# Resultado de imagen para computacion paralela

PROYECTO: MODELO DE VISTA 4 + 1 PARA SISTEMA DE PARQUEADERO



Karen Torres.



Erika Vásquez.

*Ing. Jorge López*

***30/01/2020***

***Arquitectura de Aplicaciones***

# Introducción

En el presente documento se especifica la arquitectura de manera organizada para el desarrollo de un Sistema de parqueadero, con el uso de multiples vistas.

* La vista lógica describe el modelo de objetos del diseño cuando se usa un método de diseño orientado a objetos. Para diseñar una aplicación muy orientada a los datos, se puede usar un enfoque alternativo para desarrollar algún otro tipo de vista lógica, tal como diagramas de entidad-relación.
* La vista de procesos describe los aspectos de concurrencia y sincronización del diseño.
* La vista física describe el mapeo del software en el hardware y refleja los aspectos de distribución.
* La vista de desarrollo describe la organización estática del software en su ambiente de desarrollo

Para cada vista se define un conjunto de elementos (componentes, contenedores y conectores), la forma y los patrones con que trabajan.

Al proporcionar esta arquitectura se puede dar una futura solución para implementar, consiguiendo asi resolver el problema de los parqueaderos.

# Problemática

Debido que en la Ciudad de Loja se encuentra en continuo crecimiento, se evidencia la falta de espacios disponibles de parqueo, llevando consigo el trafico vehicular especialmente en el centro de la ciudad. Este problema ha empezado a ser cubierto por los parqueaderos privados, pero aun no se ha implentado una solucion que ayude tanto los conductos a encontrar un parqueadero sin perder tanto tiempo de buscar si tienen espacios disponibles o no, como a los parqueaderos a proporcionar información de sus espacios disponibles y los servicios que ofrecen.

# Solución

Para dar la solución al siguiente problema se vio la necesidad de implementar el Modelo de Vistas 4 + 1, ya que este modelo se tata de especificar abstracciones, descomposición y composición del sistema. Para dar a conocer al usuario los requerimientos, funcionalidades que tiene el sistema.

Como se puede ver en la **Figura 1** se demuestran las cuatro vistas del modelo: vista lógica, vista de desarrollo, vista de proceso y vista fisica. Ademas de los casos de uso o escenarios, mismo que ilustran la arquitectura usada como una vista mas.

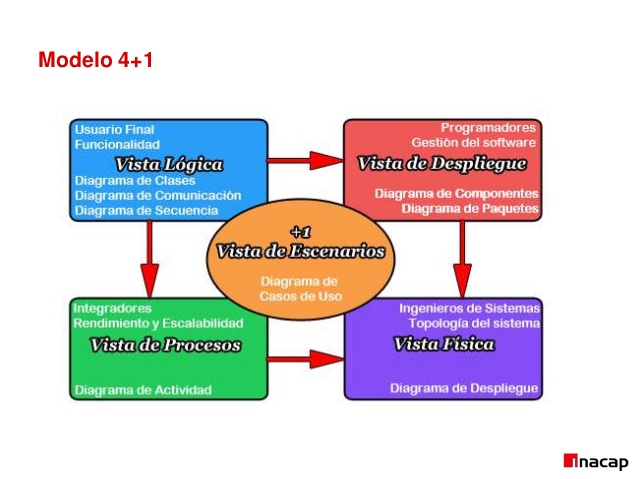


Figura . Modelo de Vista 4+1

Requisitos específicos

## Requisitos funcionales

#### **FR-011 El administrador deberá crear una cuenta de parqueo.**

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-011* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador, para poder gestionar espacios de parqueo, gestionar el alquiler de espacio de parqueo y gestionar servicios varios.* |
| ***Entradas*** | *Datos personales del parqueadero (nombre, dirección, latitud, longitud, usuario, contraseña).* |
| ***Salidas*** | *Cuenta de parqueo creada correctamente y almacenada en la base de datos.* |

#### **FR-012 El administrador podrá loguearse para poder acceder al sistema.**

#### 

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-012* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador para poder tener acceso al sistema.* |
| ***Entradas*** | *-Usuario.*  *-Contraseña.* |
| ***Salidas*** | *Usuario autenticado.* |

#### 

#### **FR-015 Cuando el administrador esté logueado, el sistema le permitirá crear nuevos espacios de parqueo.**

#### 

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-013* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador para poder crear nuevos espacios de parqueo dentro de su parqueadero virtual, mientras esté dentro del sistema.* |
| ***Entradas*** | *Información del nuevo espacio de parqueo (identificación, categoría, estado, costo, descuento, etc.).* |
| ***Salidas*** | *Espacio de parqueo creado exitosamente y guardado en la BD.* |

#### **FR-016 Cuando el administrador esté logueado, el sistema le permitirá visualizar los espacios de parqueo existentes.**

#### 

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-016* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador para poder visualizar los espacios de parqueo creados, en el cual podrá entre otras cosas, ver el estado en que se encuentran los espacios de parqueo para su alquiler.* |
| ***Entradas*** | *Solicitud de visualización.* |
| ***Salidas*** | *Visualizar espacios de parqueo.* |

#### **FR-017 Cuando el administrador esté logueado, el sistema le permitirá modificar los espacios de parqueo existentes.**

#### 

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-017* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador para poder modificar los espacios de parqueo existentes, con la finalidad de cambiar cualquier atributo.* |
| ***Entradas*** | *Datos del espacio de parqueo a modificar.* |
| ***Salidas*** | *Datos del espacio de parqueo actualizados en la BD.* |

#### **FR-018 Cuando el administrador esté logueado, el sistema le permitirá eliminar los espacios de parqueo existentes.**

#### 

|  |  |
| --- | --- |
| ***Id. Requerimiento*** | *FR-018* |
| ***Descripción*** | *Es el proceso que debe realizar el administrador para poder eliminar los espacios de parqueo existentes.* |
| ***Entradas*** | *Solicitud de eliminación de espacio de parqueo.* |

Modelo Vista 4 + 1

# Vista de Escenarios

En esta vista se va a representar los casos de uso y va a tener la función de unir y relacionar las otras 4 vistas, con lo que tendremos una coherencia de componentes, clases, paquetes, etc. para realizar cada caso de uso.

