|  |  |
| --- | --- |
| Geo Parking | Modelado del Sistema  2014 |

# Control de la documentación

### Control de la Configuración.

|  |  |
| --- | --- |
| Título: | Modelado del Sistema |
| Referencia: | GeoP\_Proyecto\_ModeladoDelSistema.docx |
| Autores: | Lucas Toneatto |
| Fecha: | 30/09/2014 |

### Histórico de Versiones.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versión | Fecha | Estado | Responsable | Cambios |
| 1.0\_DraftA | 30/09/2014 | Pendiente de Revisión | Lucas Toneatto[autor] |  |
|  |  |  |  |  |
|  |  |  |  |  |

Contenido

[Control de la documentación 2](#_Toc400185527)

[Control de la Configuración. 2](#_Toc400185528)

[Histórico de Versiones. 2](#_Toc400185529)

[INTRODUCCIÓN 5](#_Toc400185530)

[MODELO DE NEGOCIO 6](#_Toc400185531)

[MODELO DE INTERACCION 7](#_Toc400185532)

[Búsqueda Playas por Ciudad 7](#_Toc400185533)

[Búsqueda de Playas por Ciudad por Filtro 7](#_Toc400185534)

[Login de Usuario 8](#_Toc400185535)

[Registro de Usuario 9](#_Toc400185536)

[Registro de Playa 10](#_Toc400185537)

[MODELO DE ARQUTECTURA/DEPLIEGUE 11](#_Toc400185538)

[Servidor 11](#_Toc400185539)

[ Presentación 11](#_Toc400185540)

[ Negocio 11](#_Toc400185541)

[ Entity Framework 11](#_Toc400185542)

[ SQL Server 11](#_Toc400185543)

[Cliente Web 11](#_Toc400185544)

[Cliente Móvil 12](#_Toc400185545)

[MODELO DE ARQUTECTURA AJAX 13](#_Toc400185546)

[MODELO DE COMPONENTES 14](#_Toc400185547)

[Vista de Componentes 14](#_Toc400185548)

[Vista de Datos 14](#_Toc400185549)

[Vista de Negocio 15](#_Toc400185550)

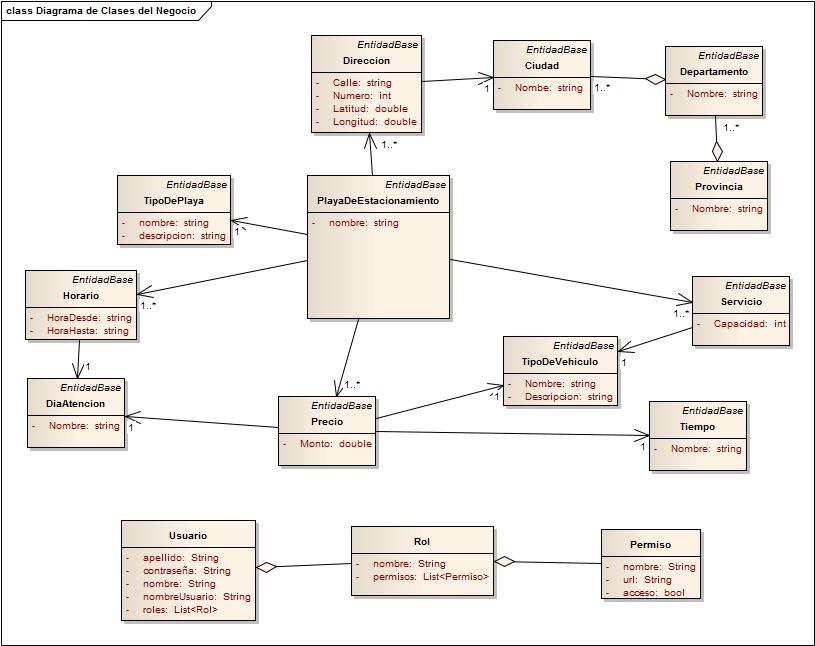
[Vista de WebService 16](#_Toc400185551)

[Vista de Presentación 16](#_Toc400185552)

# INTRODUCCIÓN

El siguiente documento lo que trata de mostrar es la distribución de los principales componentes utilizados y los cuales componen el sistema GeoParking. Las imágenes a continuación son pequeños modelos que tratan de esquematizar la forma en que se estructura el sistema.

