FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — A.L.T.E.G.O.

[7507/9502] Algoritmos y Programación III
Grupo 12
Primer cuatrimestre de 2021

Alumnos	Padrón	Email
Francisco Insua	100804	finsua@fi.uba.ar
Nahuel Spiguelman	104644	nspiguelman@fi.uba.ar
Facundo Luzzi	105229	fluzzi@fi.uba.ar
Ignacio Argel	104351	iargel@fi.uba.ar
Alejandro Paff	103376	apaff@fi.uba.ar

Índice

1. Introducción **2**

2. Supuestos **2**

3. Diagramas de clases **2**

4. Diagramas de secuencia **5**

5. Diagramas de Estado **6**

6. Diagramas de Paquetes **7**

7. Detalles de implementación **7**

8. Excepciones **7**

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 el juego de Teg en Java, desarrollado utilizando los conceptos del paradigma orientado a objetos.

2. Supuestos

A lo largo del Trabajo Práctico debimos contemplar algunos escenarios del juego que no contaban con una resolución específica desde la consigna. A continuación se enumeran los supuestos adoptados para la aplicación.

- La cantidad de países es fija y se pasa a través de un JSON.
- Los objetivos son fijos y se pasan, también, a través de un JSON con un cierto formato. Estos tienen solo dos tipos:
  - Conquista: Estos refieren a una conquista territorial.
  - Destrucción: Estos refieren a una destrucción a un jugador en particular.
- Los tipos de tarjetas de paises (globo, barco y cañón) se asignan aleatoriamente a los paises.
- Cuando el objetivo es destruir un jugador que ya ha perdido se debe destruir al jugador de la derecha

3. Diagramas de clases

El siguiente diagrama de clase muestra como se compone el juego con las clases que lo conforman.

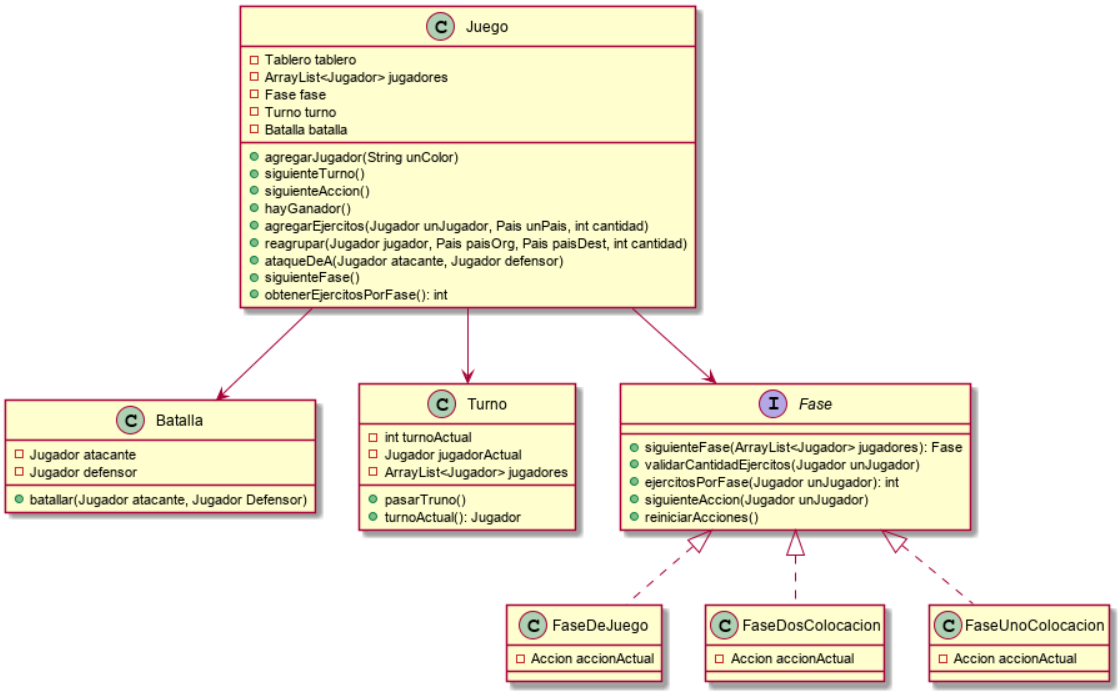

Figura 1: Diagrama de juego.

Este diagrama muestra el tablero, y las clases que lo componen para su correcto funcionamiento.