## Diagrama de Casos de Uso

Como se refleja en la **Figura 2** es el diagrama de casos de uso realizado en UML, en el cual se muestra a manera general la vista del sistema de parqueadero, definido desde la perspectiva del usuario.

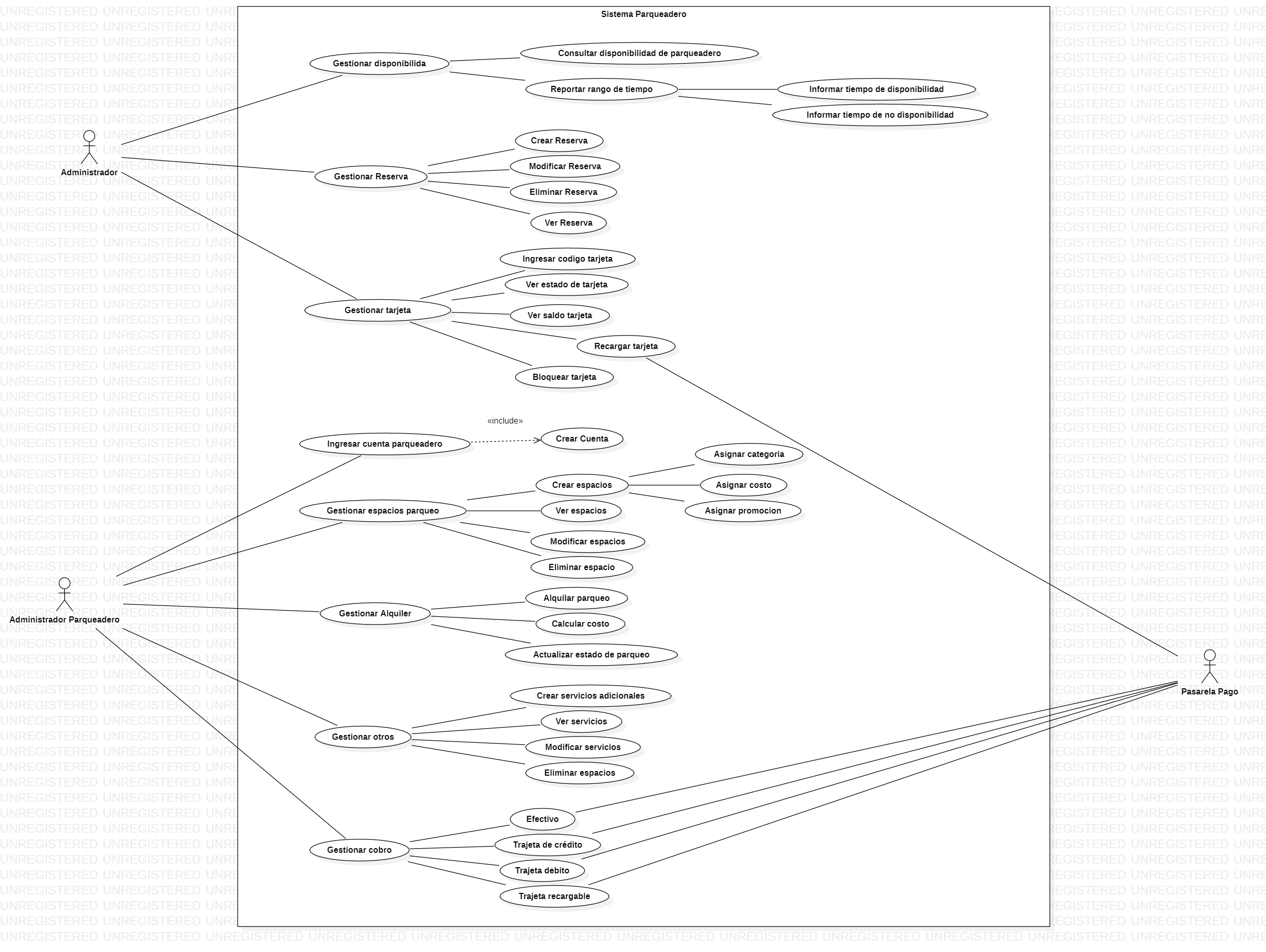


Figura . Diagrama Caso de Uso en UML

# Vista Lógica

En esta vista se proporciona la descomposición del sistema en una serie de abstracciones, tomadas del dominio del problema en forma de objetos o clases. Aquí se aplican los principios de abstracción, encapsulamiento y herencia.

## Diagrama de Clases

En este diagrama hemos descompuesto en Entidades y los Casos de uso, lo cuales estan relacionados, en las clases incluimos los atributos y metodos correspondientes. En los casos de uso definimos los que se ha implementado, como son: Inicial Sesión y Crear Cuenta, Crear Espacio, Eliminar Espacio, Modificar Espacio, Visualizar Espacio.

