FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2

Segunda Entrega

[7507/9502] Algoritmos y Programación III
Curso 1 Segundo cuatrimestre de 2017

Grupo T10

Nombre	Padrón	Mail
Perez Ondarts, Flavio	96786	perezflavio94@gmail.com
Piersantolini, Matías	98299	matias.piersantolini@gmail.com
Ontiveros, Juan	98425	ontiverosfuertes@yahoo.com.ar
Aguirre, Ariel	100199	ariedro@gmail.com

Índice

1. Introducción

El presente informe reúne la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicación completa con interfaz gráfica de una versión simplificada del Monopoly, utilizando los conceptos del paradigma de la orientación a objetos vistos en el curso.

2. Supuestos

Para hacer esta parte del trabajo, se asumió que el tablero en el que se desarrollará el juego es siempre el mismo, es decir que los casilleros siempre están en las mismas posiciones que están en el enunciado. Se usó esto para hacer más simples las clases con las que se manejan los barrios y los servicios.

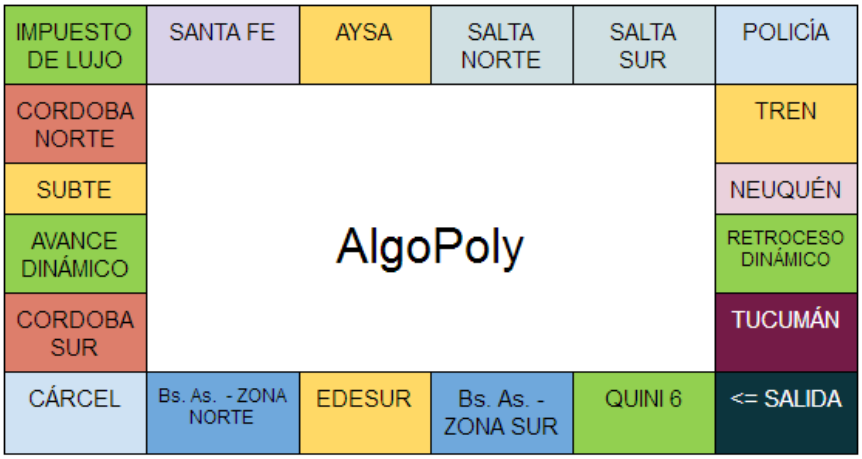

Figura 1: Tablero del enunciado.

3. Modelo de dominio

Ésta entrega cuenta con una implementación casi completa manejo general del juego, siguiendo las pruebas de integración proporcionadas por la cátedra, se pudo hacer un modelo que implementa bien las reglas del juego en el manejo de las propiedades en los barrios y servicios.

4. Diagramas de clase

A continuación se mostrarán dos diagramas de clase que ayudarán a representar el funcionamiento del programa.

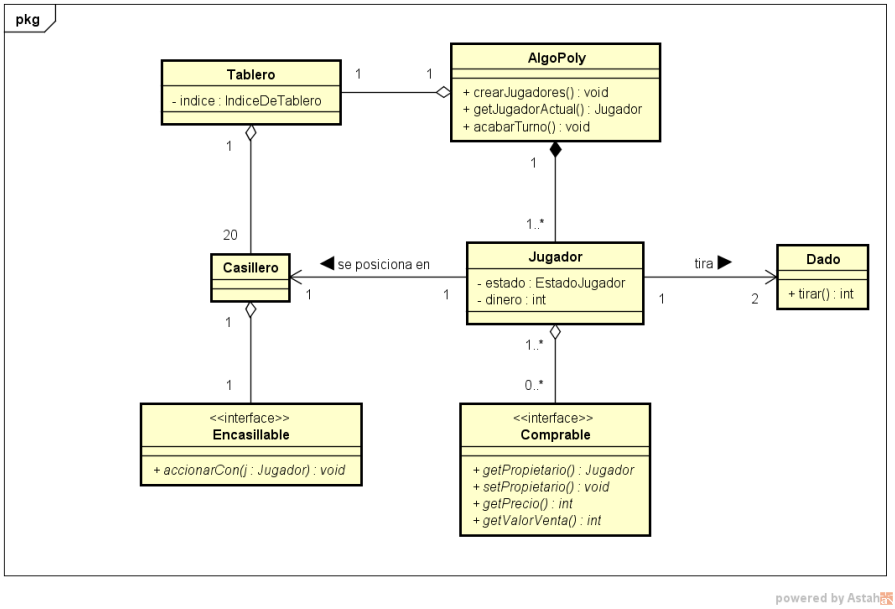

Figura 2: Diagrama general de clases.

