

Gestion de Parquaderos

Luis Alejandro Ocampo - 20172020050
Daniel Esteban Rodriguez - 20172020120

3 de junio de 2020

Índice general

I	PROYECTO	7
1.	Proyecto	9
1.1.	Introduccion	9
1.2.	Descripción General	10
1.3.	Objetivos	11
1.4.	Alcances y Limites	12
1.5.	Cronograma	13
2.	Metodología	15
2.1.	Análisis de Requerimientos	15
2.2.	Procesos de Software	16
II	DISEÑO	17
3.	UML[4, 1, 2, 3]	19
3.1.	Introduccion	19
3.2.	Historia	20
3.3.	Estructura	21
3.3.1.	Elementos de UML	21
3.3.2.	Relaciones de UML	23
3.3.3.	Diagramas de UML	25
4.	Casos de Uso	27
4.1.	Introduccion	27
5.	Inetracciones	35
5.1.	Introduccion	35
5.2.	Diagramas de Secuencia	36
5.3.	Diagramas de Comunicación	37

6. Clases	39
6.1. Introduccion	39
7. Estados	41
7.1. Introduccion	41
8. Actividades	43
8.1. Introduccion	43
8.2. WorkFlow	44
9. Componentes	45
9.1. Introduccion	45
10.Sistemas	47
10.1. Introduccion	47
11.Nodos	49
11.1. Introduccion	49
III CIERRE	51
12.Conclusiones	53

Índice de figuras

1.1. Cronograma	13
2.1. Cascada	16
2.2. SCRUM	16

Parte I

PROYECTO

Capítulo 1

Proyecto

1.1. Introduccion

En este proyecto se pretende desarrollar una aplicacion web de tipo economia colaborativa, en la que se gestionen los parqueaderos en la ciudad y los usuarios puedan solicitar un servicio de alquiler de los mismos, de igual manera, los mismos usuarios pueden ofrecer en alquiler un parqueadero en su propiedad, todo basandose en, como se menciono antes, un consumo colaborativo, en el cual los consumidores pueden actuar como proveedores de recursos o receptores de recursos.

1.2. Descripción General

La plataforma esta pensada para generar un comercio virtual entre arrendatario/arrendador de manera simple y practica, para poder alquilar u ofrecer en alquiler un espacio donde aparcar cualquier tipo de vehiculo, ya sean carros, camionetas, motos, e incluso bicicletas y vehiculos especiales, todo funcionando como una economia colaborativa donde cualquier persona pueda obtener un beneficio al usar la plataforma, se va a utlizar un sistema de valoraciones tanto para quien ofrece un espacio como para quien decide tomar éste en alquiler, todo a manera de guia para los usuarios.

1.3. Objetivos

El proyecto tiene como objetivo principal facilitar el transporte en la ciudad para usuarios de carros, motos, bicicletas, etc. sin tener que preocuparse por donde van a estacionar su vehiculo o por el precio que van a pagar, esto con el afan de agilizar y facilitar el proceso de apartado de un espacio para aparcar un vehiculo, ademas de combatir los arbitrarios precios de parqueaderos en sectores de mucha concentracion laboral o estudiantil.

1.4. Alcances y Limites

Alcances

El presente proyecto explorara el sector vehicular en la ciudad, apuntando sobretodo, al sector de la poblacion que se presenta constantemente a espacios donde se pagan unos muy altos precios de estacionamiento, o donde es muy limitado el espacio para aparcar, ya sea una zona de alta densidad laboral, cerca a un centro comercial, o una zona universitaria.

Limites

Las limitaciones del proyecto van mas orientadas a los sectores donde el sistema de parqueo no presente fallos criticos, es decir, sectores con baja concurrencia vehicular no se contemplan dentro del proyecto por su baja probabilidad tanto de potenciales arrendatarios como de potenciales arrendadores, aunque esto va directamente ligado al desarrollo del proyecto.

