Proyecto global diseño de software

Julio Ballesteros

Pablo García

Descripción del juego

El juego desarrollado es un juego de combates jugado por consola en el que el jugador elige una clase de personaje de entre tres disponible (guerrero, mago y druida) y lucha contra enemigos en combates. Cada personaje, jugador o enemigo, cuenta con unas estadísticas diferentes que dependen de la clase de personaje. Cada combate se juega por turnos, se elige aleatoriamente el combatiente que ataca primero y desde entonces se turnan hasta que uno de los dos se queda sin vida. El jugador acumula puntos por cada combate ganado hasta que se queda sin salud, en ese momento el juego termina.

En cada combate se combatirá contra un único enemigo decidido aleatoriamente de entre tres tipos disponibles: demonio, no-muerto y elemental. Cada combate se desarrollará en un mundo aleatoriamente dentro de tres mundos posibles: Azeroth, Terrallende y Tierras del fuego, en función del mundo donde se combata los enemigos dispondrán de unas habilidades u otras.

Tanto el jugador como el enemigo disponen de tres habilidades que pueden usar en el combate con distinto impacto, en el turno del jugador se lo preguntará por la acción que quiere realizar. Los personajes cambiarán de estado según los acontecimientos del combate, algunas habilidades pueden hacer cambiar de estado a los personajes.

Los estados en los que puede estar el personaje son: activo, aturdido, quemado y sangrando. En el estado activo los personajes no sufren ningún efecto. En el estado quemado los personajes sufren daño en cada turno. En el estado aturdido los personajes no pueden atacar. En el estado sangrando los personajes sufren daño en cada turno. Los estados con efectos adversos duran dos turnos, después de los cuales se reestablece el estado activo.

La acción que realiza el enemigo depende de la estrategia que tenga, las estrategias posibles son: defensiva, ofensiva y equilibrada, cada una da más prioridad a los ataques ofensivos que los defensivos en ese orden.

Manual de uso

Al inicio del juego se preguntará que clase de personaje queremos escoger, se debe introducir en la consola un número entero del 1 al 3.

Una vez escogido el personaje con el que se jugará comienzan los combates, la consola mostrará el próximo enemigo a batir. En el turno del jugador se preguntará que habilidad se desea realizar, se debe introducir en la consola un número entero del 1 al 3.

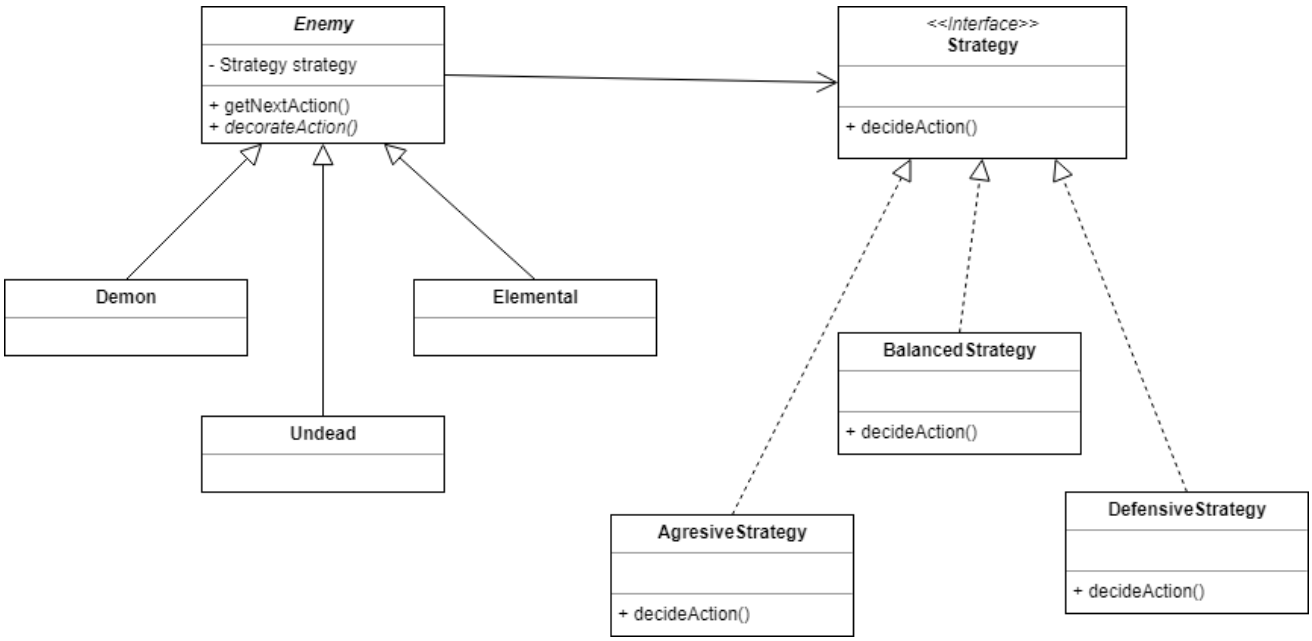
En cada acción de personaje se muestra por consola los resultados de esa habilidad, indicando el daño infligido al oponente y si ha habido algún cambio de estado. También se muestra en cada turno daños por estados del personaje.