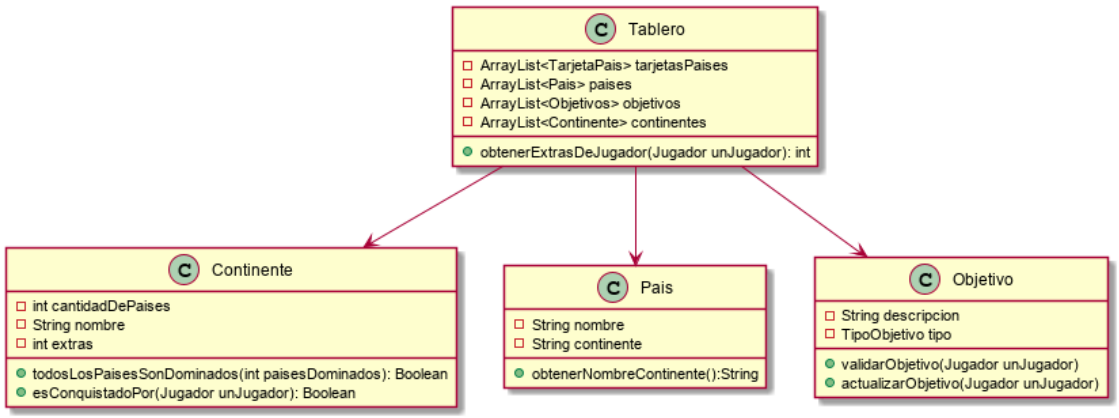

Figura 2: Diagrama tablero.

El siguiente diagrama muestra batalla y su relacion con las clases usadas en la batalla.

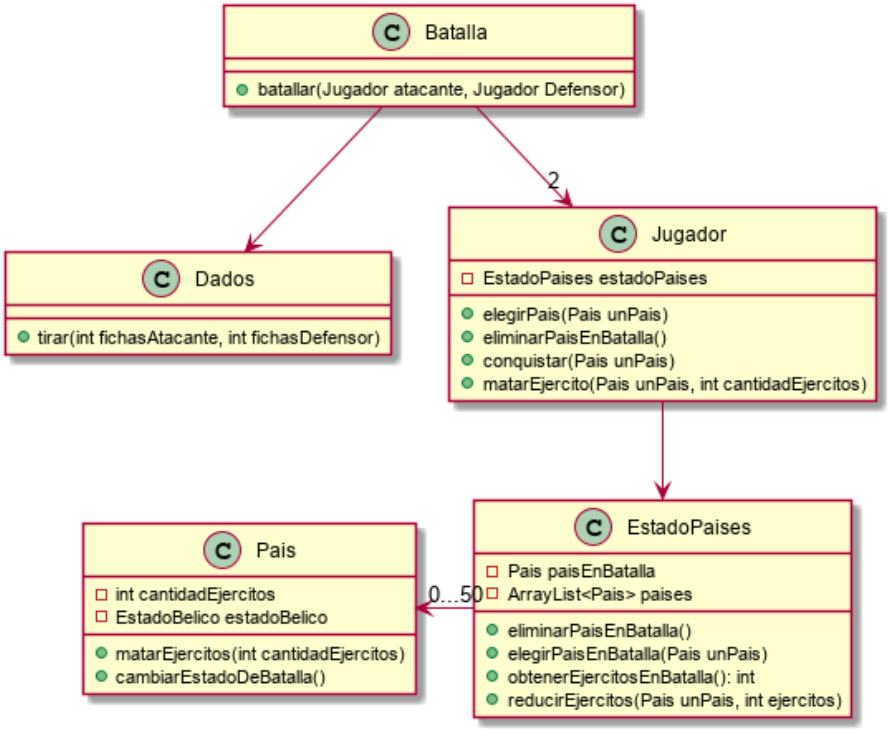

Figura 3: Diagrama batalla.

En este diagrama se muestra objetivo y su implementacion.

