FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — AlgoPoly

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

Alumno	Numero De padron	Email
Brandstetter, Martin	97873	martinbran94@gmail.com
Costa, Gonazalo	100243	costagonzalo96@gmail.com
Lizarraga, Augusto	99636	augusjoaliza@hotmail.com

Índice

1. Introducción

El presente informe reune la documentación de la solución del segundo trabajo práctico de la materia Algoritmos y Programación III que consiste en desarrollar una aplicación de un juego símil Monopoly en Java utilizando los conceptos del paradigma de la orientación a objetos vistos en el curso.

2. Supuestos

Decidimos que solo se van a poder agregar casas y hoteles si se está sobre el barrio en cuestión. También, el jugador solo podra comprar y vender un adquirible si se encuentra en la casilla.

3. Modelo de dominio

El diseño del trabajo práctico abarcó un conjunto de clases donde simulamos varios casos en el cual el Jugador cae en diferentes casillas, que llamamos Visitables, los cuales van a ser incluidos en un **Tablero**, que funcionará a través de la clase **AlgoPoly**, que se encargará del desarrollo de la partida. Paso a comentar los visitables:

Quini6 : Clase que representa una casilla que cuando el jugador la visita recibe un determinado dinero. La premisa dice que solo puede recibir dos premios por jugador.

Barrio : Clase que implementa Adquirible que se utiliza para mostrar los distintos terrenos que el Jugador dueño puede agregar casa y hoteles, lo cual sumarán valor a su alquiler. Pueden ser Simple o Doble.

Carcel : Si el jugador cae en ella, se lo retendrá por varios turnos, a menos que pague una fianza.

Policia : Al visitarla, el jugador será enviado a la Carcel.

Retroceso Dinámico y Avance Dinámico : se encargan de que el jugador se mueva dependiendo de diferentes factores, como la tirada de dados, el número de adquiribles que posee y la cantidad de Dinero.

Compania : Clase derivada de Adquirible, que cobrará a Jugadores que no sean propietarios 500 veces la tirada de los dados, 1000 veces si cuenta con el par de Companias.

Además creamos clases adicionales para representar al Jugador y lo que va manejar:

Jugador : representa al jugador de AlgoPoly, estos se crean con un dinero inicial.

EstadoJugador : Es una clase que alterara el comportamiento del Jugador, decidiendo si se mueve o no. Es una Interface, que está implementada por Libre y Encarcelado.

Dinero : representa valores en distintas clases, como en Jugador el efectivo con el que cuenta o en Carcel el valor de la fianza.

