FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — AlgoChess

[7507/9502] Algoritmos y Programación III
Curso 1
Segundo cuatrimestre de 2019

Alumno	Padrón	Email
ALVAREZ, Ernesto	102221	ernesto.alvarez1812@gmail.com
CALIZ BLANCO, Alejo	102842	alejocalizblanco@gmail.com
LYNCH, Matias Nicolas	102571	mlynch@fi.uba.ar
VELA, Daniela Ailin	97613	dani_ailin14@outlook.com

Índice

1. Introducción

El presente informe reúne la documentación de la solución del tercer trabajo práctico de la materia Algoritmos y Programación III, que consiste en desarrollar la aplicación de un juego de estrategia por turnos para dos equipos, se cuenta con una serie de unidades a colocar en un tablero, teniendo distintas propiedades de jugabilidad cada uno. La aplicación ha sido desarrollada en el lenguaje Java utilizando los conceptos del paradigma de la orientación a objetos vistos hasta ahora en el curso.

2. Supuestos

2.1. Turno

En un turno un jugador puede pasar su turno, atacar o mover una unidad a un casillero en una distancia 1(uno). Las diagonales adyacentes a cada celda se consideran a una distancia 2(dos) por lo que no es posible mover una unidad a una celda vacía diagonal en un único turno.

2.2. Batallón

Se considerará que solo dos disposiciones de soldados sobre el tablero pueden conformar un batallón: 3 soldados en línea horizontal o 3 soldados en línea vertical. No es válida ninguna de las otras disposiciones posibles. El batallón no se crea automáticamente, la decisión de crearlo será del jugador.

2.3. Ataque y curación

Se supondrá que el jugador conoce las reglas: al curar una unidad que no puede ser curada, o atacar una unidad que no debe ser atacada, la acción contará como un turno.La lógica del juego es respetada más allá de la acción antirreglamentaria del jugador.

3. Modelo de dominio

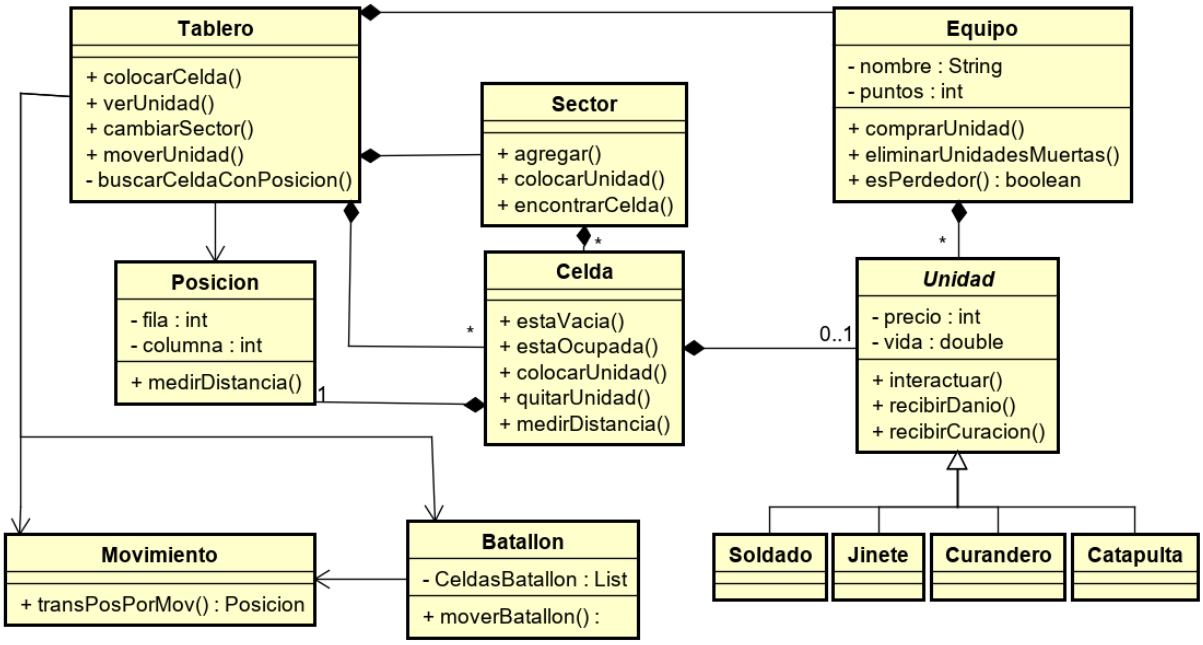
Modelo general : El modelo de dominio del trabajo se basa en que uno utiliza el objeto tablero del programa para realizar todo lo que uno quiere hacer, ya sea colocar una unidad o realizar un ataque. El tablero desde que uno le da la orden se encarga de delegar trabajo a las demás clases y a su vez coordinarlas a todas para realizar un objetivo común. Por ejemplo para mover una unidad, dadas dos posiciones, el tablero busca en sus celdas la celda destino y origen comparando posiciones. Luego si todas las condiciones para el movimiento son correctas, el mismo saca la unidad de una celda, y la pone en la otra.