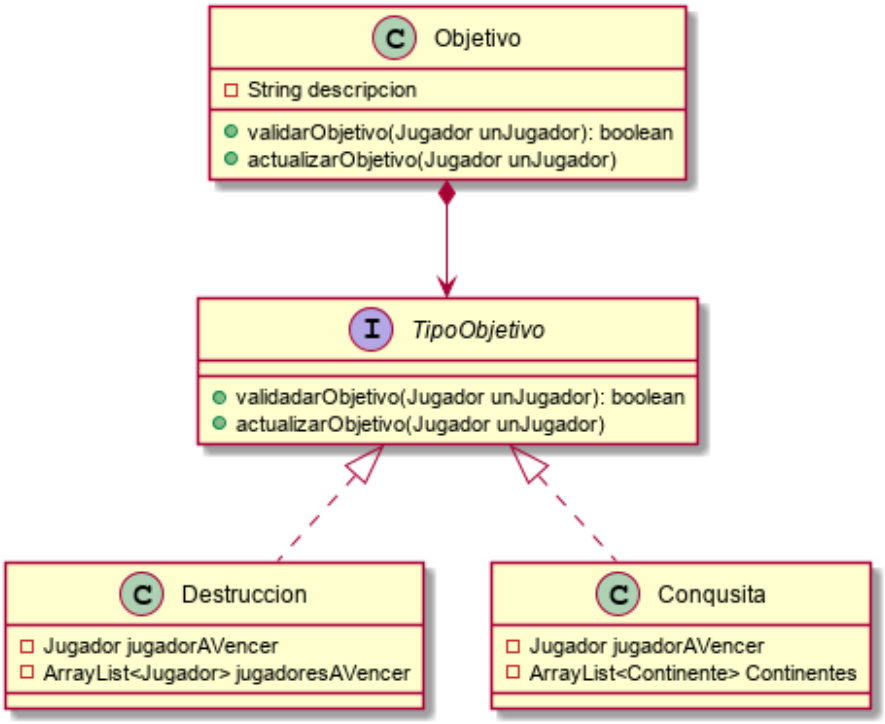

Figura 4: Diagrama objetivos.

El siguiente diagrama muestra pais y el manejo del estado belico.

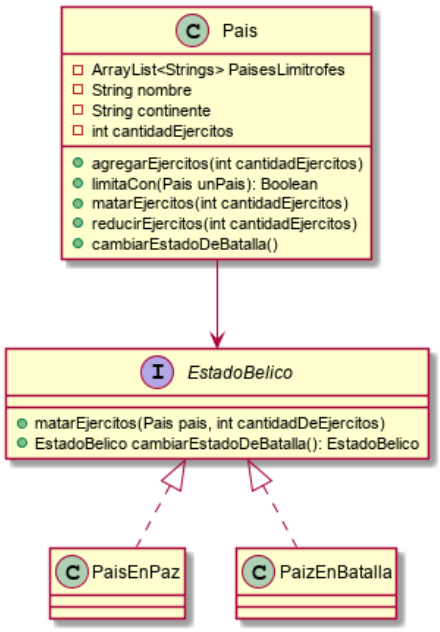

Figura 5: Diagrama principal.

4. Diagramas de secuencia

A continuación mostraremos las secuencias más interesantes de nuestro programa, la mayoría está relacionada con la ejecución de los bloques y de la forma en la que está modelado, los diagramas tienen una estructura similar.