4. Diagramas de clase

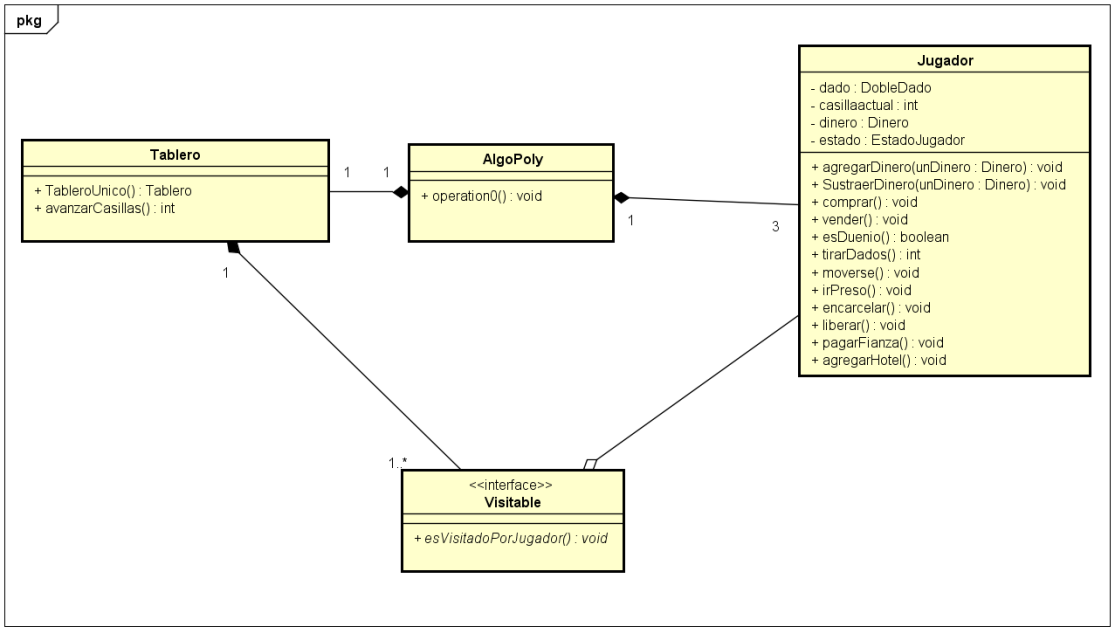

Figura 1: Diagrama general de la aplicación.