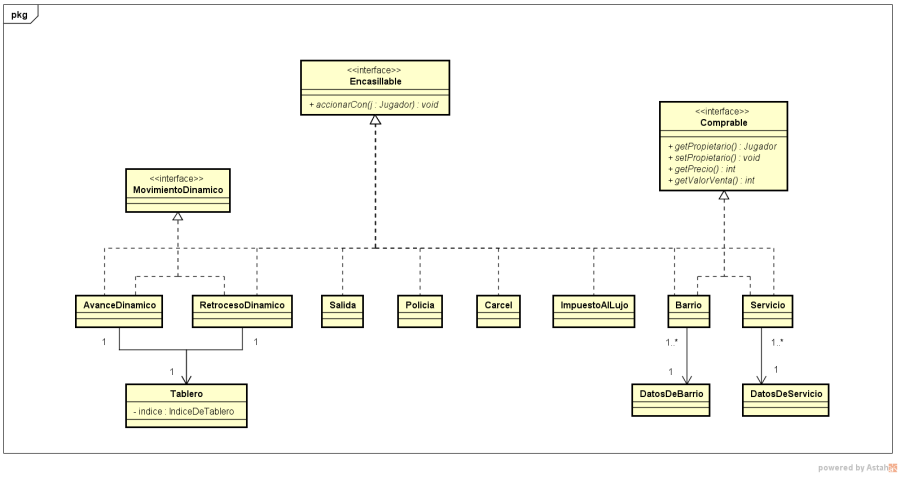

Figura 3: Diagrama de las clases que implementan la interfaz Encasillable.