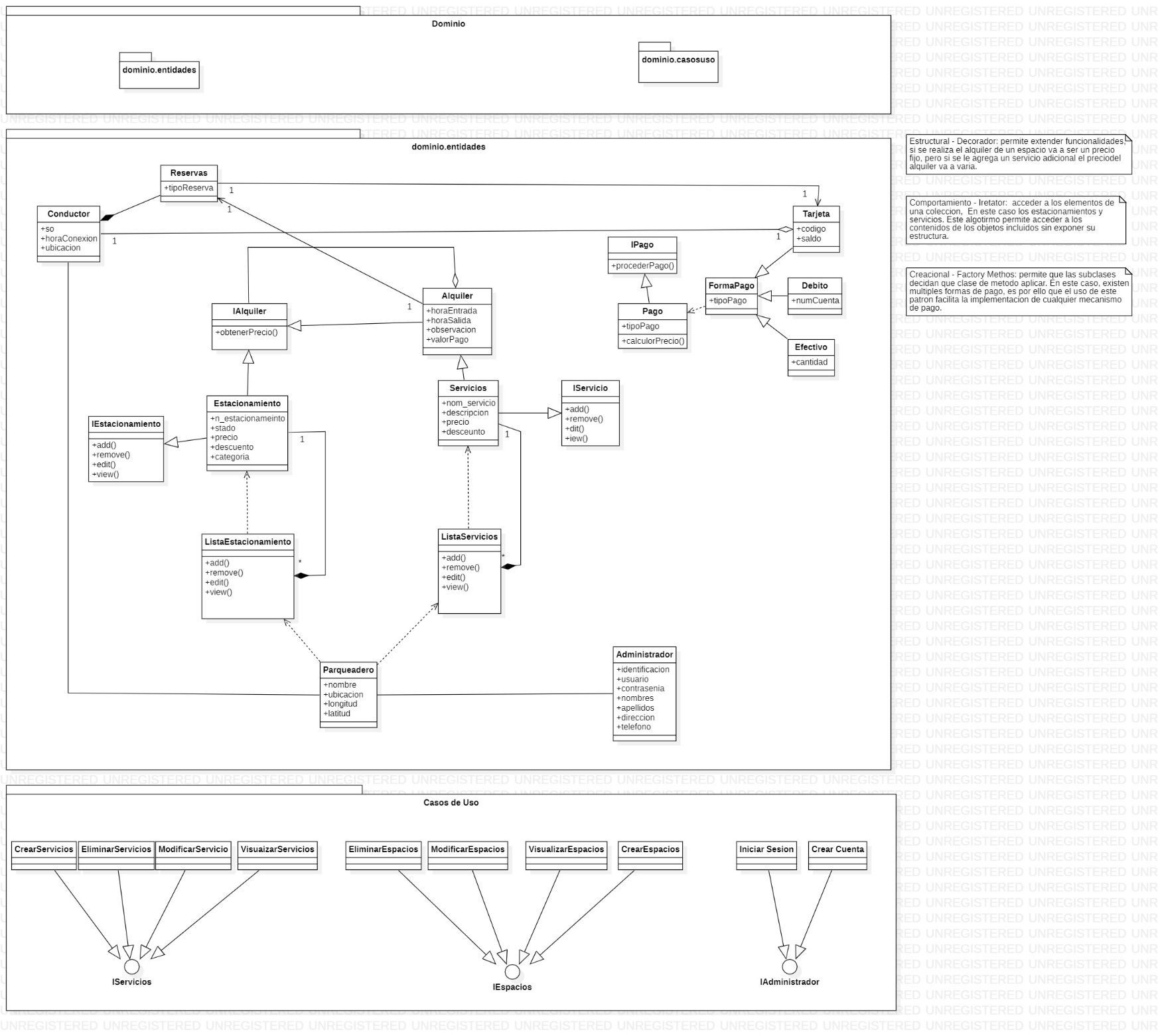


Figura .Diagrama de Clases en UML

En la **Figura 3** ya se puede identificar los patrones de diseño que hemos considerado importantes debido a sus características, tales como:

**Patrón Estructural**

Decorador: permite extender funcionalidades, si se realiza el alquiler de un espacio va a ser un precio fijo, pero si se le agrega un servicio adicional el precio del alquiler varia.

**Patrón de Comportamiento**

Iterator: este permite acceder a los elementos de una colección. En este caso los estacionamientos y servicios. Este algoritmo permite acceder a los contenidos de los objetos incluidos sin exponer su estructura.

**Patrón Creacional**

Factory Method: permite que las subclases decidan qué clase del método aplicar. En este caso, existen múltiples formas de pago, es por ello que el uso de este patrón para facilitar la implementación de cualquier mecanismo de pago.

## Diagrama de Secuencia

En esta vista se trata de representar una sucesión de interacciones entre las clases o instancia de objetos. En las siguientes **Figuras 4, 5, 6, 7, 8** se ilustra el procesamiento descrito en los escenarios de algunos casos de uso.