Tablero : El juego se desarrolla sobre un tablero de dimensiones 20x20, el mismo tiene dos sectores: el superior y el inferior, cada uno le corresponderá a uno de los dos equipos.

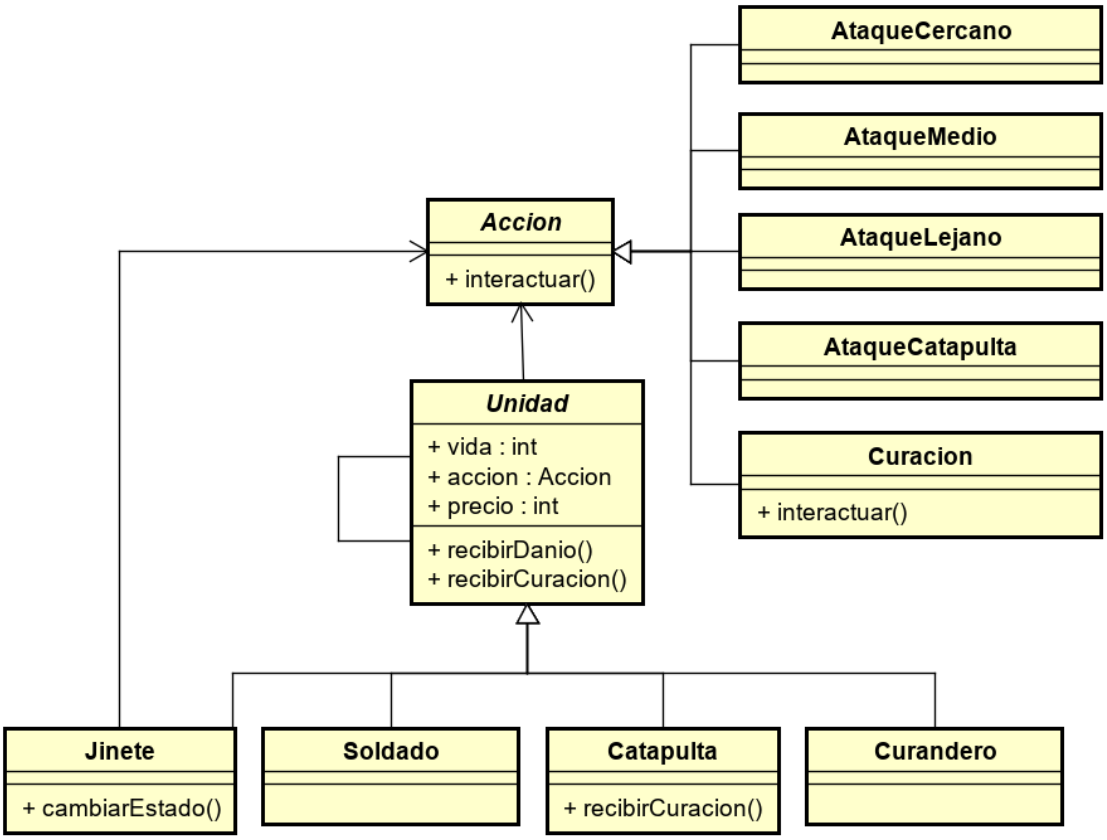
Unidades : Jinete: ataque cercano y ataque lejano. Soldado: ataque cercano. Catapulta: ataque lejano. Curador: Cura unidades en distancia cercana.