1.5. Cronograma

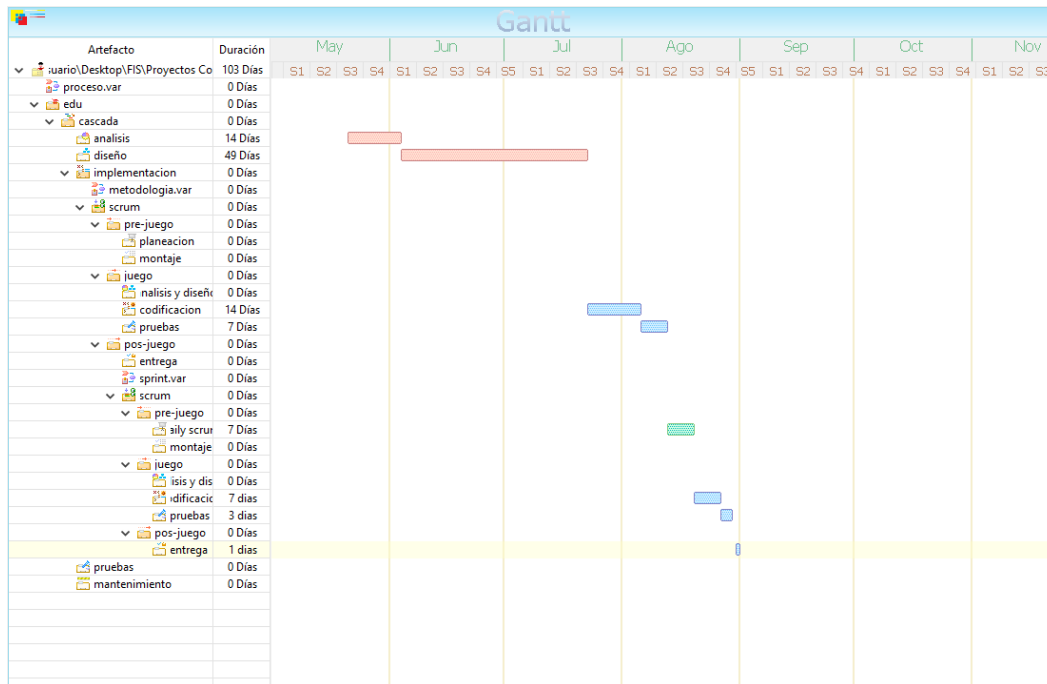


Figura 1.1: TimeLine del Cronograma

Descripcion

El Cronograma consta de una duracion de aproximadamente 14 semanas donde se hace un especial énfasis en la etapa de diseño, correspondiente al proceso que vamos a incorporar (Modelo de Cascada) se debe enfocar una gran parte de los esfuerzos del equipo en esta etapa de diseño, esto, después de haber realizado el correspondiente análisis de requerimientos, con todo lo que conlleva esta etapa.

Una vez finalizada de etapa de diseño, se incorpora una metodología agilista para enfrentar la etapa de implementación, la Metodología SCRUM, donde únicamente tomamos de esta metodología, como ya se menciona, la fase de implementación y prueba, que consta de dos iteraciones del proceso tanto de codificación como de sus respectivas pruebas, este proceso en total suele llevar entre 2 a 4 semanas, donde

en este periodo, además de realizar los procesos ya mencionados, internamente contiene una fase de daily Scrum, cuya finalidad es realizar reuniones diarias para retroalimentar el trabajo hasta ese punto, ver los posibles fallos y compartir las ideas del equipo, tal y como lo describe el modelo.

Y para finalizar, con los tiempos justos del semestre y esperando que no haya ningún tipo de contratiempo, se espera realizar la correspondiente entrega del proyecto.

Capítulo 2

Metodología

2.1. Analisis de Requerimientos

El proyecto de gestión de parqueaderos basado en el principio de economía colaborativa deberá contar con una serie de especificaciones y requisitos para su correcto desarrollo e implementación, para comenzar será una aplicación desarrollada en entorno web para su fácil acceso desde cualquier dispositivo y esta deberá contar con la posibilidad de creación y manejo de cuentas personales para la administración de sus objetos vinculados ya sean automóviles o parqueaderos como tal permitiendo añadir estos, de igual manera permitirá modificar la tarifa que desea manejar el usuario y medios de pago, cada automóvil registrado deberá contar con modelo y número de placa y cada parqueadero con la ubicación, número de plazas disponibles, y si es abierto o cerrado.

Se deberá poder consultar en un mapa la ubicación de los parqueaderos registrados disponibles y poder consultar la tarifa que manejan estos y los medios de pago disponibles, por ende este mapa debe estar actualizado a tiempo real.

2.2. Procesos de Software

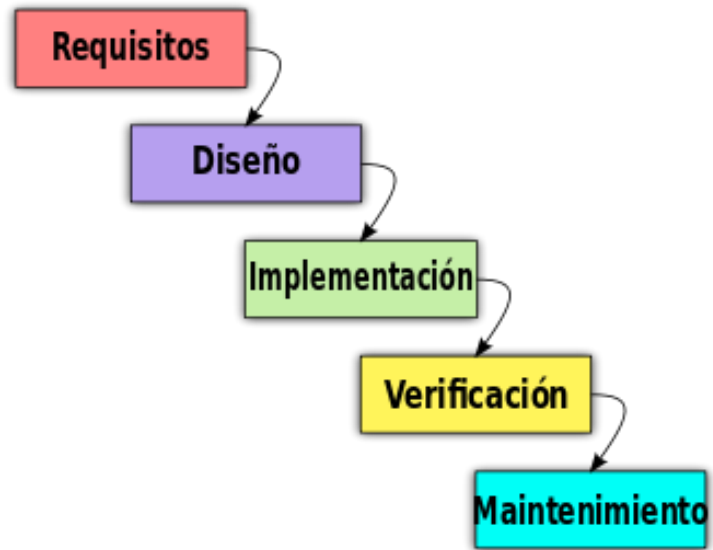


Figura 2.1: Proceso Cascada



Figura 2.2: Metodologia SCRUM

Parte II

DISEÑO

Capítulo 3

UML[4, 1, 2, 3]

3.1. Introduccion

UML es una notación estándar para el modelado de software para la arquitectura, diseño e implementación de software, ya sea en su estructura como en comportamiento siendo este bastante visual, al ser un lenguaje que usa diagramas este no se limita a usarse únicamente en el desarrollo de software. UML no es un proceso de desarrollo, es decir, no describe los pasos sistemáticos a seguir para desarrollar software. UML sólo permite documentar y especificar los elementos creados mediante un lenguaje común describiendo modelos. Es un lenguaje de propósito general para el modelado orientado a objetos, que combina notaciones provenientes desde: Modelado Orientado a Objetos, Modelado de Datos, Modelado de Componentes, Modelado de Flujos de Trabajo (Workflows).








