarrollo separar la construcción de la aplicación en diferentes capas organizadas por funcionalidad aportando escalabilidad y flexibilidad al proyecto. Esto se pudo constatar al momento de implementar nuevos requerimientos los cuales se solventaron fácilmente con la creación de pantallas y controladores sin afectar otras funcionalidades desarrolladas anteriormente

REFERENCIAS

Aguilar, M. (5 de Julio de 2015). *La importancia de una adecuada gestión de activos fijos en su negocio.* Obtenido de http://www.ey.com/Publication/vwLUAssets/Adecuada_gestion_Activos_Fijos/$FILE/Adecuada_gestion_activos_fijos.pdf

Álvarez Sánchez, A. (12 de 05 de 2015). *POO: fundamentos*. Recuperado el 21 de 06 de 2015, de http://www.desarrolloweb.com/articulos/poo-fundamentos-luis-fernandez.html

Álvarez, M. Á. (24 de 07 de 2001). *Qué es la programacion orientada a objetos*. Recuperado el 21 de 06 de 2015, de http://www.desarrolloweb.com/articulos/499.php

Álvarez, M. Á. (02 de 01 de 2014). *Qué es MVC*. Recuperado el 23 de 06 de 2015, de http://www.desarrolloweb.com/articulos/que-es-mvc.html

Bascón Pantoja, E. (2004). *El patrón de diseño modelo vista controlador y su implementación en java swing.* Recuperado el 23 de 06 de 2015, de http://www.ucbcba.edu.bo/Publicaciones/revistas/actanova/documentos/v2n4/v2.n4.bascon.pdf

Costal, D. (2014). *Introducción al diseño de base de datos.* Obtenido de http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02150.pdf

Devis Botella, R. (2010). *C++/OOP un enfoque práctico.* Autoedición.

Ganán, S. G. (2010). *Modelo XP para desarrollo de proyecto.* Milagro, Ecuador: Universidad Estatal de Milagro.

Izquierdo, L. (2014). *Introducción a la Programación Orientada a Objetos.* Recuperado el 21 de 06 de 2015

Jefatura de Gabinete de Ministros. (2010). *Manual de nuenas prácticas de gestión de activos fijos.* Obtenido de http://www.jefatura.gob.ar/archivos/pme/publicaciones/manual_gaf.pdf

Kimmel, P. (2008). *Manual de UML* (Vol. 1). (C. Tapia Montes de Oca, Ed., & J. Pérez Castellanos, Trad.) México: Mcgraw-hill interamericana.

Larman, G. (2001). *Apply UML and patterns an intruduction to object-oriented analisys and designed and the unified process.* United States of America: Patti Guerrieri. Recuperado el 22 de 06 de 2015

López, F. (27 de 07 de 2015). *Conasa*. Obtenido de Metodologías ágiles para la gestión de proyectos: http://www.conasa.es/metodologias-agiles-para-gestion-de-proyectos/#

López, J. (08 de 12 de 2010). *Ingeniería de software*. Recuperado el 05 de 07 de 2015, de http://upolijenny.blogspot.com/2010/12/ingenieria-web.html

Object Management Group. (2015). *Unified modeling language*. Recuperado el 05 de 07 de 2015, de http://www.uml.org/

Páez, P. (2012). *Resolución No. SCPM-DS-2012-001.* Superintendencia de Control del Poder de Mercado, Quito. Recuperado el 21 de 06 de 2015

Prieto Ramos, Á., Sosa Sánchez, E., & Rodríguez Echeverría, R. (2004). *Programación orientada a objetos.* Recuperado el 22 de 06 de 2015, de http://www.euroamericano.edu.ec/contenido/0321%20programacion%201/Programacion-Orientada-a-Objetos-Roberto-Rodriguez-Echeverria-Encarna-Sosa-Sanchez-y-Alvaro-Prieto-Ramos.pdf

Quiroga, A. (23 de 03 de 2015). *UML based web engineering*. Recuperado el 05 de 07 de 2015, de http://proyectogradoingenieriasistemas.blogspot.com/

Reinberg, A. C. (21 de Abril de 1986). *Normas para la administración de activos fijos de la presidencia.* Obtenido de Acuerdo ministerial 130: http://www.presidencia.gob.ec/wp-

content/uploads/downloads/2012/09/Instructivo-para-la-Administracion-de-Activos-Fijos.pdf

Reyes, V., & Urgiles, P. (2007). *Modelo administrativo para uso y control de la flota vehicular del cuerpo de ingenieros del ejército del grupo Mazar.* Cuenca, Azuay, Ecuador. Recuperado el 16 de 05 de 2015

Rivera Trejo, J. P. (2014). *Sistema de control de acceso.* Santiago de Querétaro, México. Recuperado el 16 de 06 de 2015

Roig Vásquez, P. E. (1997). *UML gota a gota.* México: Addison Wesley Longman. Recuperado el 22 de 06 de 2015

Sánchez, J. (2004). *Principios sobre bases de datos relacionales.* Recuperado el 05 de 05 de 2015, de http://www.jorgesanchez.net/bd/bdrelacional.pdf

Sommerville, I. (2006). *Ingenieria del software* (7 ed.). (J. A. Clares, Ed.) Madrid, España: Pearson educación S.A. Recuperado el 05 de 07 de 2015, de https://books.google.com.ec/books?id=gQWd49zSut4C&pg=PA5&dq=ingenieria+de+software&hl=es&sa=X&ei=m82ZVcLyDMSjNoffqNgL&ved=0CBsQ6AEwAA#v=onepage&q=ingenieria%20de%20software&f=false

Terán, P. (2012). *Análisis, diseño e implementación de un sistema de control vehicular en instituciones públicas.* Ibarra, Imbabura, Ecuador: Universidad Técnica del Norte. Recuperado el 16 de 06 de 2015, de http://repositorio.utn.edu.ec/bitstream/123456789/1080/1/04%20ISC%20223-trabajo_final.pdf

The Standish Group. (2015). *About the standish group*. Recuperado el 05 de 07 de 2015, de http://www.standishgroup.com/about

Universidad del Azuay. (05 de Julio de 2015). *El modelo Relacional.* Obtenido de http://www.uazuay.edu.ec/analisis/El%20modelo%20relacional.pdf

ANEXOS

Anexo 1. Especificación de requisitos

La especificación de requisitos del sistema se encuentra localizado dentro del directorio de la aplicación en la ruta ./Recursos/Requisitos/Especificaciones.pdf

Anexo 2. Diccionario de datos

El diccionario de datos del sistema se encuentra localizado dentro del directorio de la aplicación en la ruta ./Recursos/Base_de_datos/Diccionario_de_datos.pdf

Anexo 3. Pruebas de unidad del Sistema de Gestión de Transporte

Las pruebas de unidad del sistema se encuentran localizadas dentro del directorio de la aplicación en la ruta ./Recursos/Pruebas/Pruebas_de_unidad.pdf

Anexo 4. Pruebas de rendimiento del Sistema de Gestión de Transporte

Las pruebas de rendimiento del sistema se encuentran localizadas dentro del directorio de la aplicación en la ruta ./Recursos/Pruebas/Pruebas_de_rendimiento.pdf

Anexo 5. Manuales de usuario

Los manuales de usuario de acuerdo a cada perfil se encuentran localizados dentro del directorio de la aplicación en la ruta ./Recursos/Manuales