# MODELO DE NEGOCIO



Como se puede observar el modelo negocio esta apuntado a abarcar todos aquellos elementos y características que se destacan en el negocio al que apunta GeoParking.

El elemento principal de este negocio es la “Playa de Estacionamiento” y de ella es que derivan los demás conformando sus características principales.

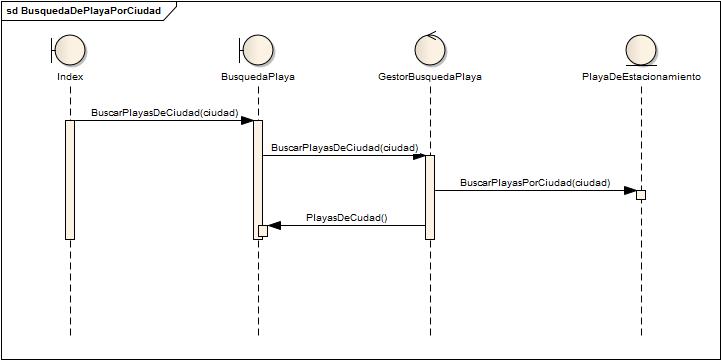
Y como todo sistema el modelo de negocio también abarca el usuario, el cual es quien interactúa con esos elementos y puede generar u obtener información a partir de los mismos.

# MODELO DE INTERACCION

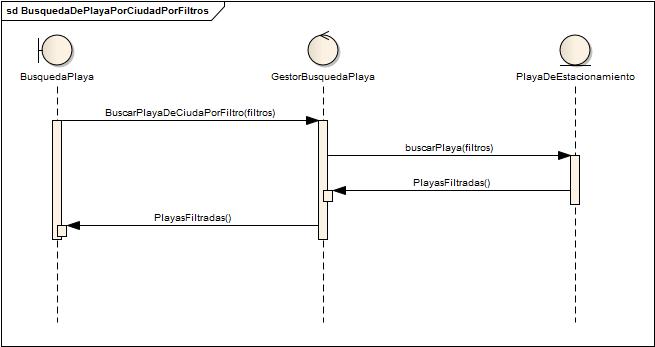
En los siguientes diagramas de secuencia lo que se trata de mostrar es la interacción general de como los componentes del sistema interactúan entre sí para poder brindar la funcionalidad que se tiene como objetivo.

EL objetivo de estos diagramas es brindar una trazabilidad a nivel de código que permita a simple vista observar, captar y analizar el impacto el cambio de alguno de los componentes intervinientes en cada funcionalidad, permitiéndole todo esto, obtener una visión más precisa del impacto de los cambios a realizar.