3.2. Historia

El lenguaje UML comenzó a gestarse en octubre de 1994, cuando Rumbaugh se unió a la compañía Rational fundada por Booch (dos reputados investigadores en el área de metodología del software). El objetivo de ambos era unificar dos métodos que habían desarrollado: el método Booch y el OMT (Object Modelling Tool). El primer borrador apareció en octubre de 1995. En esa misma época otro reputado investigador, Jacobson, se unió a Rational y se incluyeron ideas suyas. Estas tres personas son conocidas como los “tres amigos”. Además, este lenguaje se abrió a la colaboración de otras empresas para que aportaran sus ideas. Todas estas colaboraciones condujeron a la definición de la primera versión de UML. Figura 1: Logo de UML Se necesitaba por tanto un lenguaje no sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema. Con este objetivo se creó el Lenguaje Unificado de Modelado (UML: Unified Modeling Language). UML se ha convertido en ese estándar tan ansiado para representar y modelar la información con la que se trabaja en las fases de análisis y, especialmente, de diseño. Esta primera versión se ofreció a un grupo de trabajo para convertirlo en 1997 en un estándar del OMG (Object Management Group <http://www.omg.org>). Este grupo, que gestiona estándares relacionados con la tecnología orientada a objetos (metodologías, bases de datos objetuales, CORBA, etc.), propuso una serie de modificaciones y una nueva versión de UML (la 1.1), que fue adoptada por el OMG como estándar en noviembre de 1997. Desde aquella versión han habido varias revisiones que gestiona la OMG Revision Task Force. La última versión aprobada es la 1.4. En estos momentos se está desarrollando una nueva versión en la que se incluirán cambios importantes (principalmente añadir nuevos diagramas) que conducirán a la versión 2.0 planificada para fines del 2002.




3.3. Estructura

3.3.1. Elementos de UML



Elemento Estructurales

Elemento	Definicion	Notacion
Caso de uso	Un caso de uso representa la forma en como un Cliente (Actor) opera con el sistema en desarrollo, además de la forma, tipo y orden en como los elementos interactúan (operaciones o casos de uso).	
Colaboracion	La colaboración permite que una clase se comunice con otra clase con el propósito de utilizar los métodos de esta ultima para mejorar algún servicio o mantener una relación lógica de dependencia.	
Clase	Es un elemento estructural en el que se modelan atributos y operaciones de un conjunto de objeto.	
Interface	Esta especifica qué se debe hacer pero no su implementación. Serán las clases que implementen estas interfaces las que describen la lógica del comportamiento de los métodos.	
Componente	Un componente, también denominado componente simple o atómico, es un objeto que tiene una representación gráfica, que se puede mostrar en la pantalla y con la que puede interactuar el usuario.	
Nodo	un nodo es cada uno de los elementos de una lista enlazada , un árbol o un grafo en una estructura de datos. Cada nodo tiene sus propias características y cuenta con varios campos; al menos uno de éstos debe funcionar como punto de referencia para otro nodo.	
Artefacto	En términos generales de software, un artefacto es algo producido por el proceso de desarrollo de software, ya sea una documentación relacionada con el software o un archivo ejecutable.	


Elemento de Comportamiento

Elemento	Definicion	Notacion
Estado	El diagrama de estado se usa para dar forma al comportamiento de un objeto, de una clase. Se representa la secuencia de estados que un objeto de la clase tiene durante su vida, según las acciones que van sucediendo.	
Actividad	Básicamente se puede decir que el diagrama de actividades modela el flujo de actividades. Estos pueden ser procesos dentro de un sistema informático, procesos de casos de uso o procesos comerciales. Estas actividades pueden descomponerse en muchas actividades secundarias más pequeñas: las acciones, que pueden llevarse a cabo cronológicamente.	
Transicion	Es un comportamiento que comprende un conjunto de mensajes intercambiados entre un conjunto de objetos, dentro de un contexto particular para alcanzar un propósito específico.	