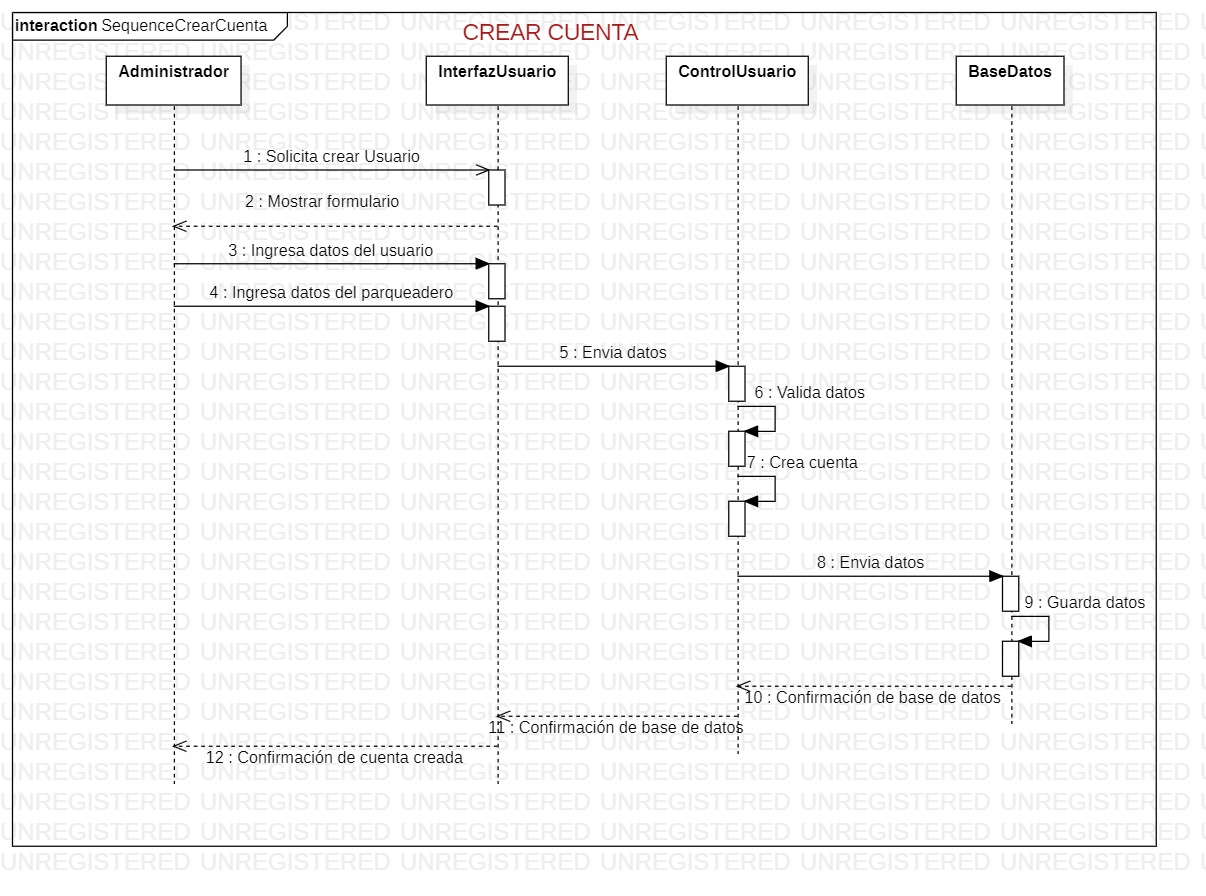


Figura .Diagrama de Secuencia para Crear Cuenta

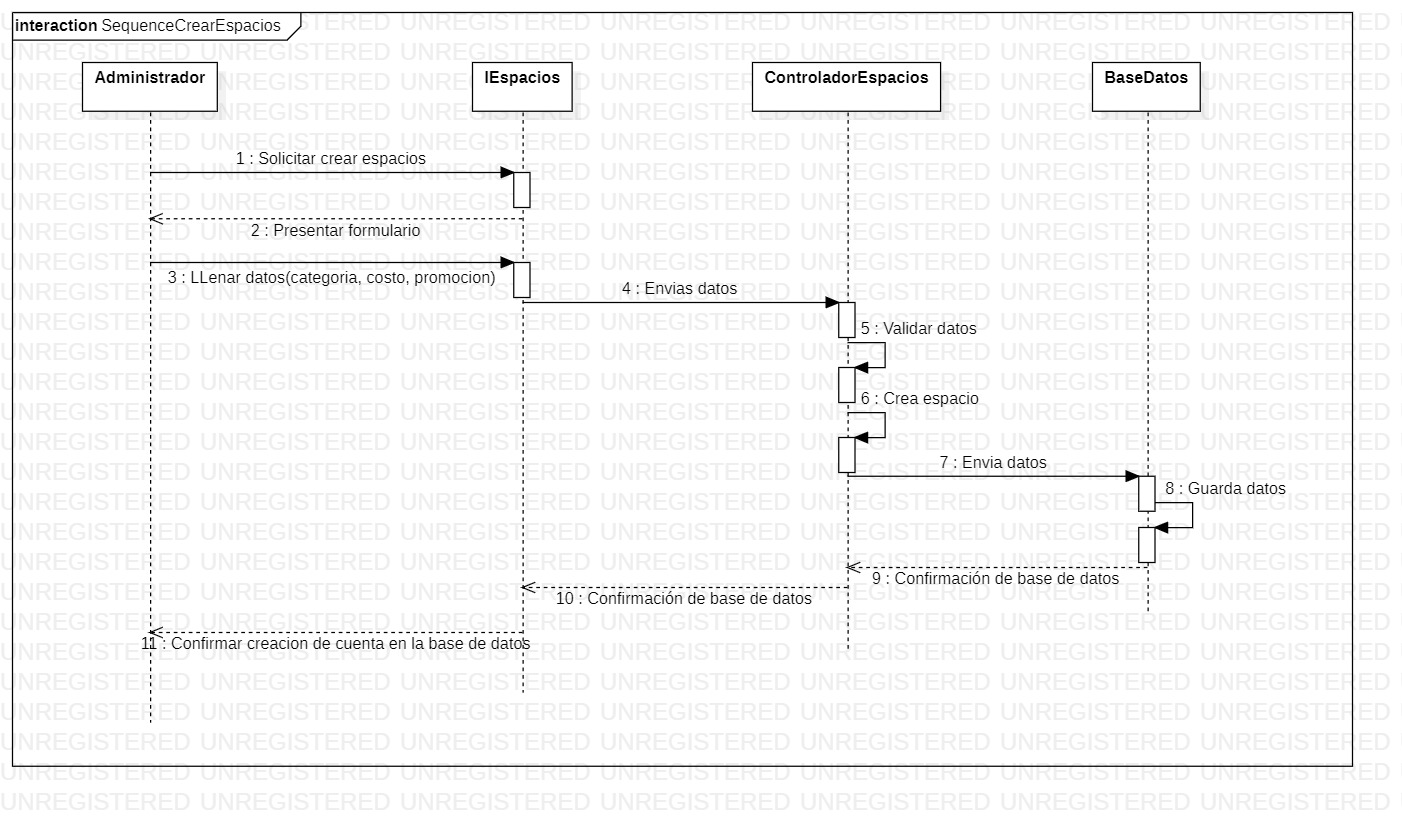


Figura . Diagrama de secuencia para Crear espacios

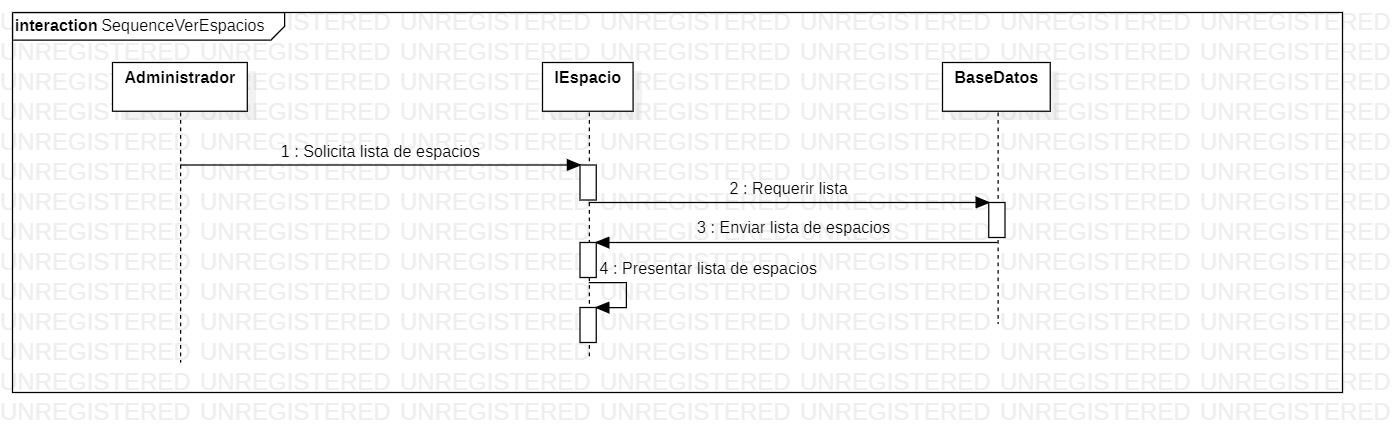


Figura . Diagrama de secuencia para Eliminar espacios

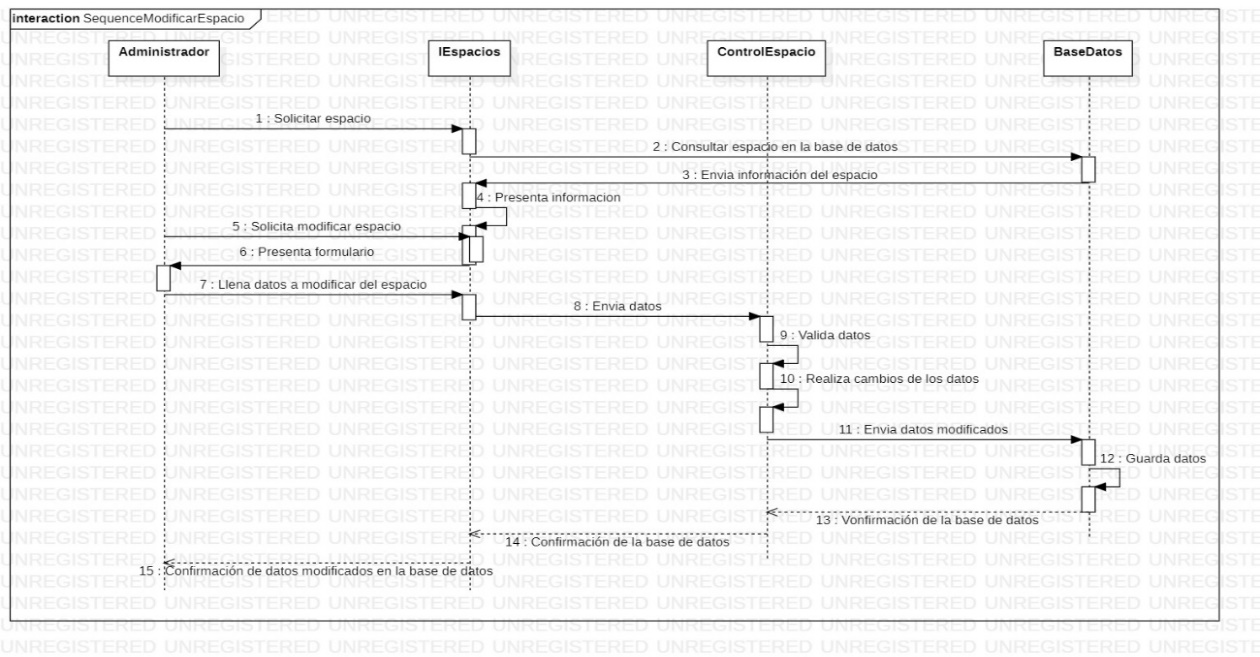


Figura . Diagrama de secuencia para Modificar espacios

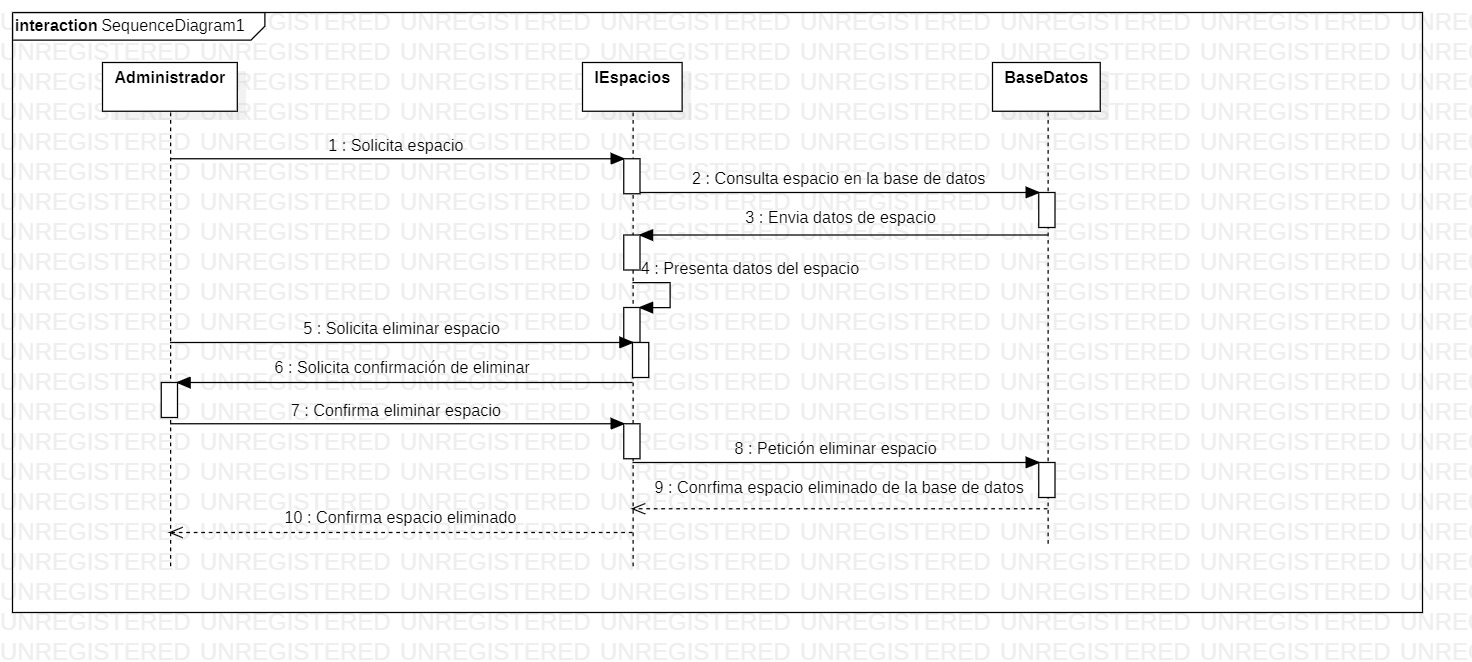


Figura . Diagrama de Secuencia para Eliminar Secuencia

# Vista de Despliegue

En esta vista se muestra el sistema desde la perspectiva del programador, mostrando como esta dividido el sistema en componentes y las dependencias que hay entre los mismos.