5. Diagramas de secuencia

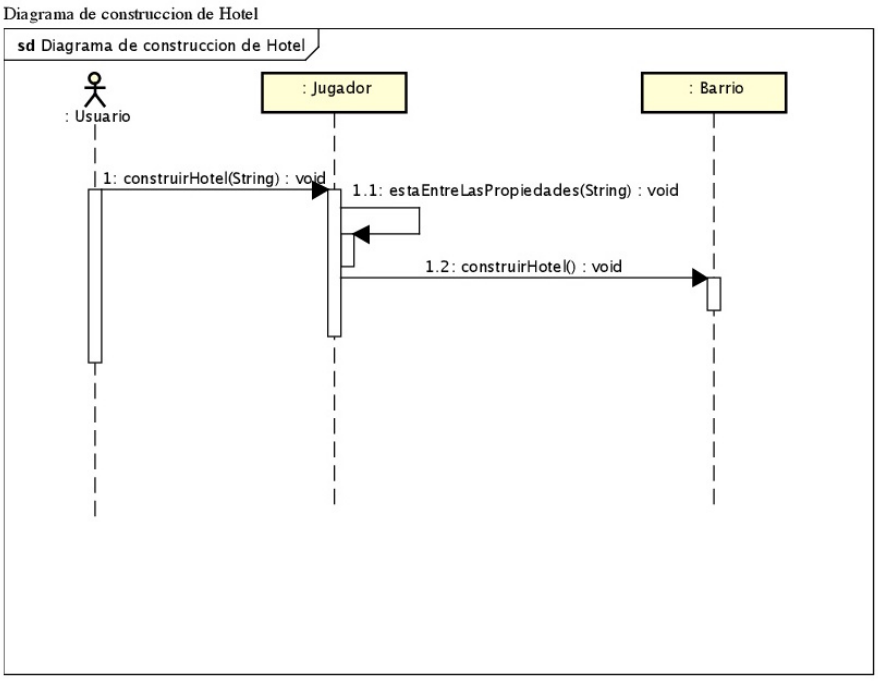

Figura 4: Construcción del hotel

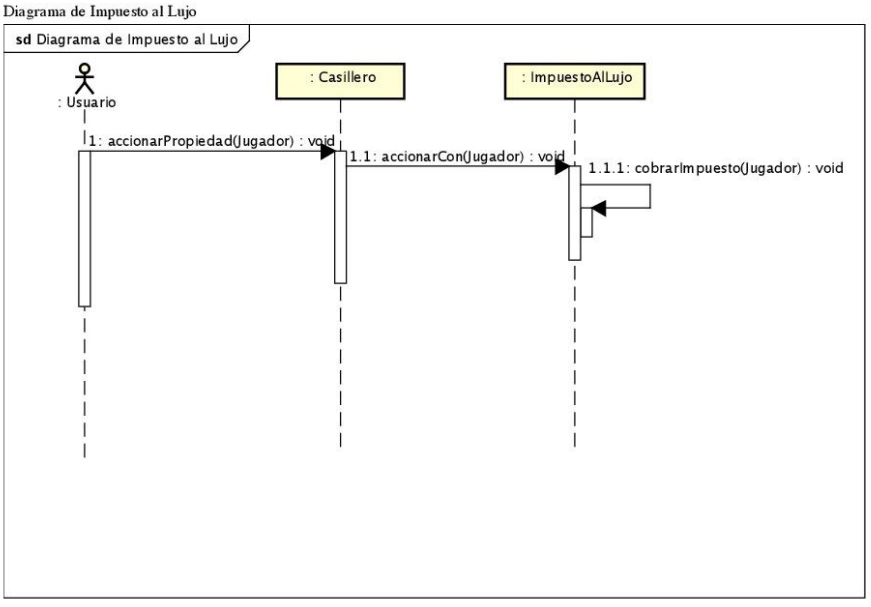

Figura 5: Impuesto al lujo

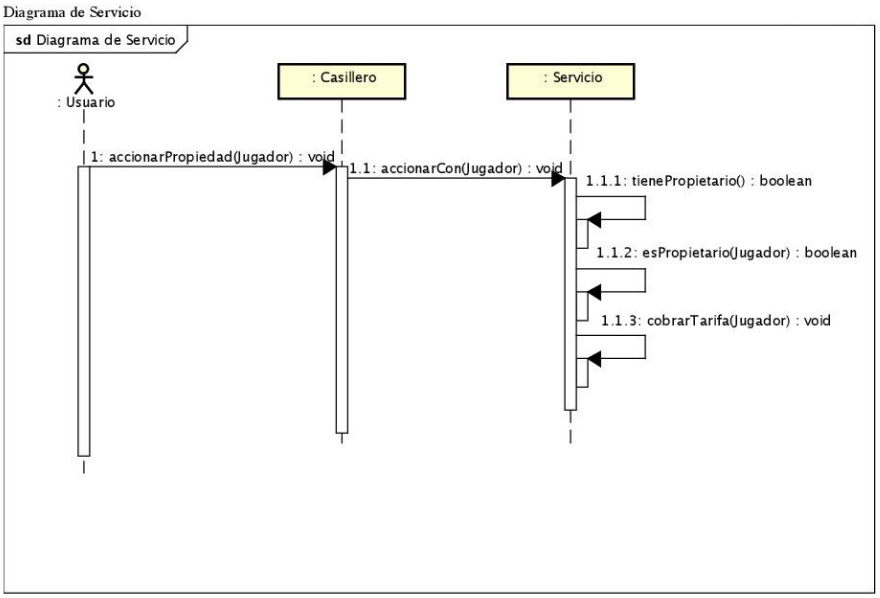

Figura 6: Funcionamiento de Servicio