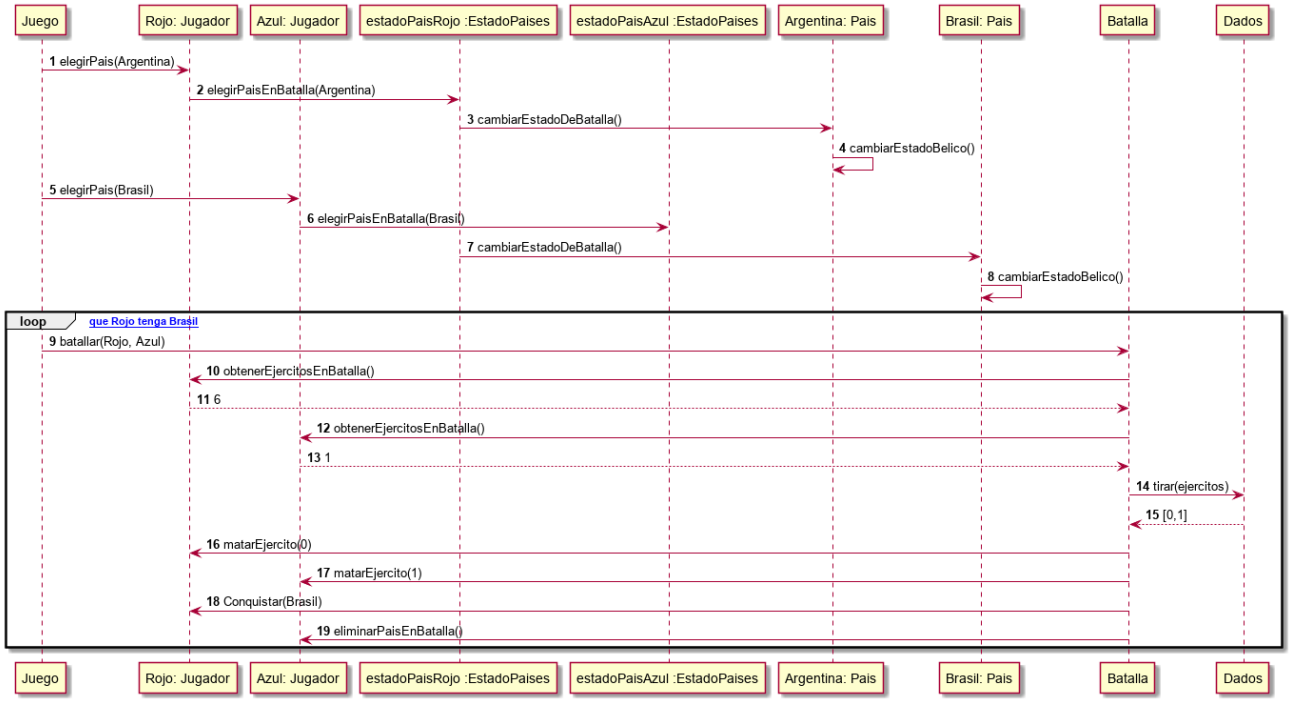

Figura 6: Diagrama batalla.

El siguiente diagrama muestra un caso en el que un jugador conquista un pais y recibe una tajeta pero no es un pais suyo.

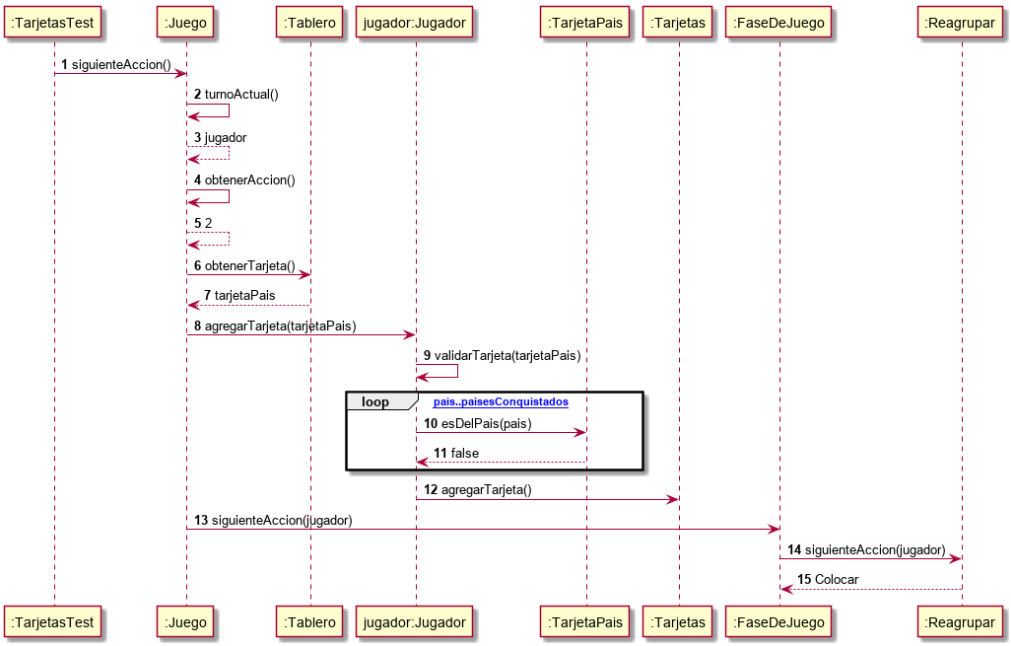

Figura 7: Diagrama canje.

5. Diagramas de Estado

Este diagrama muestra como el juego avanza en cuanto a estados refiere.

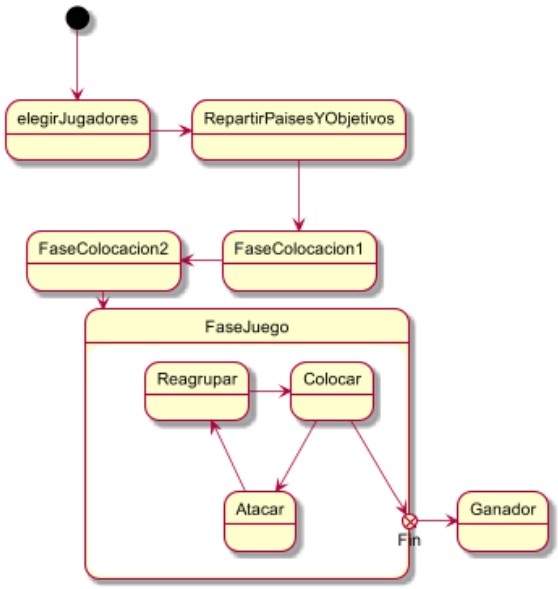

Figura 8: Diagrama de estado del juego.

