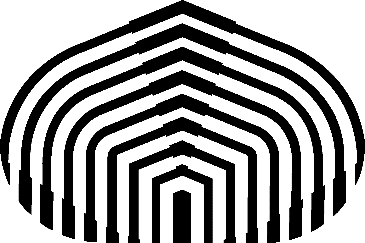
Universidad Simón Bolívar

Decanato de Estudios Profesionales

Departamento de Computación y Tecnología de la Información

CI3715 – Ingeniería de Software I

**PROYECTO DE ESTACIONAMIENTOS**

<Ucode>

Karla Alzuro <08-10037>

Gabriel Álvarez <09-10029>

Carlos Farinha <09-10270>

Francisco Martínez <09-10502>

Miguel Sarabia <09-10782>

Ricardo Vethencourt <0910894>

# INTRODUCCION

Este documento hace referencia al avance obtenido en el desarrollo del sistema de desarrollo de software para un fin específico referente a nuestro caso a un estacionamiento, con objetivos claros y desarrollados en el documento de visión dotado por la colegiatura profesoral

# **ESQUEMA DE TRABAJO**

Notación:

* KA: Karla Alzuro
* CF: Caros Farinha
* MS: Miguel Saravia
* RV: Ricardo Vethencourt
* GA: Gabriel Álvarez
* FM: Francisco Martínez
* <FUN> (): Función a hacer.
* DUML: Diagrama de clases del proyecto.
* TDD: Filosofía TDD, la cual tiene que ser aplicada al programar.
* PPP: Es necesario que se programe por parejas.
* EP: Todos debemos hacer esto, tenemos que indicar las fortalezas y debilidades de cada integrante del equipo, así como su aporte al proyecto.
* AD: Un grupo según lo indicado en el enunciado debe hacer un análisis de los otros 2 diagramas que se realizaron por los otros 2 grupos, ya que somos 7, agregamos a RL al grupo de CF, KA solo para el análisis, no es necesario que escribas ningún código RL.
* DOC == > Es la parte en la que hay que recopilar la documentación de código, la indicaciones de tdd, como fue la programación en parejas, los diagramas uml, etc.

División de tareas (Chores):

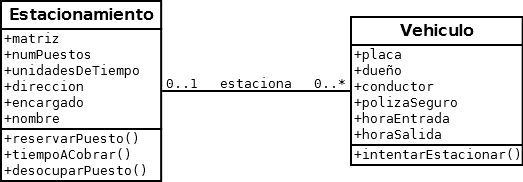
MS, GA == > ReservaPuesto(), DUML (Versión 1) , TDD, PP, EP.

CF, KA == > IntentarEstacionar(), AD, PP, EP.

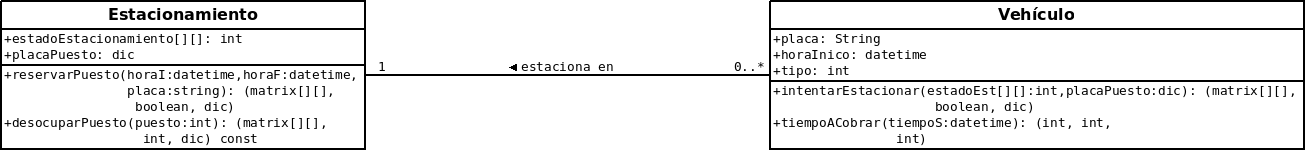
FM, RV == > TiempoaCobrar(),DesocuparPuesto(), DUML (Versión 2), PP, EP.

RL == > DOC, AD, PP, EP.

# **DIAGRAMAS UML (VERSION 1 Y 2)**



Versión 1 (MS, GA)



Versión 2 (FM, RV)

# **ANÁLISIS COMPARATIVO DE LOS DIAGRAMAS DE CLASES**

Ambos diagramas poseen la estructura de dos clases, Estacionamiento y Vehículo; sin embargo difieren entre sí en su estructura:

1. Estacionamiento: La identificación de la clase corresponde con el modelo que se presenta en el problema planteado en el proyecto.

En los atributos existe cierta diferencia, pues en un diagrama se refleja el apego a lo que se pide en principio en el proyecto, que es la matriz estadoEstacionamiento, que contiene el estado de cada puesto en una unidad de tiempo determinada y el diccionario placaPuesto, que relaciona el vehículo con el puesto; y en el otro se considera que deben agregarse los siguientes atributos: la matriz estacionamiento, numPuestos, unidadesDeTiempo, dirección, encargado, nombre; sin tomar en consideración el diccionario placaPuesto. Con estos atributos de alguna manera se estaría perdiendo la estructura especificada en el enunciado para establecer la relación entre la placa y el puesto, lo que conlleva a cambios en las estructuras de las funciones.