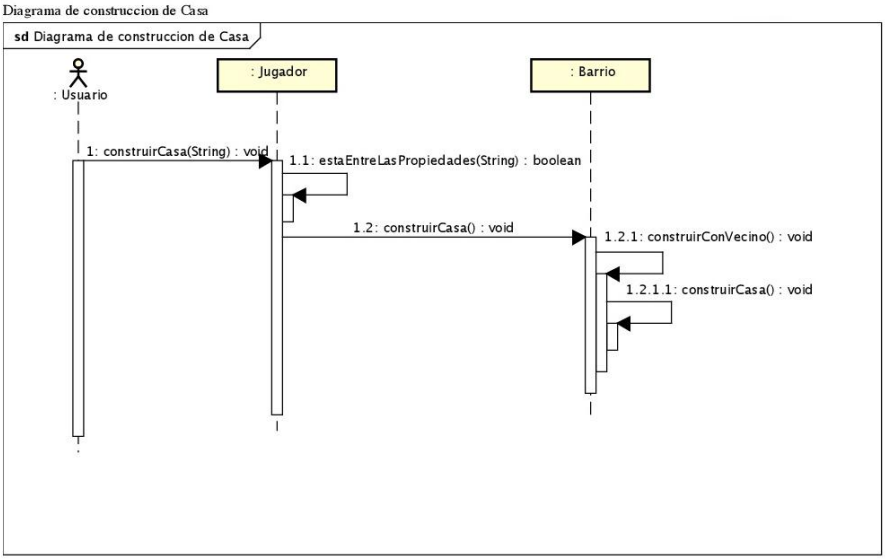

Figura 7: Construcción de casa

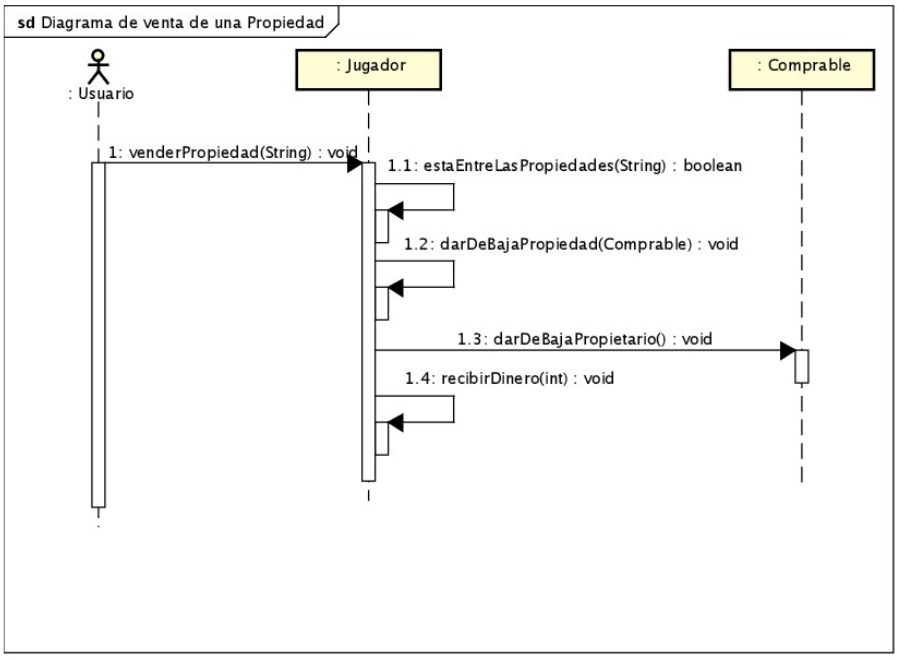

Figura 8: Venta de propiedad

6. Diagrama de paquetes

Ésta es una ilustración de cómo se manejaron los paquetes en el programa.