Elemento Agrupación

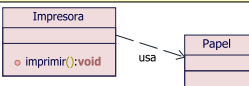
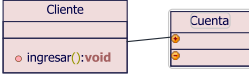
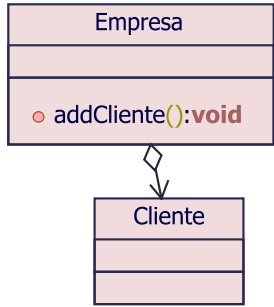
Elemento	Definicion	Notacion
Sistema	Es un mecanismo de propósito general para organizar elementos en grupos. En estos se pueden agrupar los elementos estructurales, de comportamiento e incluso otros elementos de agrupación, son los elementos de agrupación básicos con los cuales se puede organizar un modelo UML.	
Frame	Similar al Sistema, usado mas en diagramas dinamicos	

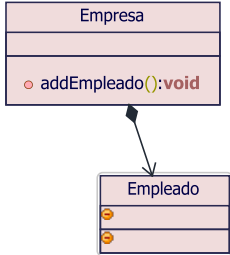
Elemento Anotación

Elemento	Definicion	Notacion
Nota	<p>Son comentarios que se pueden aplicar para describir, clasificar y hacer observaciones sobre cualquier elemento de un modelo.</p> <p>El tipo principal de anotación es la nota que simplemente es un símbolo para mostrar restricciones y comentarios junto a un elemento o un conjunto de elementos.</p>	

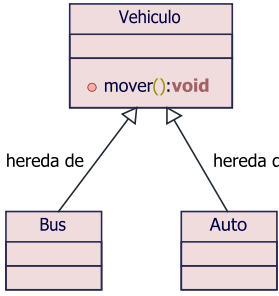
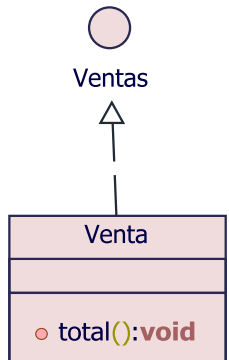
3.3.2. Relaciones de UML

Relaciones Estructurales

Elemento	Definicion	Notacion
Dependencia	Es una relación de significado entre dos elementos, donde cualquier cambio a un elemento independiente, puede afectar el significado de otro elemento dependiente.	
Asociacion	Las asociaciones representan las relaciones más generales entre clases, es decir, las relaciones con menor contenido semántico. Para UML una asociación va a describir un conjunto de vínculos entre las instancias de las clases.	
Agregacion	Es muy similar a la relación de Asociación solo varía en la multiplicidad ya que en lugar de ser una relación uno a uno. ^{es} de uno a muchos".	

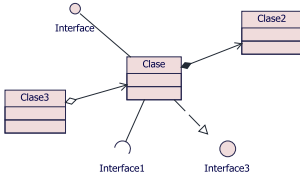
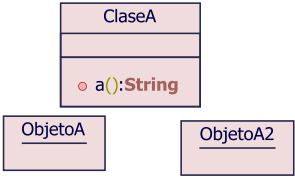
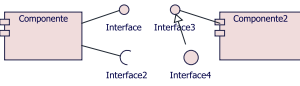
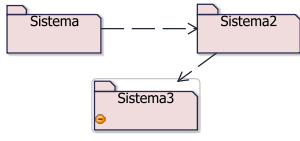
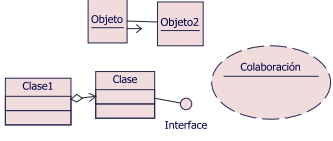
Composicion	Aporta documentación conceptual ya que es una relación de vida”, es decir, el tiempo de vida de un objeto está condicionado por el tiempo de vida del objeto que lo incluye.	 <pre> classDiagram class Empresa { +addEmpleado():void } class Empleado { +instance variables } Empresa "1" *-- "*" Empleado </pre>
-------------	--	---

Relaciones de Comportamiento

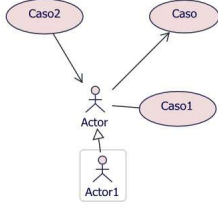
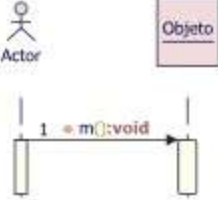
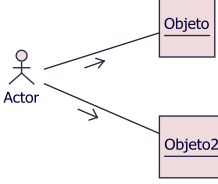
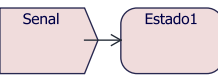
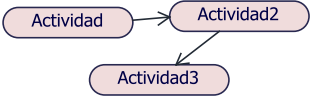

Elemento	Definicion	Notacion
Generalizacion	Esta hace referencia a la relación de una súper clase o clase padre con una subclase o clase hija. La generalización significa que los objetos hijos se pueden emplear en cualquier clase donde pueda aparecer el padre, pero no a la inversa.	 <pre> classDiagram class Vehiculo { +mover():void } class Bus class Auto Vehiculo < -- Bus Vehiculo < -- Auto </pre>
Realizacion	Es una relación de contrato con otra clase. Se la utiliza para implementar una interfaz.	 <pre> classDiagram class Ventas class Venta { +total():void } Venta -- > Ventas </pre>

3.3.3. Diagramas de UML

Diagramas Estáticos (o Estructurales)

Elemento	Definicion	Notacion
Clases	Describe la estructura de un sistema mostrando las clases del sistema, sus atributos, operaciones (o metodos), y las relaciones entre los objetos.	
Objetos	Los diagramas de objetos modelan las instancias de elementos contenidos en los diagramas de clases. Un diagrama de objetos muestra un conjunto de objetos y sus relaciones en un momento concreto.	
Componentes	Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes.	
Sistemas	Muestra como un proyecto esta dividido en agrupaciones logicas mostrando las dependencias entre esas agrupaciones.	
Estructura Compuesta	Muestra la estructura interna de una clase y las colaboraciones que esta estructura hace posibles.	