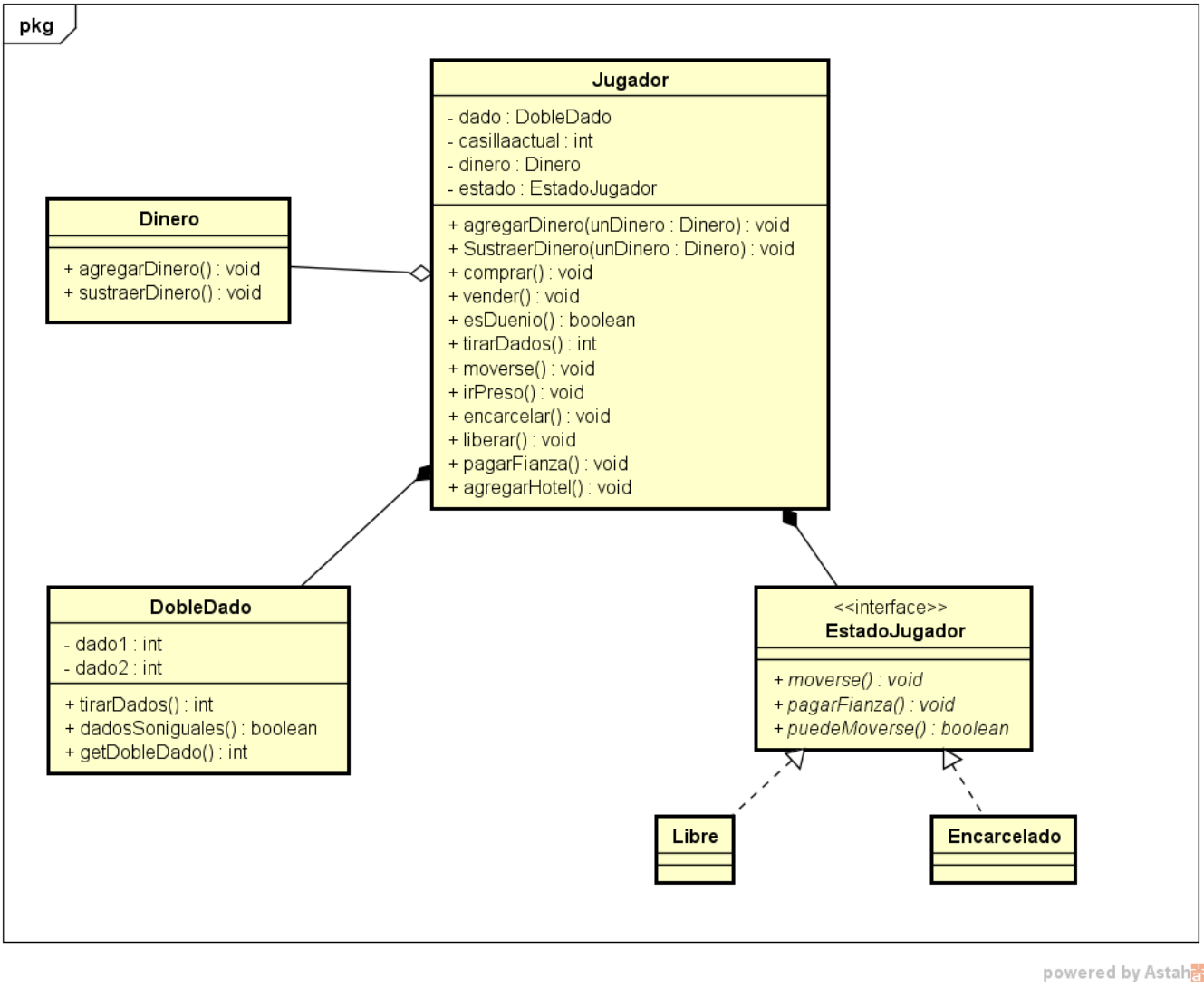

Figura 2: Diagrama de clases de relaciones de Jugador

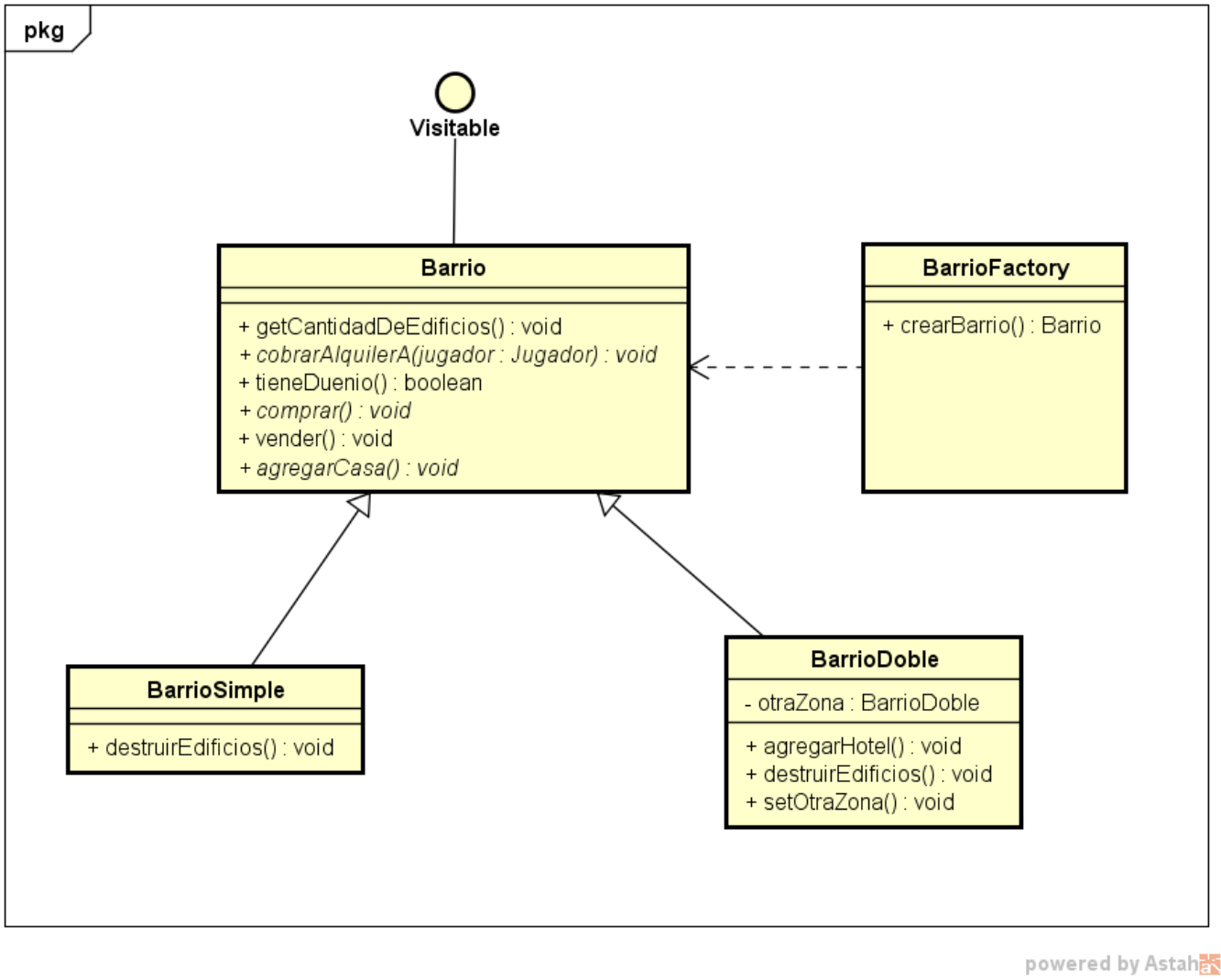

Figura 3: Diagrama De Clases de la estructura de Barrios

5. Detalles de implementación

5.1. Tablero Unico

Para que haya un solo Tablero durante todo el desarrollo de la partida, utilizamos el patron Singleton, para que se cree el Tablero una única vez y se vaya pidiendo esa misma instancia a lo largo del programa.

```
public static Tablero TableroUnico() {
if(tableroUnico == null) tableroUnico = new Tablero();
return tableroUnico;
}
```