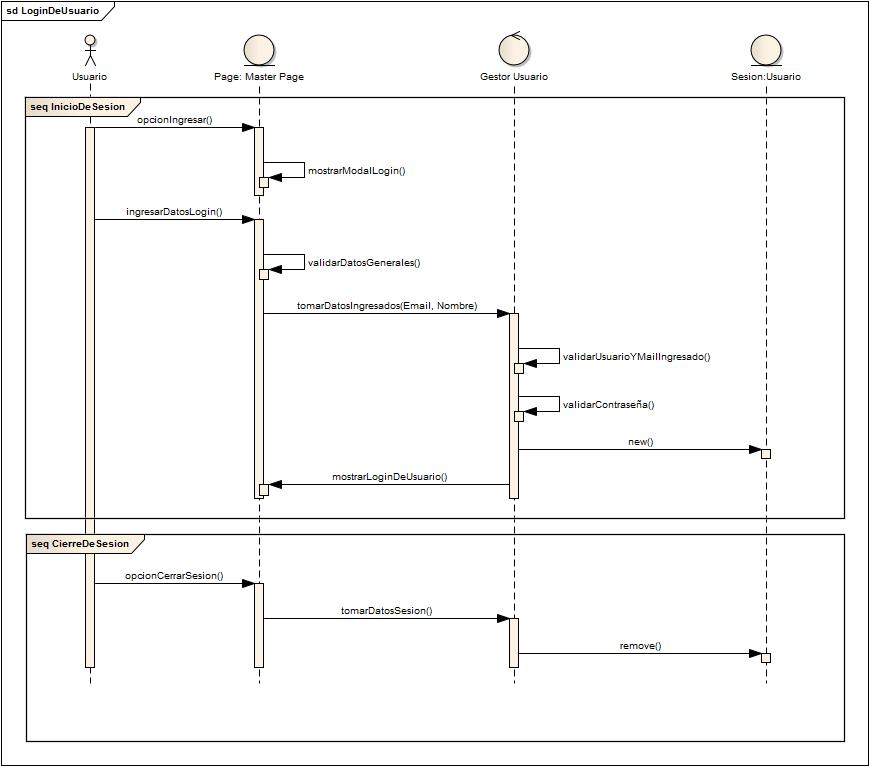
### Búsqueda Playas por Ciudad - Web



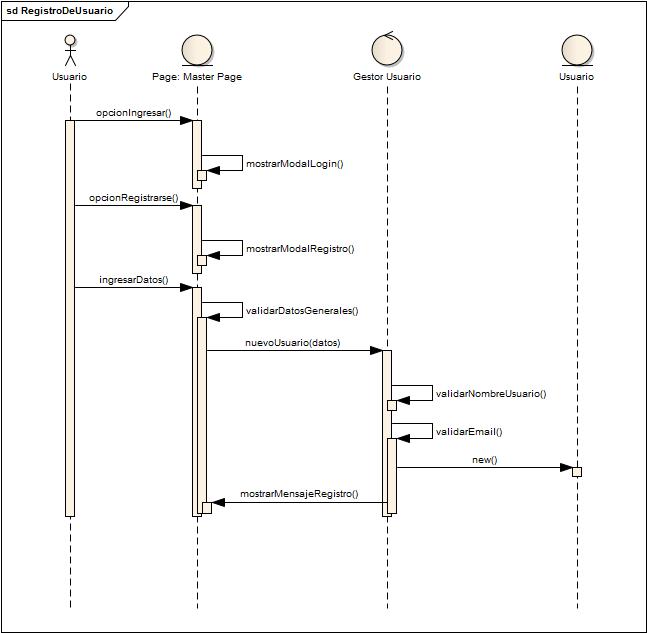
### Búsqueda de Playas por Ciudad por Filtro - Web