Diagramas dinámicos (o de Comportamiento)

Elemento	Definicion	Notacion
Casos de uso	Los diagramas de caso de uso modelan la funcionalidad del sistema usando actores y casos de uso.	
Secuencias	Un diagrama de secuencia es un tipo de diagrama de interacción porque describe cómo (y en qué orden) un grupo de objetos funcionan en conjunto.	
Comunicacion	Un diagrama de comunicación modela las interacciones entre objetos o partes en términos de mensajes en secuencia.	
Estados	Es un diagrama que muestra transiciones entre diversos objetos.	
Actividades	Representa los flujos de trabajo paso a paso de negocio y operacionales de componentes en un sistema.	
Vista Conjunta de Interaccion	Es una representacion grafica de una interaccion, este se distingue fuertemente de los diagramas desecuencia y de comunicacion, dos de los otrosdiagramas de interaccion.	

Capítulo 4

Casos de Uso

4.1. Introduccion

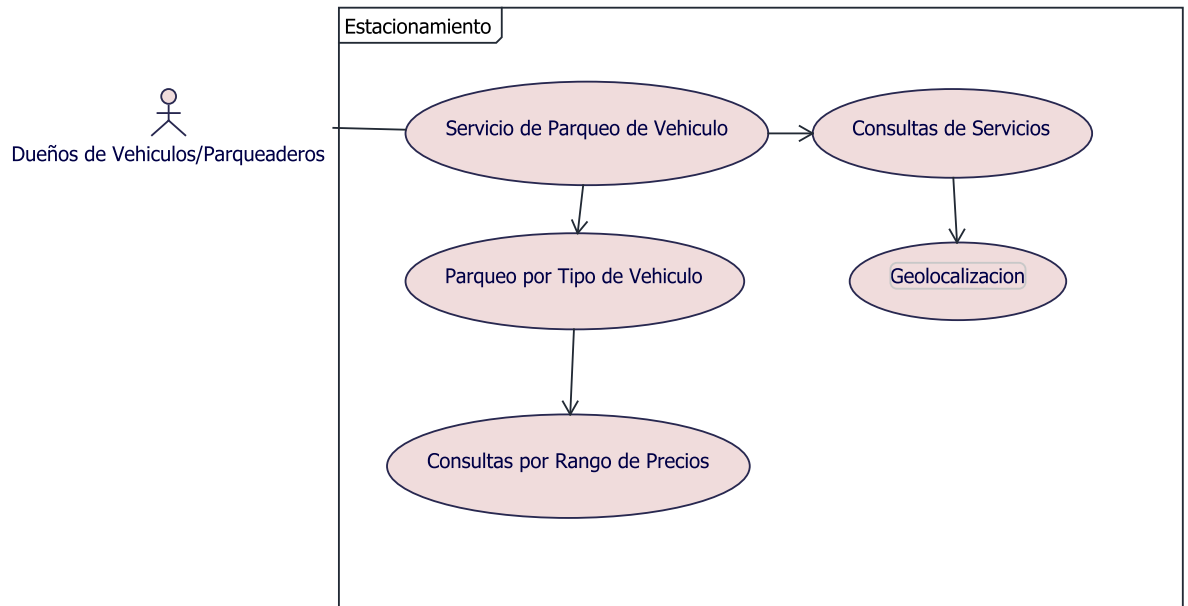
En base a lo que hemos revisado anteriormente nos damos cuenta que existen diferentes metodologías que nos permiten desarrollar software de calidad enfocadas a las necesidades que se tengan.

Dentro de un proyecto de software existen diferentes etapas, una de estas independientemente de la metodología que se esté utilizando es definir de manera concisa y general qué es lo que se quiere, ya que es fundamental para definir los requerimientos de software porque muchas veces lo que se plantea no es lo que realmente se ve en el proyecto terminado, es por esto que se definen formas de presentar una perspectiva primitiva de lo que será el software una vez finalizado.

Existen diferentes formas de representar la funcionalidad del software sin estar terminado, una de estas es el Lenguaje Unificado de Modelado “UML”, que es el sistema de modelado de software más conocido y utilizado en la actualidad.

Dentro de UML se pueden encontrar diversos diagramas que permiten representar las diferentes perspectivas de un sistema, además, apoyados en la herramienta Coloso se consiguió el correcto modelado de dichos diagramas.

Los Casos de Uso son diagramas que permiten representar que hará el sistema pero no como funciona, a continuación se analiza su implementación en nuestro proyecto de software.



A continuación, se realizaron una serie de tablas con descripciones de los casos de uso del diagrama anterior.