En los métodos, un diagrama incluye los métodos: ReserReservarPuesto() y DesocuparPuesto(), y en el otro diagrama además se incluye el método TiempoACobrar(); lo que implica una diferencia en el diseño, a parte de que la información de la hora de llegada y salida del estacionamiento se encuentra en la clase Vehículo, para ambos diagramas.

2. Vehículo: Al igual que la clase Estacionamiento, esta clase coincide con lo especificado en el enunciado.

Ambos diagramas contienen información de la placa y de la hora de llegada al estacionamiento dentro de los atributos de la clase, sin embargo en uno de los diagramas se reflejan atributos como: dueño, conductor, polizaSeguro, que no se encuentran en el enunciado pero que son considerados relevantes a la clase.

Las dos clases poseen el método intentarEstacionar(), sin embargo, como se resaltó antes, el método TiempoACobrar() está incluido en la clase Vehículo de uno de los diagramas.

La relación entre las clases Estacionamiento y Vehículo coinciden en que un estacionamiento puede estar asociado a 0 o más carros. Cabe destacar que en uno de los diseños, se dice que un vehículo está asociado exactamente a un estacionamiento, mientras que el otro diseño plantea que un vehículo puede estar asociado a 0 o 1 estacionamiento.

# **PROGRAMACIÓN POR PARES**

En cada prueba se define quien condujo, si hubo cambios se indicaran respectivamente, de lo contrario no hubo. Las dificultades encontradas en cada prueba son indicadas al igual que la manera en la que se resolvieron.

**Def ReservarPuesto (estacionamiento,tiempo,placa,dicPlacaPuesto)**

Conductor: Miguel

Navegante: Gabriel

Ninguna dificultad encontrada.

**def testReservaPuestoDicNoInicializado()**

Conductor: Miguel

Navegante: Gabriel

Ninguna dificultad encontrada.

**def testReservaPuestoTiemposIncorrectos()**

Conductor: Miguel

Navegante: Gabriel

Ninguna dificultad encontrada.

**def testReservaMatrizIncorrecta()**

Conductor: Miguel

Navegante: Gabriel

Ninguna dificultad encontrada.

**def testReservaPuestoPlacaNoInicializada()**

Conductor: Miguel

Navegante: Gabriel

Ninguna dificultad encontrada.

**def testReservaPuestoMatrizOcupada()**

Conductor: Gabriel

Navegante: Miguel

Ninguna dificultad encontrada.

**def testReservaPuestoMatrizReservadaDesocupado()**

Conductor: Gabriel

Navegante: Miguel

Ninguna dificultad encontrada.

**def testReservaPuestoMatrizReservadaOcupada()**

Conductor: Gabriel

Navegante: Miguel

Ninguna dificultad encontrada.

**def tiempoACobrar (placa, tEntrada, tSalida, placaPuesto)**

Navegante: Francisco

Conductor: Ricardo

Las dificultades en esta funcion se presentaron como dudas, las cuales fueron informadas por email al profesor Teruel. (Respecto a necesitar la hora de entrada)

**def desocuparPuesto (placa, tSalida, placaPuesto)**

Navegante: Ricardo

Conductor: Francisco

Ninguna dificultad encontrada.

**def testBasico()**

Test de funcionamiento básico

Fallo al primer intento, retorno en posición equivocada

Corrió bien al segundo intento

Navegante: Ricardo

Conductor: Francisco

**def testBasico2()**

Test de chequeo con matriz más llena

Corrió bien al primer intento

Navegante: Ricardo

Conductor: Francisco

**def testBasico3()**

Test de chequeo con múltiples valores para la salida

Corrió bien al primer intento

Navegante: Ricardo

Conductor: Francisco

**def testPlacaErronea()**

Test para probar caso con placa errónea

Fallo al primer intento, corregido con un try

Funciono bien al siguiente intento.

Navegante: Ricardo

Conductor: Francisco

**def testDiccionarioVacio()**

Probar darle un diccionario vacío

Funciono bien al primer intento

Navegante: Ricardo

Conductor: Francisco

**def intentarEstacionar (placa, tEntrada)**

Navegante: Carlos

Conductor: Karla

Básicamente la edición de esta función fue de simple manejo sin mayores dificultades, aunque persistieron dudas por errores ligeros de formulación de la función

# **DESCRIPCION DE LA EXPERIENCIA EN EQUIPO**

Roles y Responsabilidades de cada miembro:

GA: Programador, Diseñador, Documentación

KA: Programadora, Diseñadora, Analista de Diagramas, Documentación

CF: Programador, Diseñador, Analista de Diagramas, Organizador de repositorio

FM: Programador, Diseñador, Documentación

MS: Programador, Diseñador, Documentación

RV: Programador, Diseñador, Documentación

Dificultades Encontradas

Las dificultades encontradas en el proyecto fueron las siguientes:

1. Análisis de un diseño inicial ambiguo
2. Organización de trabajo
3. Comunicación grupal
4. Manejo de herramientas particulares entorpece ritmo de trabajo
5. Restricciones de firmas de las funciones