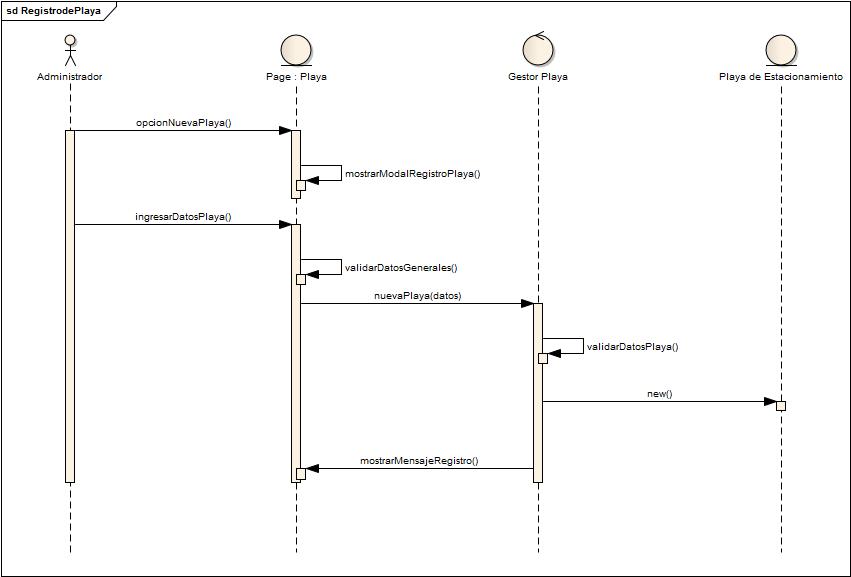
### Login de Usuario - Web



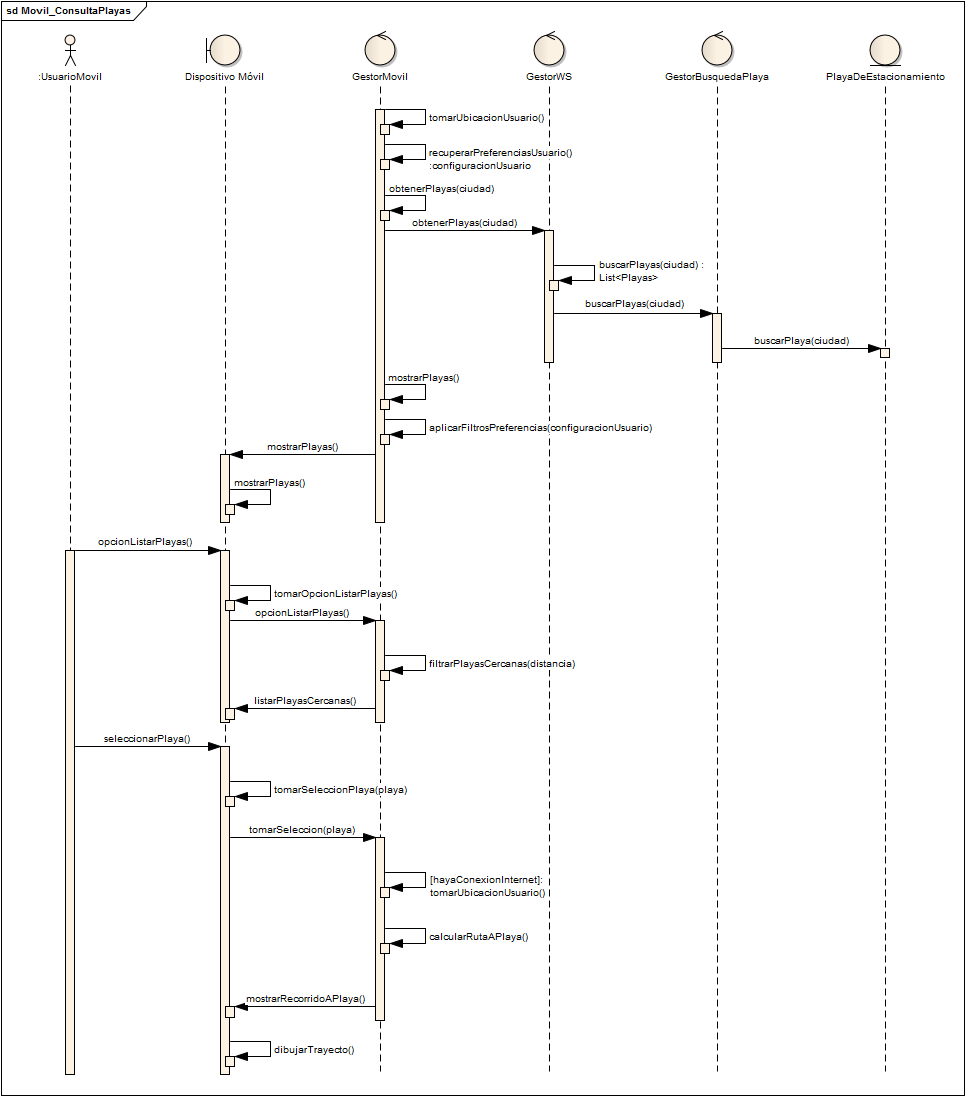
### Registro de Usuario - Web



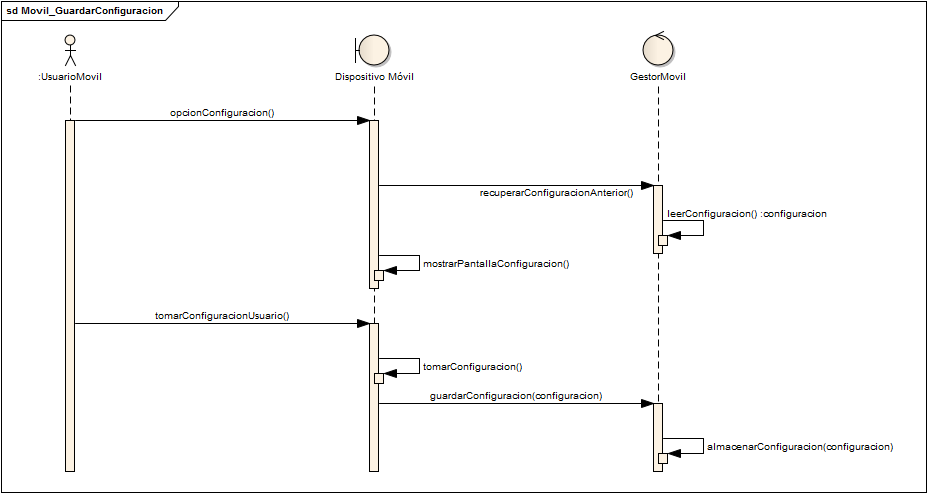
### Registro de Playa - Web



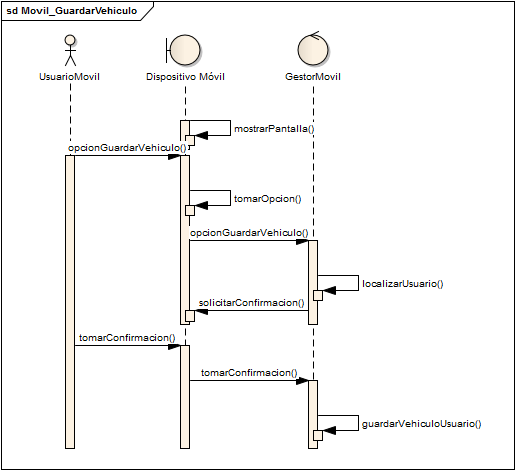
### Consulta Playas – Móvil



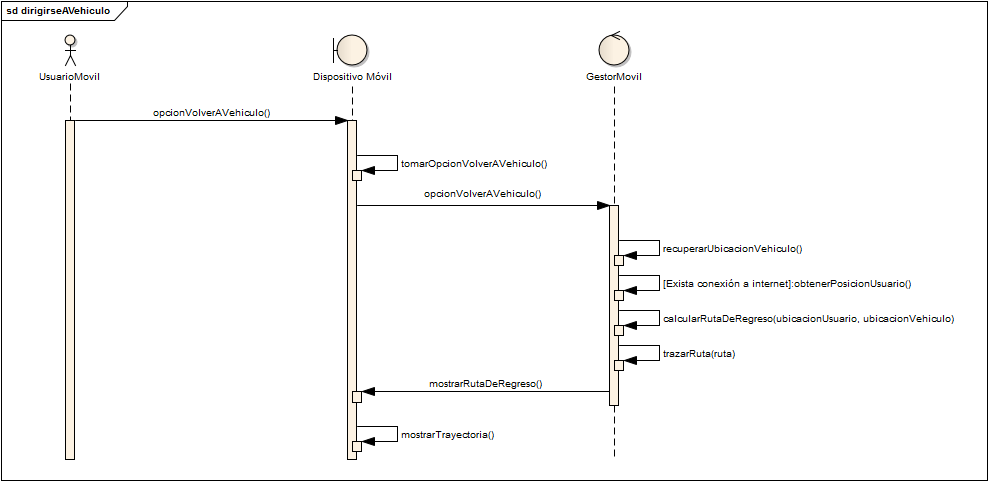
**Guardar Configuración – Móvil**

****

**Guardar Vehículo – Móvil**



**Dirigirse a Vehículo – Móvil**



# MODELO DE ARQUTECTURA/DEPLIEGUE

La arquitectura que presenta el sistema GeoParking es como el diagrama que se muestra a continuación. La arquitectura se organiza en 3 bloques principales que están en contante interacción: El servidor de la aplicación, el cliente Web y el Cliente Móvil.