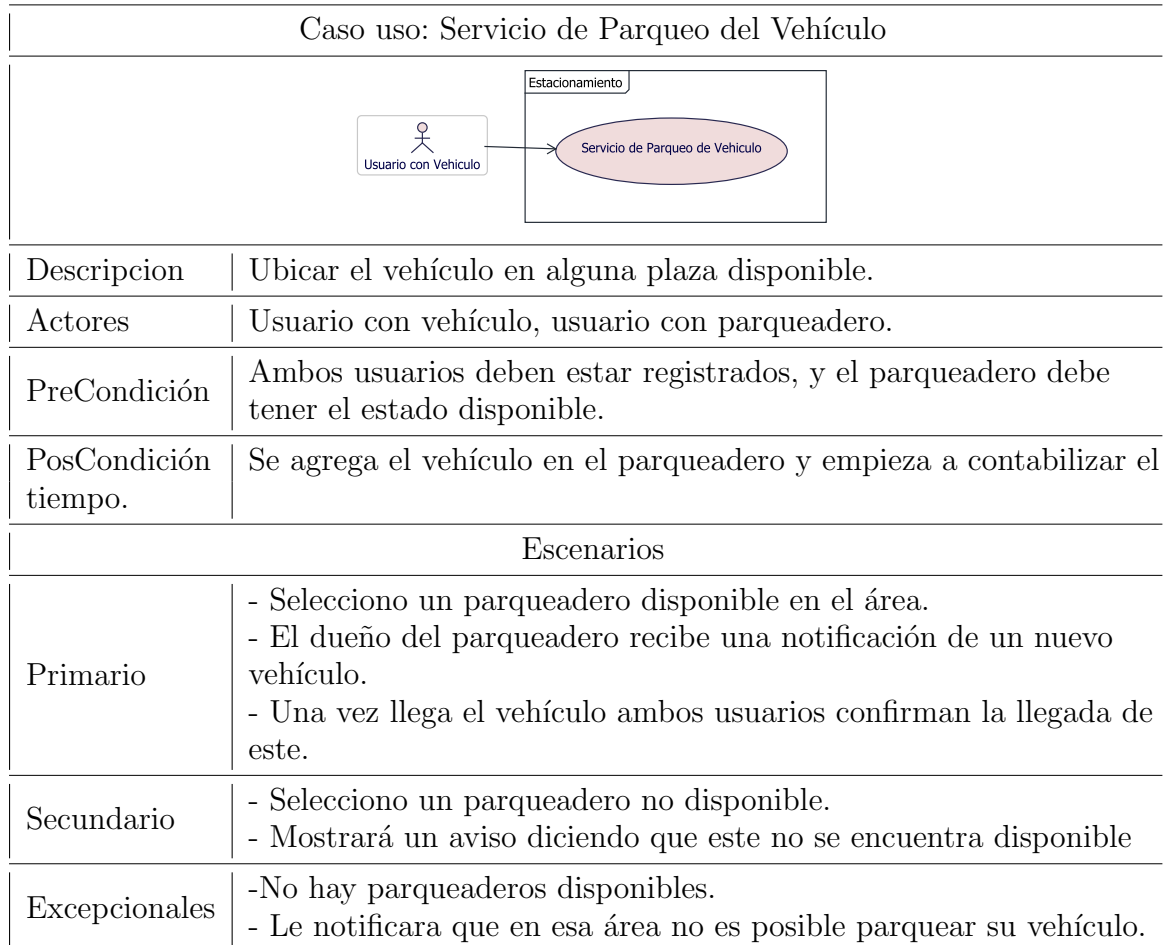


Figura 0.0. Descripción de caso de uso "Servicio de Parqueo de Vehículo"