## Diagrama de Componentes

Aquí se presenta como un sistema de software ya dividido en componentes, demostrando las dependencias entre estos.

Aquí se utilizo los principios de componentes: para los componentes de cohesión y acoplamiento de componentes:

* Principio de reutilización común: asegurar que las clases de un componente son inseparables. Además, ayuda a no depender de clases que no se utilice.
* Principio de dependencias estables: evaluar métricas de estabilidad. Fan-in / Fan-out.
* Principio de estación estable: evaluar medidas de atracción.

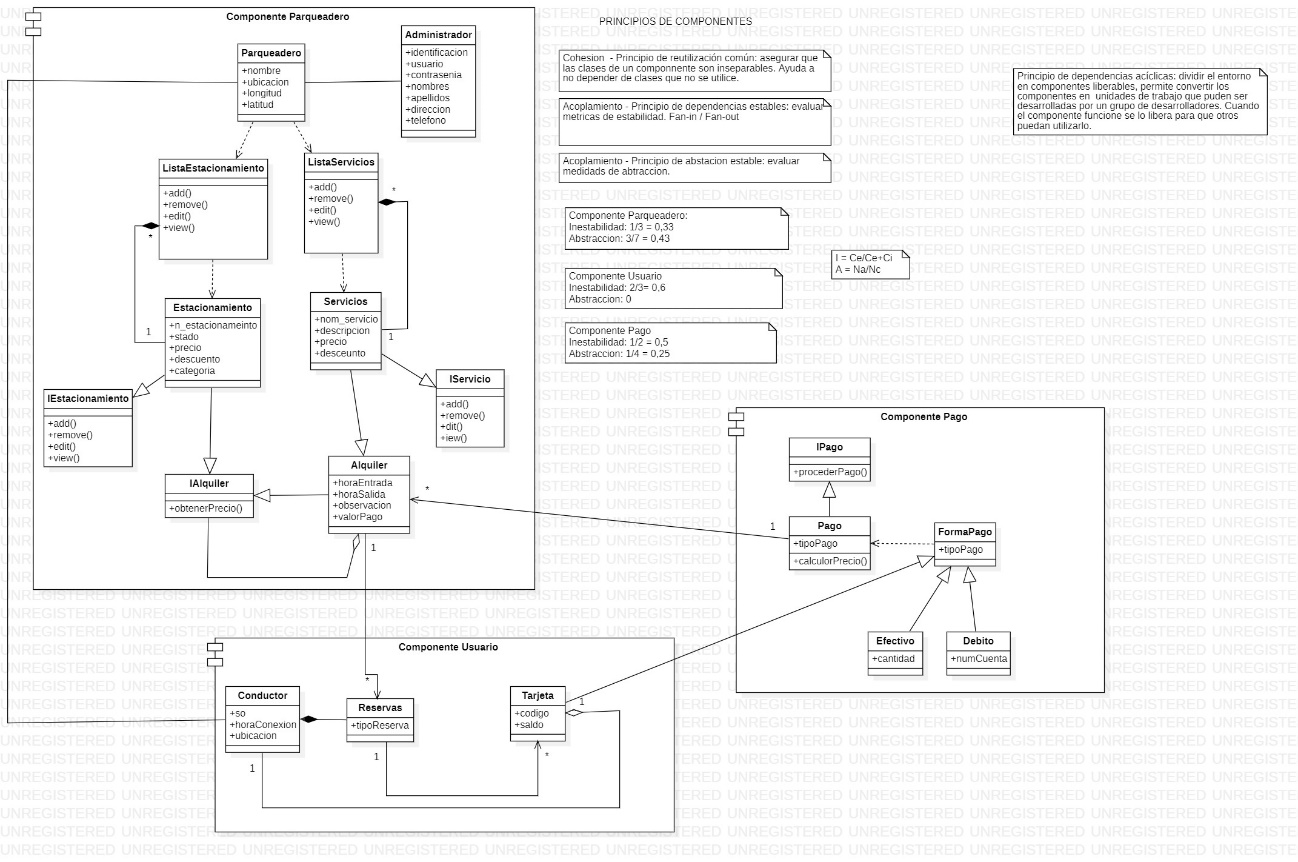


Figura . Diagrama de Despliegue en UML

Luego de terminar con el diagrama, se evalúa las métricas de abstracción y la inestabilidad obtenidas en cada componente.

Para sacar la abstracción, se utiliza la siguiente ecuación:

*Na: El número de clases abstractas dentro del componente*

*Nc: El número de clases dentro del componente*

Para sacar la inestabilidad, se utiliza la siguiente ecuación:

De acuerdo con las ecuaciones, obtuvimos como resultado lo siguiente:

|  |  |  |
| --- | --- | --- |
|  | Abstracción | Inestabilidad |
| Componente Parqueadero | A = 3/7 = 0,43 | I = 1/3 = 0,3 |
| Componente Usuario | A = 0 | I = 2/3 = 0,6 |
| Componente Pago | A = 1 / 4 = 0,25 | I = 1 / 2 = 0,5 |

Por lo tanto, se puede observar que cada uno de los componentes está dentro del rango [0, 1] de estabilidad y de abstracción.

