Universidad de Costa Rica
Escuela de Ciencias de la Computación e Informática

CI-0136: Diseño de Software

Proyecto Final Etapa III: Justificación de Diseño

Integrantes:
Hansell Solís Ramírez, C07677
Henry Ledezma, C04185
Josué Retana Rodríguez, C06440
Judith Vargas Esquivel, C08168

Profesor:
Allan Calderon Castro

Ciudad Universitaria Rodrigo Facio
II Ciclo-2022

El diseño creado en el presente proyecto, “Senet”, se efectuó con el objetivo de fomentar la mayor flexibilidad, independencia y oportunidad de actualización e integración de futuros juegos. Por ello, se siguieron los principios provistos por SOLID. Por otro lado, se presenta la justificación de las clase, las cuales son:

- ConstructorAbstractoReglas: se creó con la empresa de contar con un expresión de conceptos generales referente a las reglas, esto ayuda a poder ingresar nuevas reglas independientes entre sí; pero que comparten, obligatoriamente, una estructura. Además, esto permite poseer un código más reutilizable, al respetar uno de los principios SOLID, “Principio de Abierto/Cerrado”.

- ReglaProteccion, ReglaBarrera y ReglaCasillaEspecial: clases concretas que heredan de “ConstructorAbstractoReglas”, con ellas se logra detectar si existe una protección, una barrera o si el jugador está en un casilla especial. Contar con estas múltiples clases separadas, permite modificarlas o hasta eliminarlas en el futuro, de forma independiente a las demás. Cabe destacar que, con esto se respeta dos de los principios SOLID, “Principio de responsabilidad única” y “Principio de Inversión de Dependencias”.

- ConstructorSerializadorAbstracto: expresión de conceptos generales referente a la construcción y lectura de un archivo serializado que cuenta con el estado del juego. Con esto se pueden agregar nuevas formas de crear y leer archivos de juegos, dependiendo de las necesidades de este, como se puede observar esta característica fortifica la creación de un código flexible y adaptable a otros juegos.

- ConstructorSerializadorCSV: clase concreta que hereda del “ConstructorSerializadorAbstract”, que permite crear un archivo CSV para guardar y leer archivos del estado del juego, esta clase se puede modificar sin afectar el estado del juego.

- Casilla: esta clase forma parte del tablero, por lo que tenerla separada al resto del código fomenta la flexibilidad de modificación futura, al separar las zonas del código que pueden ser clases independientes. Al igual que las pasadas, esto ayuda a poseer un programa con la mayor reutilización posible, pues está clase se puede adaptar para contar con un tablero, completamente, diferente.

- Árbitro: Esta clase tiene reglas del juego, y se asegura de que estas se cumplan cuando el mediador hace los movimientos. Verifica el turno, si una ficha salió del tablero y si algún jugador ya ganó. De igual forma lleva la cuenta de cuántas fichas le quedan a cada jugador

- Jugador: Esta clase abstracta permite poder agregar funcionalidades de un jugador al juego. En el caso de Senet, no es verdaderamente necesario pero para otros juegos de mayor complejidad podría llegar a serlo. Los atributos de esta clase son el id del jugador, el nombre, la puntuación que este posee y el turno, es decir, si le toca jugar o no.