## C:\Users\Lukas\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Modelo de arquitectura.jpgServidor

En este nodo se despliegan 4 componentes, los cuales están encargados de realizar o brindar distintas funciones, es por ello que la arquitectura a desplegar en el servidor está organizada en capas. Las capas son las siguientes:

* Presentación: es el componente encargado de albergar todo aquello que genere una forma de interacción con el usuario, será aquella interfaz de comunicación a través de la cual se ofrezca las funcionalidades del sistema.
* Negocio: es el componente encargado de albergar todo aquello que establezca la manera en que funcionara el sistema, y las reglas que rigen ese comportamiento. Este componente se subdivide a su vez en:
  + **Entidades**: contendrá todos aquellos elementos que formen parte del problema que trata el sistema como a su vez también elementos para la solución de los mismos. Dichos elementos forman parte del modelo del negocio como del modelo de solución.
  + **Reglas de Negocio**: contendrá todo aquellos agentes que regulan el funcionamiento del sistema, y maneja el mundo de entidades que lo conforman.
  + **Web Service**: será el componente que brinde los servicios del sistema a la aplicación móvil, la cual está desarrollada de forma externa al sistema.
* Entity Framework: es aquel componente encargado del mapeo Objeto-Relacional de las entidades del sistema.
* SQL Server: es la instancia de la Base de Datos que alberga toda la información del sistema.

## Cliente Web

Este nodo contendrá el **navegador** a través del cual se realiza la conexión con nuestro sistema mediante de peticiones Http (Internet).

## Cliente Móvil

Este nodo contendrá la **aplicación móvil** desarrollada para que consuma los servicios del Web Service deployado en el Servidor y así podrán brindar las funcionalidades del sistema.

## SEGURIDAD

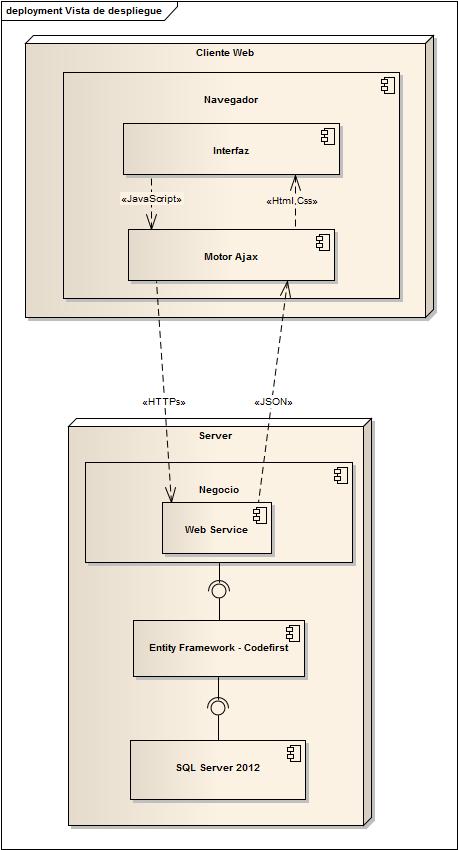
El manejo de la seguridad de la aplicación en cuanto al acceso de la información que el sistema obtiene, procesa y muestra, está basado en la autenticación de quien se comunica con el sistema ya sea web o desde la aplicación móvil.

En cuanto a la aplicación móvil la seguridad de acceso a la información esta manejada por el manejo de roles y permisos para los diferentes usuarios que interactúan con el sistema. Cada funcionalidad está restringida a roles específicos.

En la aplicación móvil, la obtención de la información que se muestra en la aplicación es obtenida a través de los servicios del componente WebService del sistema. Todas las peticiones que realiza nuestra aplicación móvil envían un clave al servicio web para que valide que es nuestra aplicación la que está realizando esas peticiones y no otra ajena al sistema.