# Vista de Procesos

En esta vista se muestran los procesos que hay en el sistema y la forma en que se comunican estos procesos es decir representa desde la perspectiva de integrador de sistema, el flujo de trabajo.

## Diagrama de Actividad

En la **Figura 9** se ilustra el flujo de trabajo que tiene Crear Cuenta y Crear Espacios, estas se encuentran en el diagrama de casos de uso definidas anteriormente.

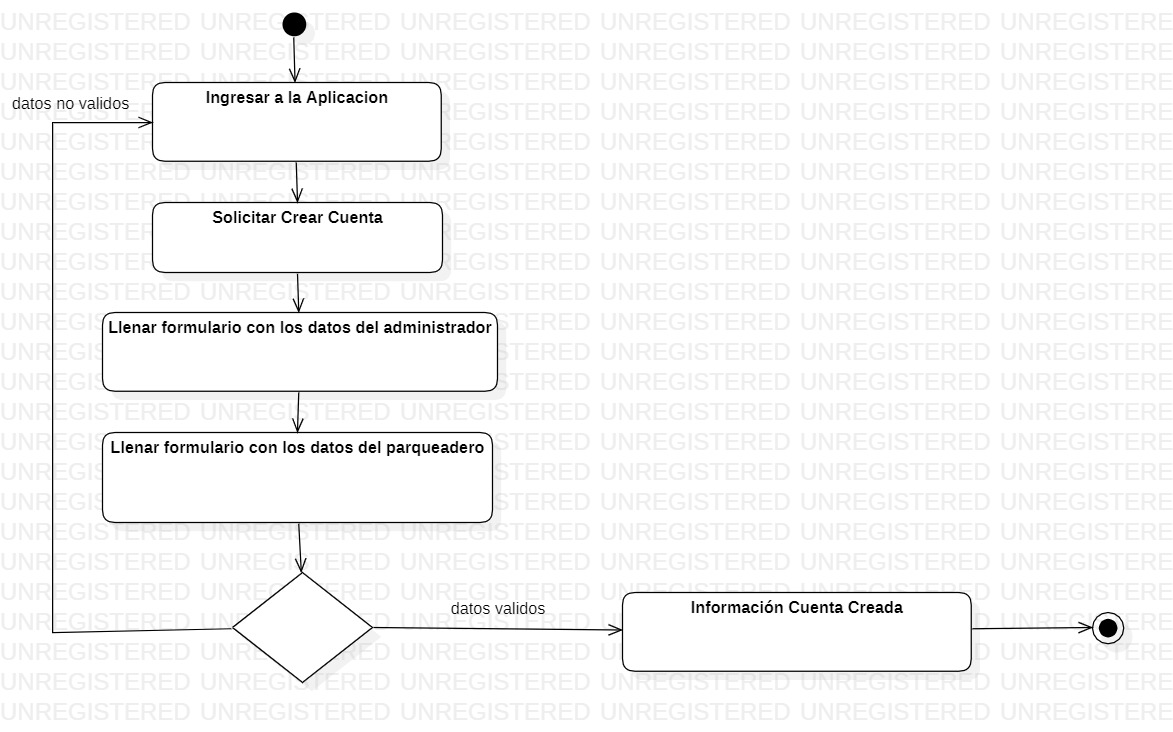


Figura . Diagrama de actividades para Crear cuenta

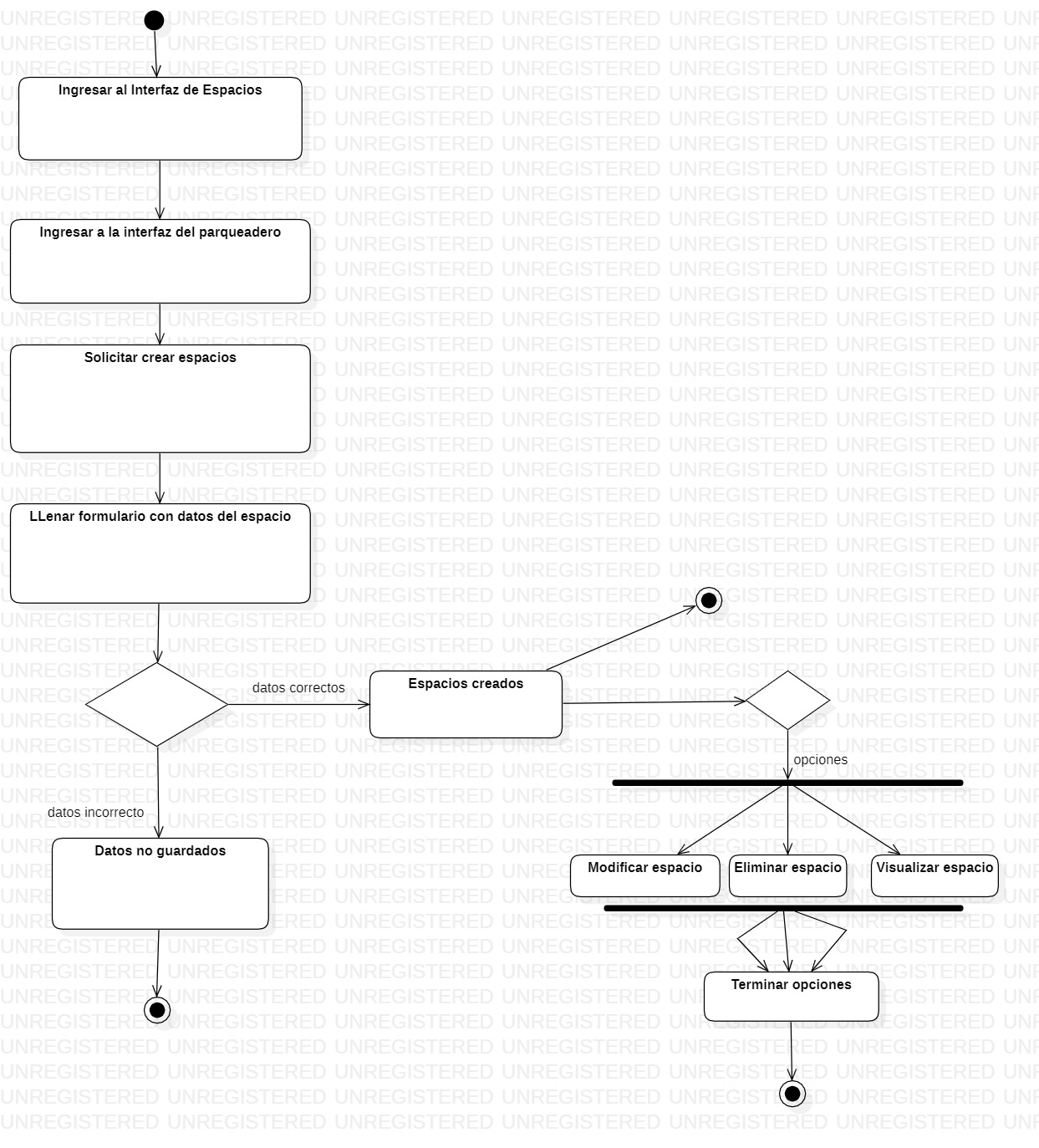


Figura . Diagrama de actividades para crear espacio

# Vista Física

En esta vista se muestra los componentes físicos del sistema, así como las conexiones entre los componentes que forman parte de la solución (incluyendo los servicios).