4. Diagramas de clase

4.1. Diagrama de clase principal



4.2. Diagrama de clase detallado de unidad



5. Detalles de implementación

5.1. Patrón de diseño: Factory

Este patrón facilita la creación de objetos instanciados de subclases hijas que tienen en común la clase padre, es decir, todas son del mismo tipo. En este trabajo se deben crear una serie de unidades diferentes elegidos por el usuario. Por lo tanto se creó una FabricaUnidad, que es instanciada y utilizada para crear una unidad en específico elegida por el usuario en tiempo de ejecución. Si más adelante se quisiera agregar una nueva unidad simplementa habría que agregar la clase hija al package y el constructor a la FabricaUnidad.

6. Excepciones

CeldaNoEstaEnElTablero

CeldaNoEstaEnMiSector

CeldaNoTieneUnidad Excepción que se lanza cuando el usuario quiere mover una unidad de una celda que no contiene ninguna.

CeldaYaTieneUnidad Excepción que se lanza cuando el usuario quiere mover posicionar una unidad en una celda que ya está siendo ocupada por otra.

EquipoNoPuedeComprarMasUnidades Excepción que se lanza cuando el equipo ya no posee puntos para comprar más unidades.

EquipoNoTienePuntosSuficientes Excepción que se lanza cuando el equipo quiere comprar una unidad que sale más puntos de lo que tiene, teniendo aún puntos disponibles para gastar.

EquipoQuiereCrearUnidadInvalida Excepción que se lanza cuando quiere crear una unidad inválida, es decir, una unidad que no existe o no se encuentra en la factory.

MovimientoInvalido Excepción que se lanza cuando el usuario quiere mover una unidad que se puede mover, una distancia mayor a 1(uno), o si quiere mover una unidad que no puede moverse.

NoEsBatallon

NoSeEncontroLaCelda Excepción que se lanza cuando el tablero busca entre sus celdas una que no le pertenece.