# MODELO DE ARQUTECTURA AJAX

Aquí se trata de mostrar de qué manera se hace uso de la tecnología Ajax en la implementación de nuestro proyecto GeoParking. En el diagrama se puede visualizar los componentes intervinientes en el flujo de datos.

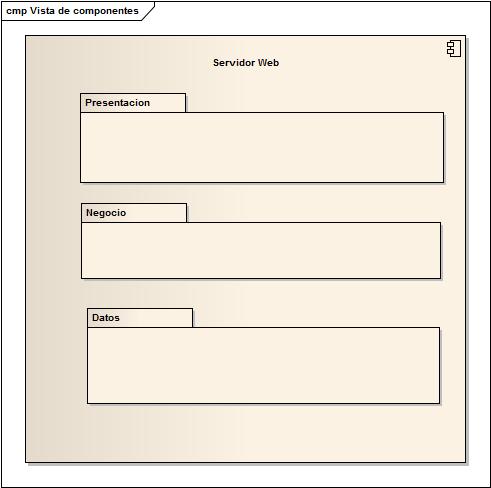


# MODELO DE COMPONENTES

## Vista de Componentes

Aquí se puede observar como nuestro sistema el componente principal está constituido o estructura en una división por capas, las cuales son:

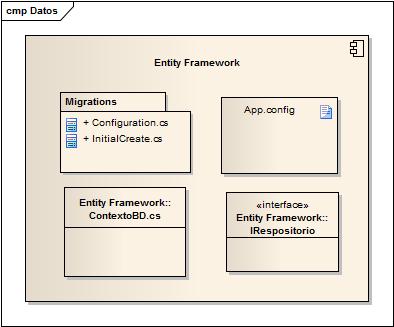
1. **Datos**: capa dedicada a la materialización y desmaterialización de los datos de la aplicación.
2. **Negocio**: capa dedicada a aplicar la lógica del negocio en el sistema y todas aquellas reglas que rigen su funcionamiento.
3. **Presentación**: capa dedicada a brindar una interfaz visual con la cual los usuarios interactúen con el sistema.



## Vista de Datos

Aquí se puede observar como esta capa está basada en la tecnología Entity Framework, empleando el modelo Codefirst. Para ello estos son los principales componentes que permiten en funcionamiento de esta capa del sistema:

* **Migrations**: este paquete contiene dos clases muy importantes en el funcionamiento de Entity Framework
  + **Configuration.cs**: tiene establecido el contexto de la BD donde se montara la conexión para el mapeo de los objetos.
  + **InitialCreate.cs**: tiene establecido la creación del modelo en memoria de los objetos para el posterior mapeo.
* **App.config**: tiene establecido la conexión a la instancia de la Base de Datos. Y además la versión de Framework con la que debe realizar el mapeo objeto relacional.
* **ContextoBD.cs**: crea el contexto en memoria en relación a los objetos presentes en la Base de Datos. Establece contexto particulares para cada objeto del sistema.
* **IRepositorio:** interfaz que establece las acciones que se pueden realizar sobre cada objeto del sistema en cada contexto.



## Vista de Negocio

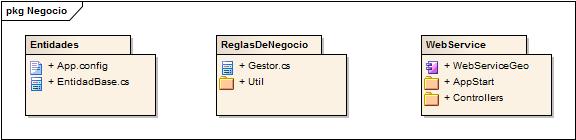
Aquí se puede observar como esta capa está estructurada y subdividida en tres paquetes, los cual tienen funcionalidades muy diferentes pero que así ves se relacionan para cumplir un objetivo en común, brindar la funcionalidad necesaria al usuario aplicando las distintas reglas del negocio. En esta capa se divide en:

1. Entidades
2. ReglasDeNegocio
3. WebService

**Entidades:** alberga todas las clases que representan los objetos manejados en la aplicación.

**ReglasDeNegocio:** alberga todos los controladores que aplican las distintas reglas de negocio sobre los objetos del sistema.