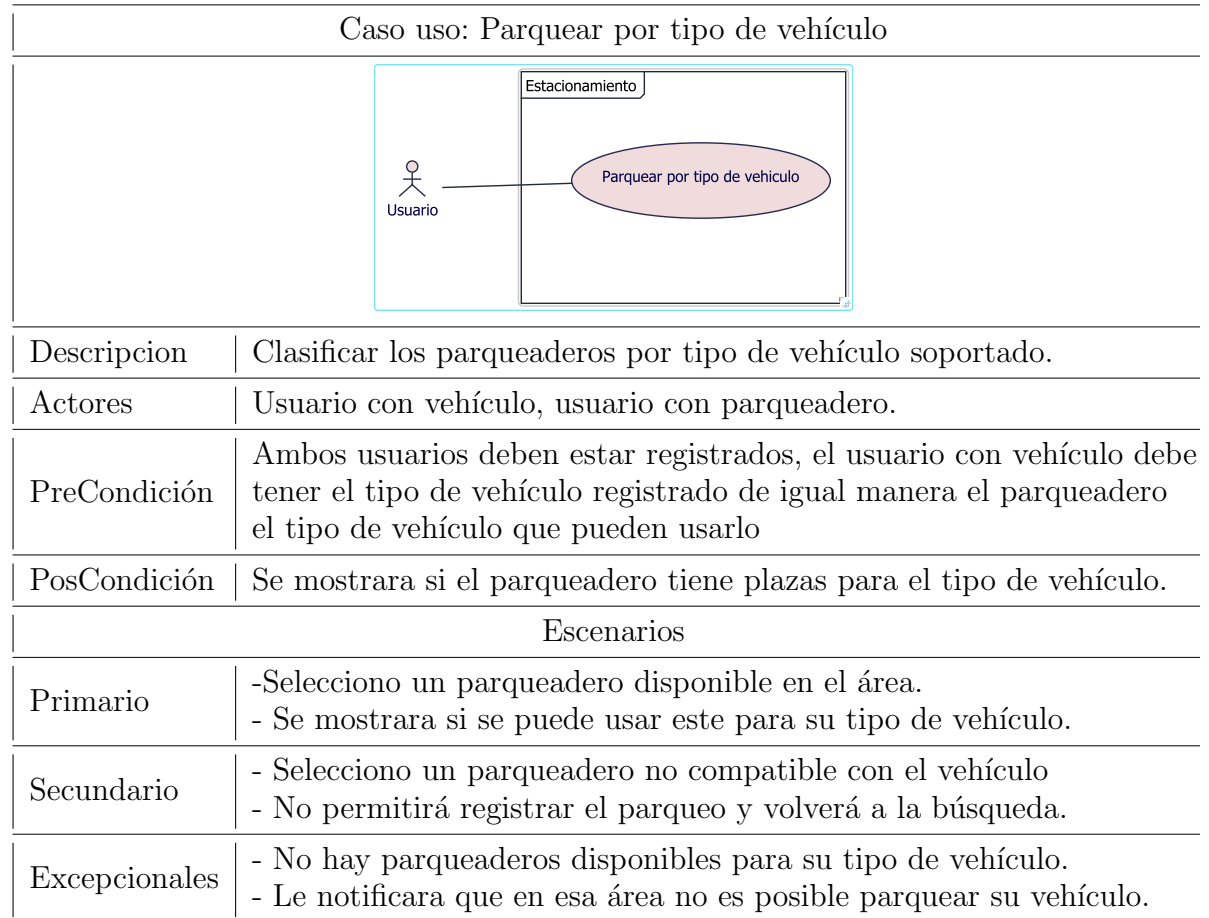


Figura 0.0. Descripción de caso de uso "Parqueo por Tipo de Vehículo"

Caso uso: Consulta por Rango de Precios	
<p>The diagram shows a use case titled 'Consulta por Rango de Precios' (Consult by Price Range) within a system boundary labeled 'Estacionamiento' (Parking). The actor is 'Usuario' (User), represented by a stick figure. A line connects the actor to the use case oval.</p>	
Descripcion	Mostrar solo los parqueaderos que se encuentren en el rango de precios seleccionado por el usuario
Actores	Usuario con vehículo
PreCondición	El usuario debe estar en el menú de búsqueda
PosCondición	Solo se verán los parqueaderos filtrados por precio.
Escenarios	
Primario	<ul style="list-style-type: none">-Selecciono un rango de precios en el menú- El mapa se actualizara mostrando solo los parqueaderos que entran en ese rango
Secundario	<ul style="list-style-type: none">- Selecciono in rango de precios en el menú-No aparece ningún parqueadero en ese rango.- Se mostrara un mensaje diciendo que en esa área no hay parqueaderos dentro de ese rango de precios
Excepcionales	<ul style="list-style-type: none">- - El rango de precios seleccionado es de 0.- No se mostrara ningún parqueadero en el área.