6. Diagramas de Paquetes

En esta seccion, detallamos por medio de la imagen a continuación la organización general de los paquetes del programa.

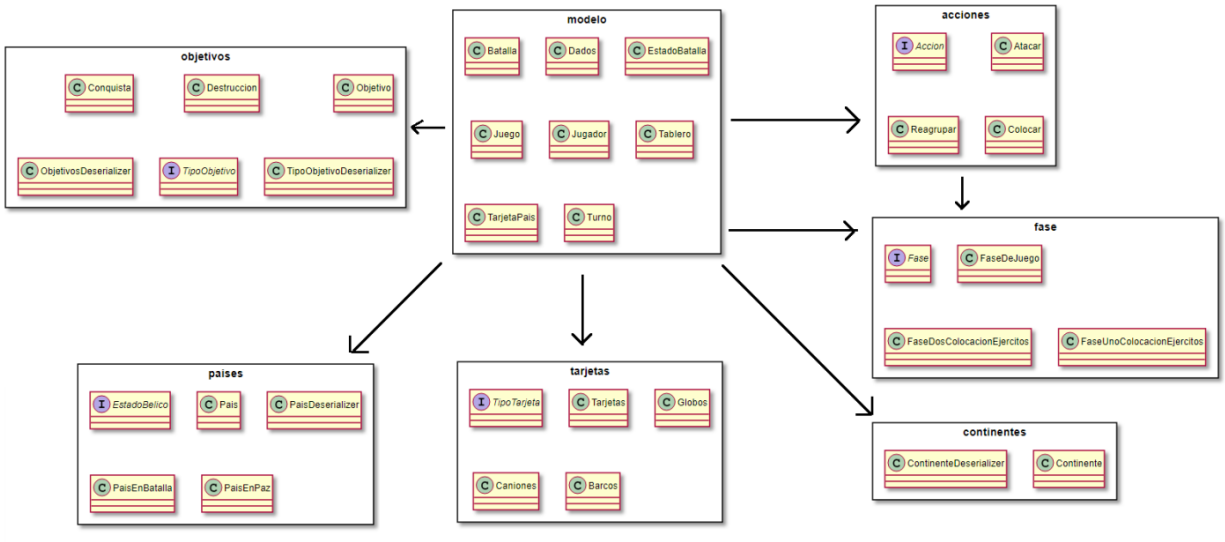

Figura 9: Diagrama de estado del juego.

7. Detalles de implementación

En la implementacion se utilizo State Pattern para:

- Juego: Al pasar de fase, esta cambia en funcion de la inmediata anterior
- Objetivo: Cada jugador tiene un objetivo, el cual depende de su tipo que es una variable de instancia la cual maneja el comportamiento del mismo.
- Pais: El pais maneja su estado belico, a traves de una variable de instancia que determina el comportamiento del mismo en diferentes etapas del juego.

8. Excepciones

A continuación detallamos las excepciones que implementamos a lo largo del modelo (todas heredan de TegException):

- AccionesException

Se lanzará cuando no se siga el orden de la fase de cada juego, es decir, atacar, reagrupar, colocar ejercitos.

- CantidadDeEjercitosInvalida
- CantidadEjercitosInsuficientesException

Se invoca cuando no hay suficientes ejercitos para realizar una reagrupación.

- ColocarEjercitosException

Si la cantidad de ejercitos a colocar no se corresponde con la cantidad que un jugador puso. Por ejemplo en la fase de colocación cuando un jugador no ha terminado de poner sus todas sus fichas.

- JugadorExistenteException

Al agregar un jugador en caso que el color ya haya sido elegido levantará la excepción

- PaisInvalidoException

Se lanza cuando un pais no está en los que fueron cargados en el Json.

- PaisNoEstaEnBatallaException

Cuando no se está en la fase de ataque entre paises y se intenta descontar ejercitos de un pais.

- PaisNoPerteneceAJugadorException

Si un jugador intenta color ejercitos en un pais que no es suyo.

- SiguienteFaseException

En caso que un jugador no tenga la cantidad de ejercitos suficientes para pasar a la siguiente fase, se lanza la excepcion indicando la cantidad necesaria para hacerlo.

- TurnoException

Se invoca en caso que un jugador intente hacer un movimiento y no sea su turno actualmente.