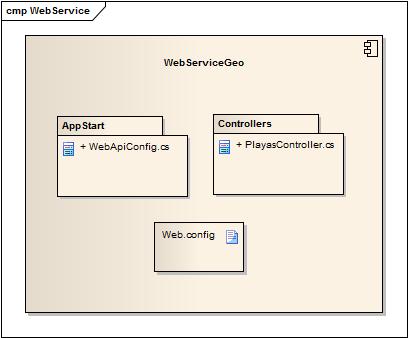
**WebService:** alberga la aplicación que se corresponde al webService para poder ofrecer la funcionalidad del sistema a la aplicación móvil.



## Vista de WebService

Aquí se puede observar que el componente principal es el servicio web WebServiceGeo, el cual interactúa con los demás componentes de la capa de negocio para que la funcionalidad que provee el sistema esté disponible a través de la aplicación móvil. Este servicio web se compone de:

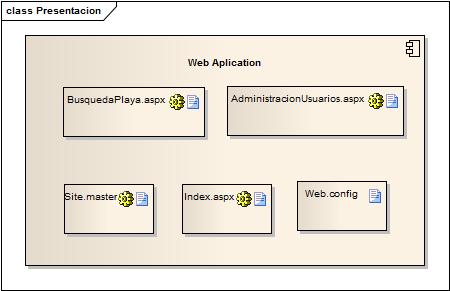
* **WebApiConfig.cs:** establece el formato de ruta con la cual se puede acceder a través de un protocolo de internet a las distintas funcionalidades que ofrece el WebService.
* **Web.config:** establece la conexión con la BD.
* **PlayasController.cs:** ejemplo de unos de los controladores que brinda la funcionalidad del WebService, es a estos métodos a los cuales accede la aplicación móvil a través del formato de ruta establecida en WebApiConfig.



## Vista de Presentación

Aquí se puede observar como esta capa del sistema está conformada por una aplicación web donde se pueden observar componentes como:

* **Site.master:** platilla modelo de interfaz de donde heredan las demás páginas.
* **Index.aspx**: ejemplo de página que hereda de la master.
* **Web.config**: configuración de distintos aspectos de la aplicación web.



# MODELO DE PATRONES

Aquí se muestra el diagrama de implementación de los patrones utilizados en el desarrollo del producto GeoParking.

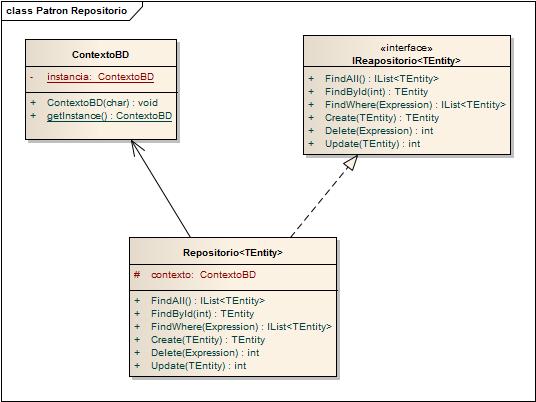
## Patron SingletonSingleton

**JUSTIFICACION**

La aplicación del patrón Singleton sobre la clase ContextoBD surge por:

* Necesidad de que exista una instancia única en todo el sistema; esto es porque a la hora de instanciarse un objeto del tipo ContextoBD requiere de un tiempo para levantar toda la estructura para relacionarse con la base de datos, y poder mapear todos los objetos. Si se instanciara cada vez que se lo requiere la performance de la aplicación decaería precipitosamente.
* Acceso desde todas las clases DAO: por cada clase entidad, necesitaremos una clase específica que realice todas las tareas relacionadas a la base de datos a través de las funcionalidades que le preste el contexto.

## Repositorio



**JUSTIFICACION**

* En pocas palabras, un repositorio es un mediador entre el dominio de la aplicación y los datos que le dan persistencia. Con este planteamiento podemos pensar que el usuario de este repositorio no necesitaría conocer la tecnología utilizada para acceder a los datos, sino que le bastaría con saber las operaciones que nos facilita este “mediador”, el repositorio.
* Utilizando como tecnología de acceso a datos el ORM Entity Framework, con una aproximación “code-first”, se podrá utilizar el repositorio para cualquier entidad compleja que se defina.