5.2. BarrioFactory

Para los diferentes Barrios, se realizó un Factory, en el cual se tienen distintos métodos creacionales de todas las instancias necesarias.

```
public class BarrioFactory {
public Barrio crearTucuman() {
Barrio tucuman = new BarrioSimple(25000, 7000, 2500, 4500, "Tucuman" );
return tucuman;
}

public Barrio crearSantaFe() {
Barrio santafe = new BarrioSimple(15000, 4000, 1500, 3500,"Santa Fe");
return santafe;
}

public Barrio crearNeuquen() {
Barrio neuquen = new BarrioSimple(17000, 4800, 1800, 3800, "Neuquen");
return neuquen;
}
...

}
```

5.3. Patron State

Como se dijo antes, se crearon dos estados que modifican el comportamiento del Jugador, que son Libre, el cual permite que el jugador se mueva libremente por el tablero, y Encarcelado, estado en el que estará en la carcel y no se podrá mover por varios turnos.

6. Excepciones

BarrioLleno Esta excepción es lanzada cuando un barrio tiene todas las propiedades posibles y se quiere agregar una mas.

BarrioNoLleno Esta excepción se lanza si se quiere construir un hotel en un BarrioDoble si su el o su par no tiene todas las casas posibles para construir.

DineroInsuficiente Se lanza si un jugador no puede afrontar un gasto.

JugadorInvalido Se lanza si un Jugador intenta agregar una casa y/o hotel a una casilla.

JugadorNoPuedePagarFianza Es lanzada cuando el jugador no puede pagar fianza entando en la carcel.

HayGanador Excepcion que se lanza al haber un ganador y se quiera avanzar un turno

NoEsDuenioDeLosDosTerrenos Es lanzada al querer agregar una casa en un Barrio Doble sin ser dueño de su par.

TieneDuenio Se lanza si se quiere comprar un adquirible que ya tiene dueño

7. Diagramas de secuencia

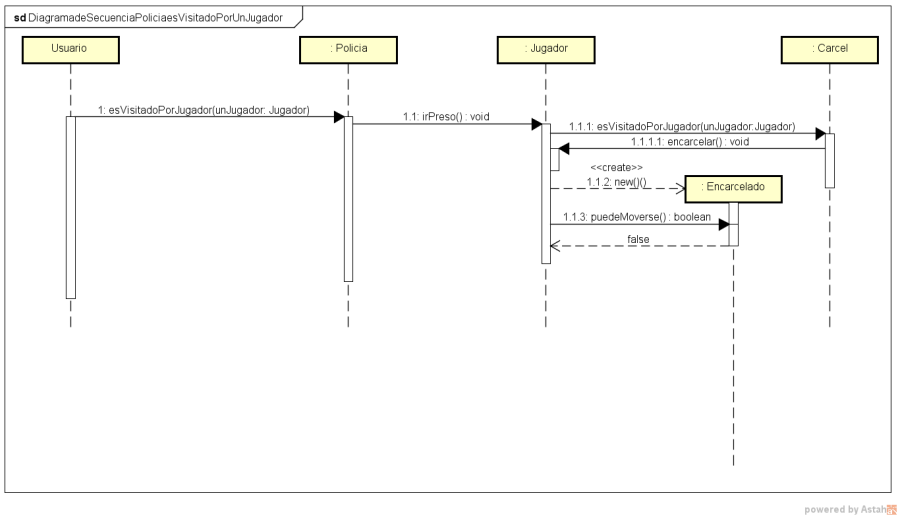

Figura 4: Diagrama de secuencia Policia es visitado por unJugador.