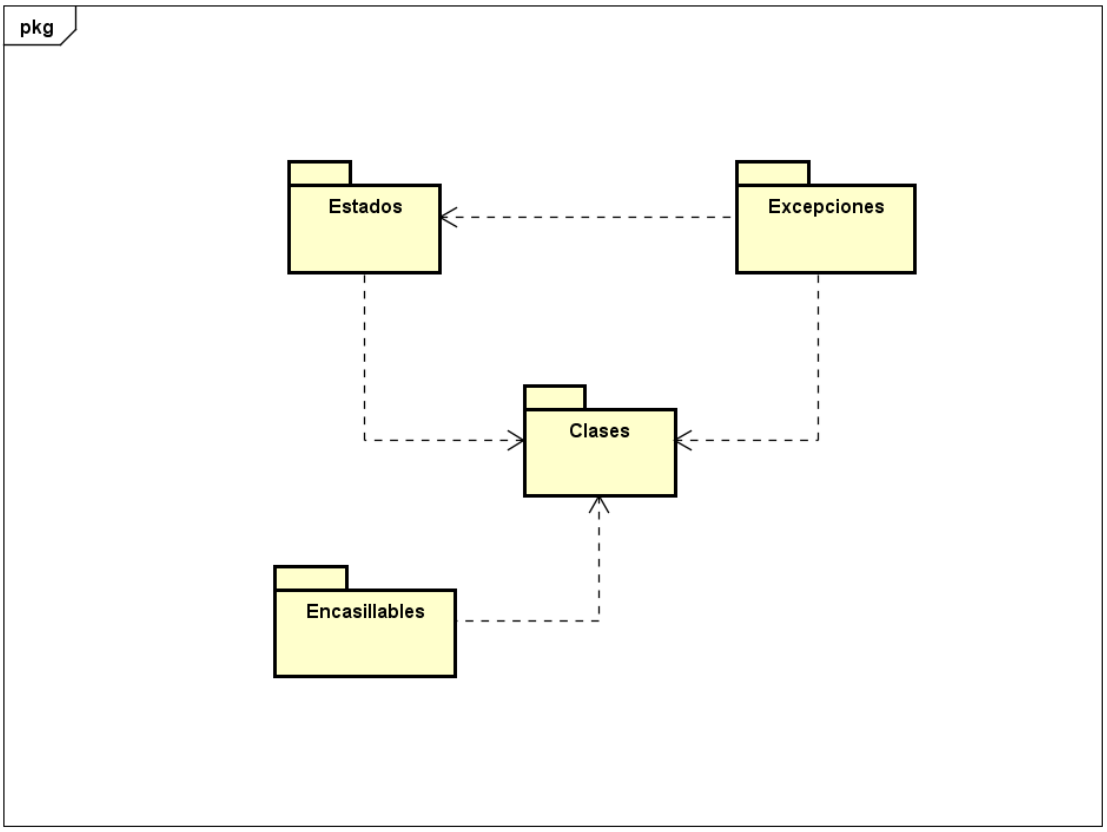

Figura 9: Paquetes

7. Detalles de implementación

7.1. DatosDeBarrio

El barrio fue implementado de manera genérica, es decir que todos los barrios, tengan vecinos, diferentes cantidades de casas máximas o hoteles, son objetos de la misma clase.

Tomando en cuenta lo dicho en la parte de Supuestos, si asumimos que la posición de los casilleros siempre será la misma, entonces los datos se pueden guardar en una estructura estática, para esto se usó la clase DatosDeBarrio, la cual contiene todos los nombres y los datos de los costos, de esta manera cada barrio delega su propia información con gets de esta clase.

```
public class DatosDeBarrio {
    private static String[] NOMBRES = { "Buenos Aires Sur", "Buenos Aires Norte", ...
    private static String[] VECINOS = {"Buenos Aires Norte","Buenos Aires Sur", ...
    private static int [] PRECIO = { 20000, 25000, 18000, ...
    private static int[] PRECIO_CASA = { 5000, 5500, 2000, ...
    private static int[] PRECIO_HOTEL = { 8000, 9000, 3000, ...
    private static int[] MAX_CASAS = { 2, 2, 2, ...
    private static int [] ALQUILER_SIMPLE = { 2000, 2500, 1000, ...
    private static int[] ALQUILER_UNA_CASA = { 3000, 3500, 1500, ...
    private static int[] ALQUILER_DOS_CASAS = { 3500, 4000, 2500, ...
    private static int[] ALQUILER_HOTEL = { 5000, 6000, 3000, ...

    public static DatosDeBarrio getDatosBarrio(Barrios unBarrio) {
        int numBarrio = unBarrio.getNumBarrio();
        DatosDeBarrio datosDeUnBarrio = new DatosDeBarrio();
        datosDeUnBarrio.llenarDatos(numBarrio);
        return datosDeUnBarrio;
        }
```

De manera análoga se implementó de la misma manera con DatosDeServicios.

7.2. Tablero

Tablero, además de contener los casilleros, conoce los índices de cada uno (los casilleros no saben en qué posición están), de esta manera, en el futuro será el que se encargue de manejar las ubicaciones de los jugadores por cada turno.

Nuevamente todo esto se pensó ausmiendo que el tablero conservará siempre los mismos índices, de esta manera siempre se inicializa con una clase Inicializador, que se encarga de agregar los casilleros manualmente al momento de crearse.

8. Excepciones

BarrioNoPuedeConstruirHotelException Ésta excepción se lanza cuando un jugador trata de construir un hotel pero no se puede o no cumple con los requisitos para hacerlo.