Las partes esenciales de mi modelo son TEGEngine y ALTEGO. TEGEngine es el ‘engine’ del juego, digamos, no sabe nada sobre interacción del usuario, sino que recibe mensajes de juego como ‘colocar ejércitos’ o ‘terminar fase’. Luego ALTEGO es una clase que utiliza el engine de acuerdo a eventos de UI, es decir es el único que sabe de JavaFX y los eventos de JavaFX. Así en las primeras entregas podemos concentrarnos en desarrollar TEGEngine y nos hacemos la vida más fácil cuando llegue hora de pensar en el UI.

TEG está compuesto de 2 fases:

- fase de inicio
- fase de juego

Y la fase de juego es un ciclo de 3 etapas:

- atacar
- reagrupar
- colocar ejércitos

Donde cada iteración es un turno de un jugador. Opino que TEGEngine debería ser quien define las reglas del juego. Durante la fase de inicio TEGEngine no acepta mensajes como atacar(), durante la fase de juego no acepta repartirPaises(). Además no se puede agregar más jugadores después de repartir los paises, en ese sentido existen ‘etapas’ dentro de la fase de inicio. Se me ocurre esto:

TEGEngine tiene estrategias como ‘faseDeInicio’ y ‘faseDeJuego’, A lo mejor a su vez fase de inicio tiene estrategias como ‘etapaDefinirJugadores’, ‘etapaColocarEjercitos’ y la fase de luego tiene estrategias como ‘etapaAtacar’. Si el usuario (ALTEGO) trata de hacer algo en la etapa incorrecta, TEGEngine lanza una excepción, es decir que es responsabilidad de ALTEGO no hacer ese tipo de pedidos inválidos.

Lo otro es que TEGEngine no lance error cuando se usa la etapa equivocada, sino que devuelva una notificación de algun tipo, pero no creo que eso sea correcto

Para cosas como el turno hay que esperar que el usuario termine explícitamente las etapas (terminarTurno, terminarEtapaAtacar) pero no sé si seria así para todo.

Modelo

Juego/TEGEngine: el engine que les mencioné, sabe de los jugadores, el mapa y el orden de turno. A lo mejor el orden de turno se le puede delegar a un objeto OrdenDeTurno:

```
ordenDeTurno.definirJugadores(jugadores)
ordenDeTurno.siguienteJugador()
ordenDeTurno.jugadorALaDerechaDe(jugador)
```

Jugador: tiene un color, unos soldados, países conquistados y las cartas de país.

Mapa: tiene un grafo que define todos sus países como nodos, y si son adyacentes estan conectados por una arista

Pais: Responsable de atacar a otros países. tiene un jugador que lo controla, y una cantidad de soldados. cuando hay un ataque el pais NO recibe soldados que no sean de su jugador. Cuando es conquistado por otro jugador B, B tiene que poner al menos uno de sus soldados en el país.

Soldado: no sé si hace falta definir una clase soldado, porque no son únicos y cada país sólo posee sus propios soldados en un momento determinado. O sea cada país puede tener un int ‘cantidadSoldados’ y sirve.

Notas

Correcciones de errorcitos que venían con el proyecto base

https://stackoverflow.com/questions/56063566/maven-how-to-remove-module-info-class-warning-for-shaded-jara

https://web.archive.org/web/20210528104923/https://openjfx.io/openjfx-docs/