Figura 0.0. Descripcion de caso de uso Consulta por rango de precios

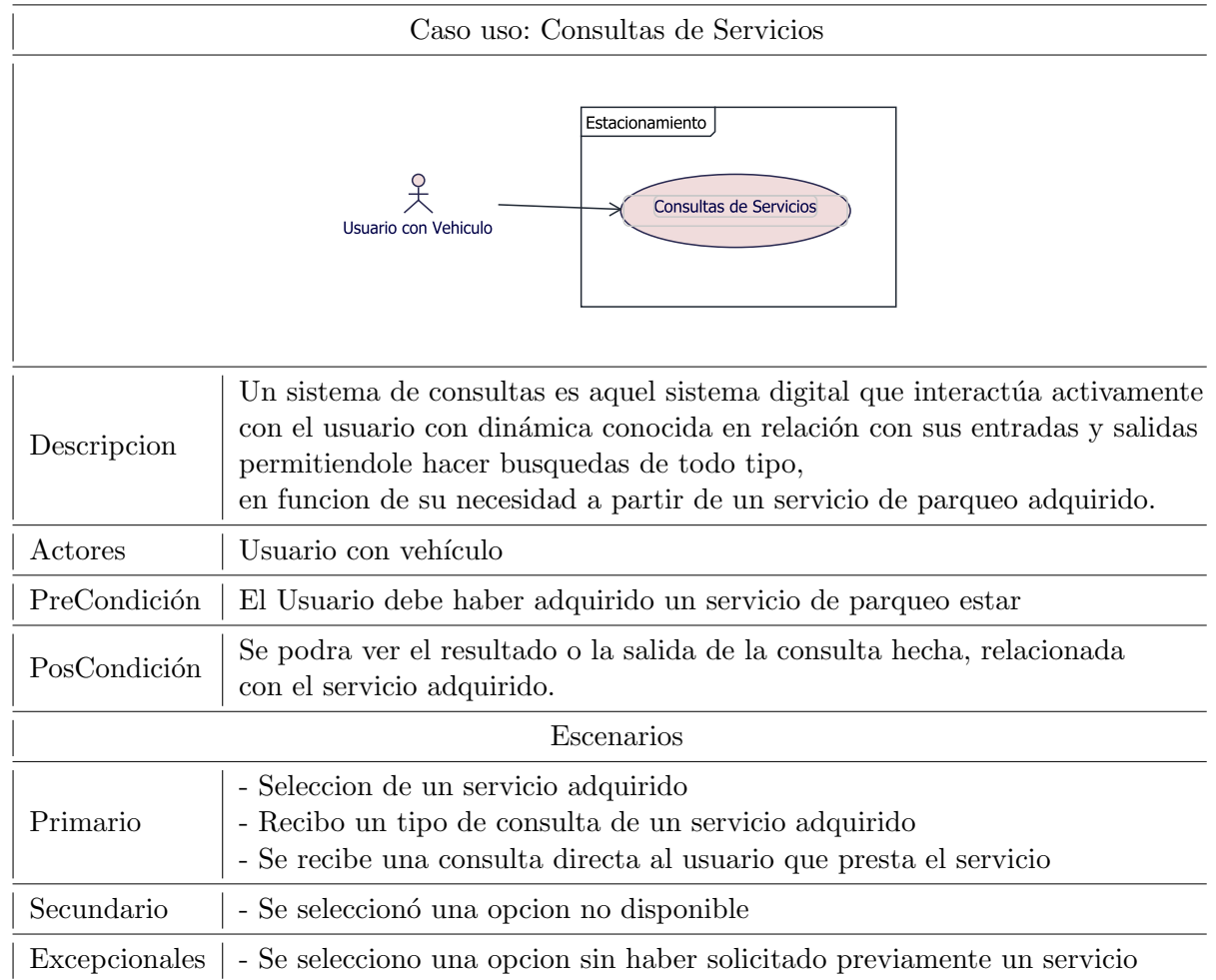
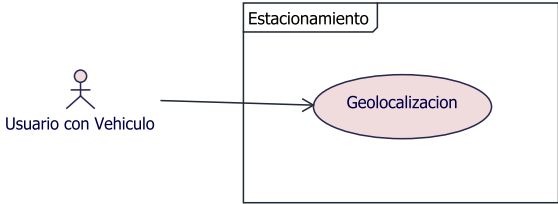


Figura 0.0. Descripción de caso de uso Consulta de Servicios

Caso uso: Geolocalizacion real	
 <pre> graph LR Actor[Usuario con Vehiculo] --> UseCase((Geolocalizacion)) subgraph Estacionamiento UseCase end </pre>	
Descripcion	La Geolocalización es un sistema que determina la ubicación de un objeto en un entorno físico o virtual (Internet). A menudo, el objeto es una persona que quiere utilizar un servicio basado en la ubicación, mientras mantiene su privacidad.
Actores	Usuario con vehículo
PreCondición	<ul style="list-style-type: none"> - El Usuario debe haber registrado su vehiculo - Haber seleccionado un servicio previamente - Haber dado los requeridos permisos de ubicacion
PosCondición	<ul style="list-style-type: none"> - Se podra seleccionar o visualizar directamente desde el mapa la ubicacion deseada con las opciones previamente seleccionadas.
Escenarios	
Primario	<ul style="list-style-type: none"> - Se genera un mapa de la zona con los requerimientos seleccionados - Se gestiona una solicitud de servicio
Secundario	<ul style="list-style-type: none"> - Se seleccionó una zona no disponible para visualizacion geo- espacial - No hay disponibilidad para el servicio solicitado
Excepcionales	<ul style="list-style-type: none"> - Se selecciono la geolocalizacion sin haber solicitado previamente un servicio - No fueron autorizados los permisos (ubicacion) requeridos para el funcionamiento de la herramienta

Capítulo 5

Inetracciones

5.1. Introduccion

contenido.....

5.2. Diagramas de Secuencia

5.3. Diagramas de Comunicación

Capítulo 6

Clases

6.1. Introduccion

contenido.....

Capítulo 7

Estados

7.1. Introduccion

contenido.....

Capítulo 8

Actividades

8.1. Introduccion

contenido.....

8.2. WorkFlow

Capítulo 9

Componentes

9.1. Introduccion

contenido.....

Capítulo 10

Sistemas

10.1. Introduccion

contenido.....

Capítulo 11

Nodos

11.1. Introduccion

contenido.....

Parte III

CIERRE

Capítulo 12

Conclusiones

Bibliografía

- [1] R. Iakovlev, I. Vatamaniuk, and D. Malov. Architecture transformation of the corporate information providing system for a scientific organization. In *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, pages 873–878, 2019.
- [2] S. Iram, D. Al-Jumeily, and J. Lunn. An integrated web-based e-assessment tool. In *2011 Developments in E-systems Engineering*, pages 271–275, 2011.
- [3] T. Rais Castro and S. Nice Alves de Souza. Graphic logical desing for ordb. *IEEE Latin America Transactions*, 11(4):1097–1103, 2013.
- [4] Booch G Rumbaugh J, Jacobso I. *The Unified Modeling Language Reference Manual, Second Edition*.