## Diagrama de Despliegue

En este diagrama se ha identificado un principio de dependencias acíclicas, que trata de dividir el entorno en componentes liberables, permite convertir los componentes en unidades de trabajo que pueden ser desarrolladas por un grupo de desarrolladores. Cuando el componente funcione se lo libera para que otros puedan utilizarlo.

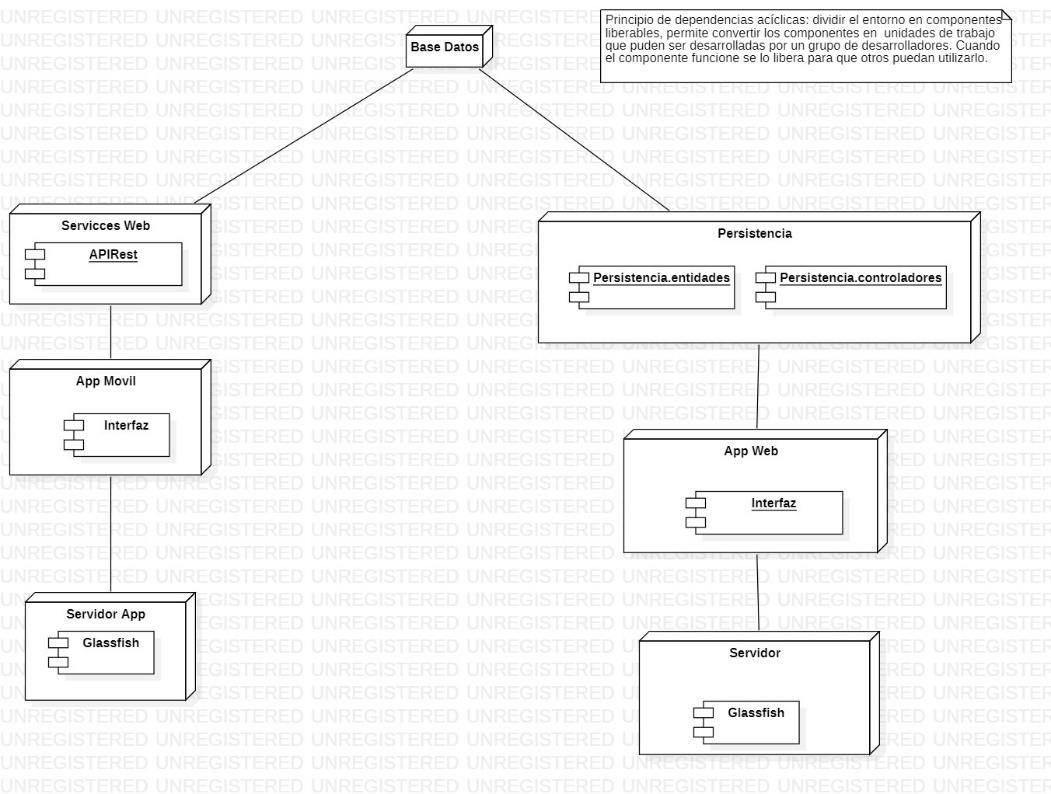


Figura .Diagrama de Despliegue en UML

Modelo de base de datos

Este modelo representa las entidades utilizadas para la implementacion de Crear Cuenta y de Crear Estacionamiento, se incluye las relaciones y las limitaciones qque se determinan como se almacenan los datos.

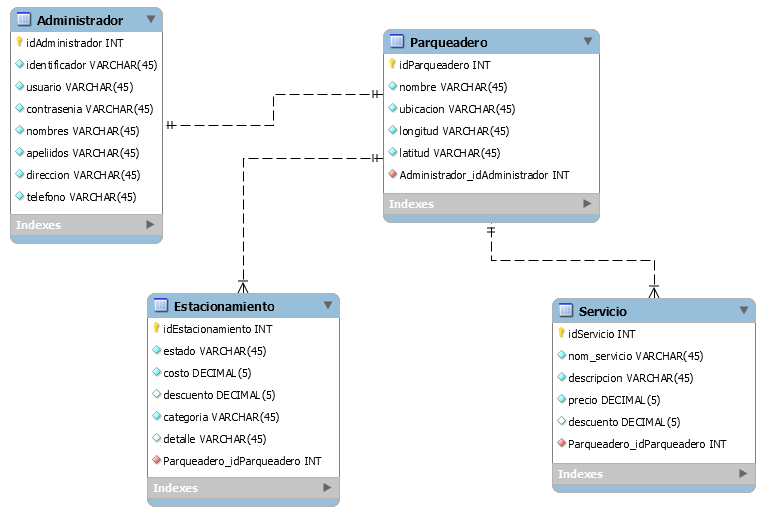


Figura . Modelo de la Base de datos en Word Beach Mysql