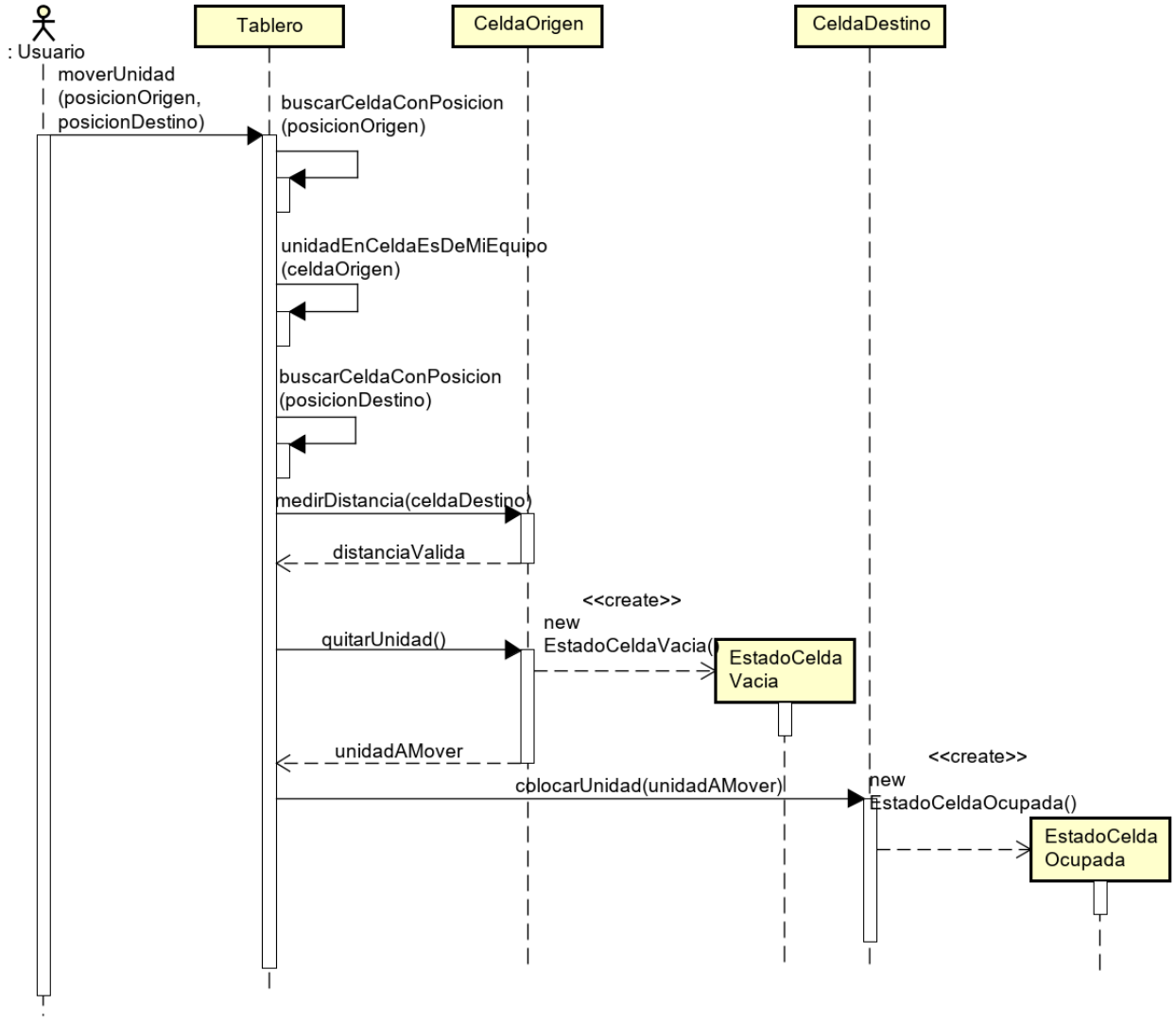
RangoMuyCercano

RangoMuyLejano

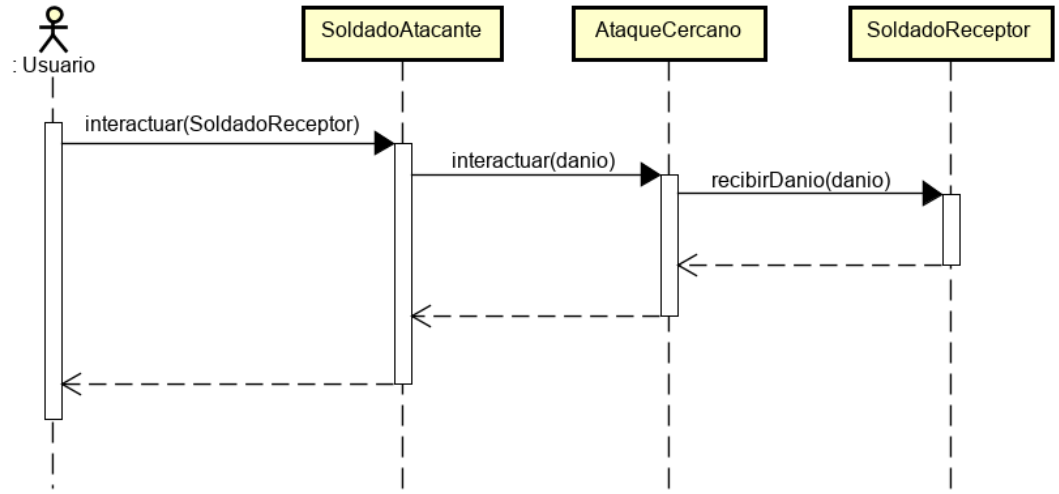
TableroSectorInvalido Excepción que se lanza en el tiempo de preparación del juego cuando un jugador quiere posicionar una unidad comprada en el sector enemigo.

7. Diagramas de secuencia

7.1. Mover una unidad de una celda a otra



7.2. Atacar con soldado a soldado enemigo



7.3. Atacar con jinete a unidad enemiga cercana