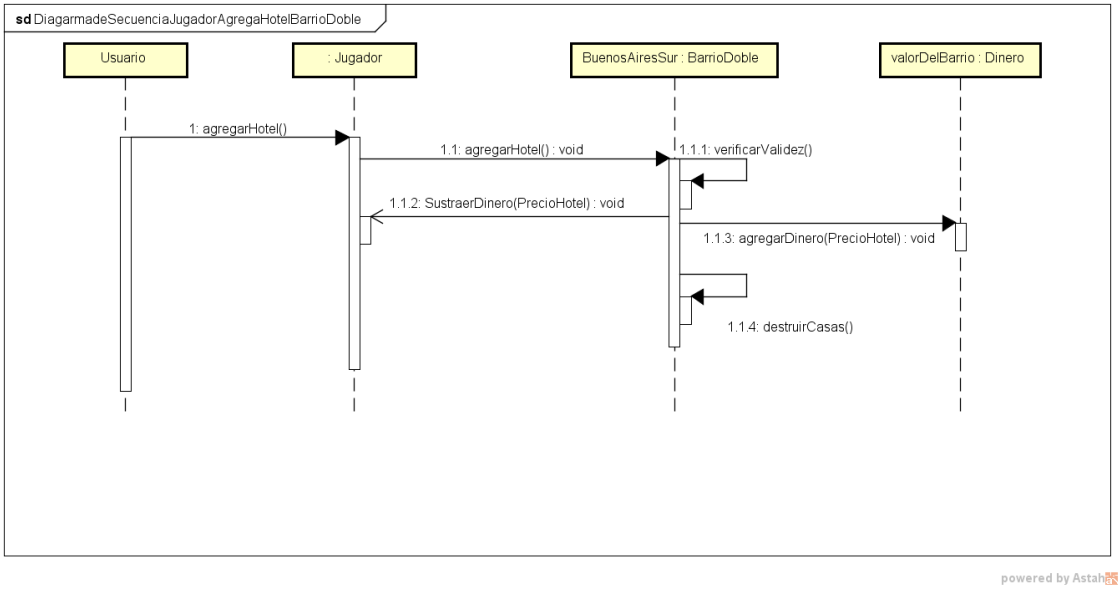

Figura 5: Diagrama de secuencia para jugador que agrega hotel en un barrio doble.

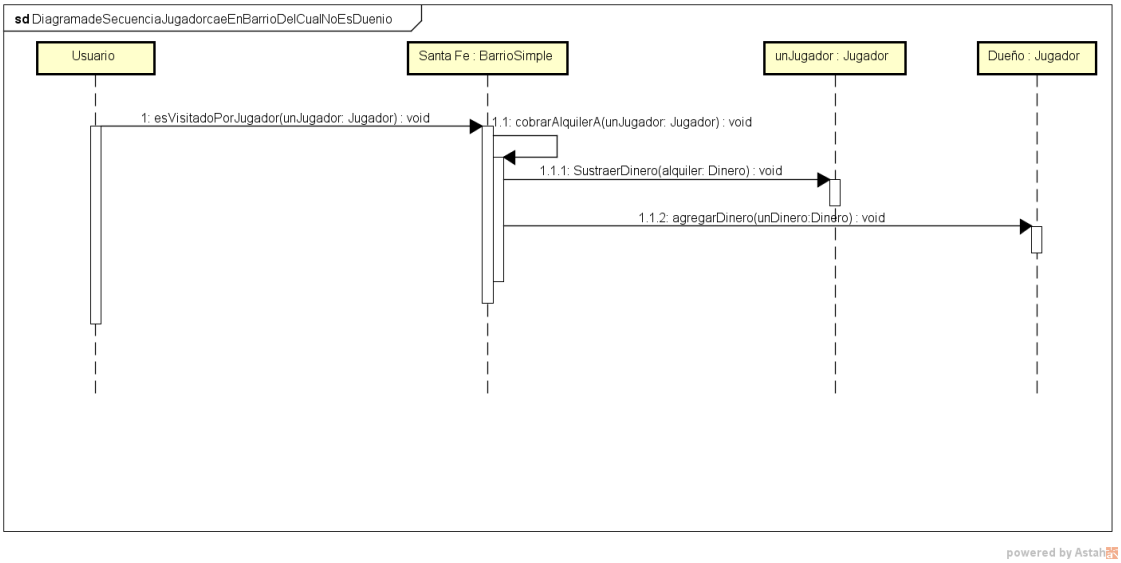

Figura 6: Diagrama de secuencia para un jugador que cae un barrio doble de cual no es dueño.