Cuando se ha logrado reducir la vida del enemigo a 0 termina el combate. Y se procede a iniciar otro combate contra un enemigo diferente.

Cuando la salud del jugador se reduce a 0 el juego termina y se muestra por pantalla los puntos obtenidos.

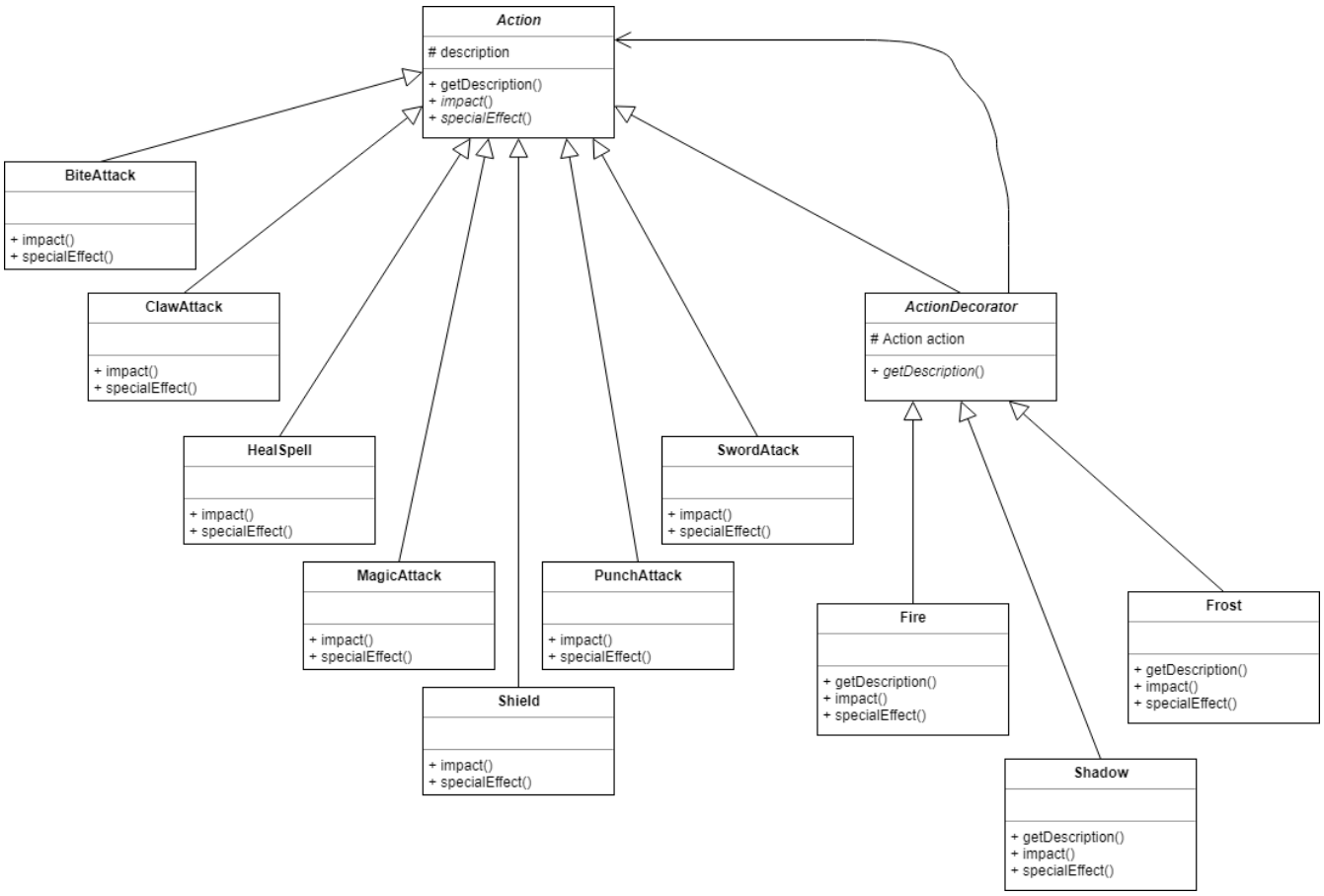
Patrones

Patrón Strategy:



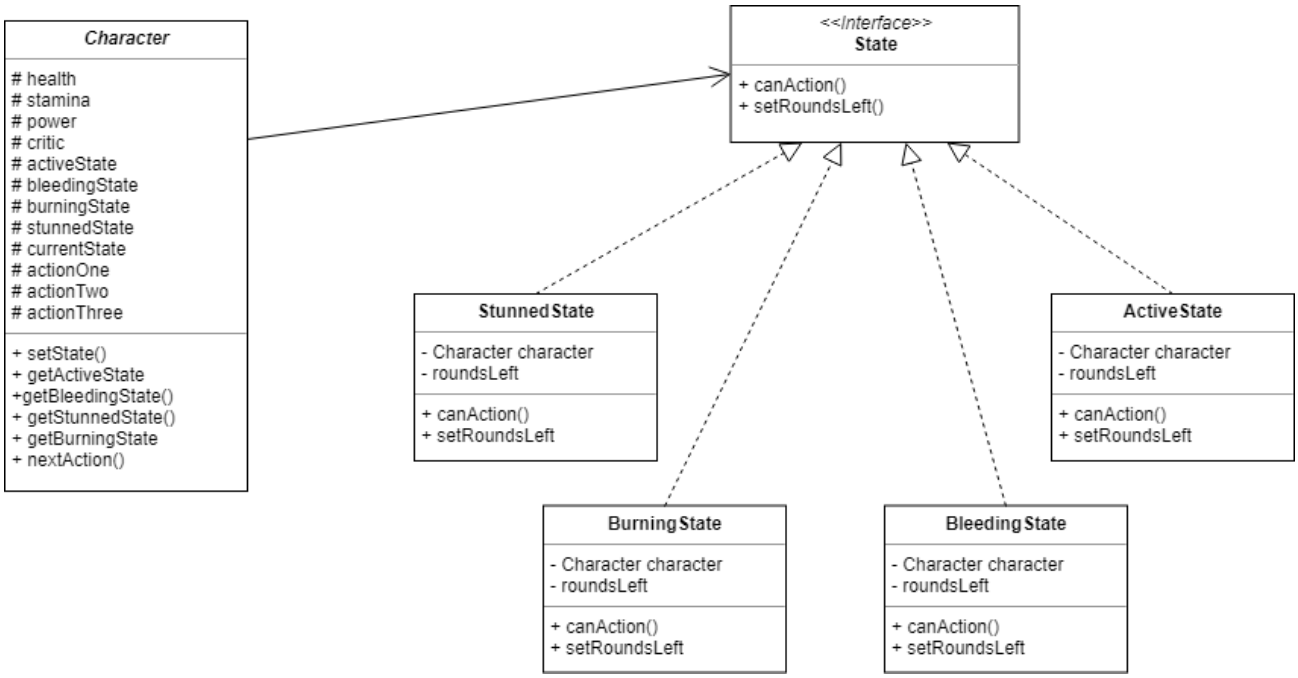
Se ha empleado un patrón strategy para gestionar las estrategias que adopta el enemigo. Cuando se crea un enemigo, en el constructor se le asigna al azar una estrategia, la estrategia se encarga de decidir que acción será la siguiente que utilizará el enemigo. La clase enemigo llama a la estrategia en el método getNextAction(), invocando el único método de las estrategias, decideAction(), según la estrategia que sea se devolverá una acción u otra.

Patrón Decorator:



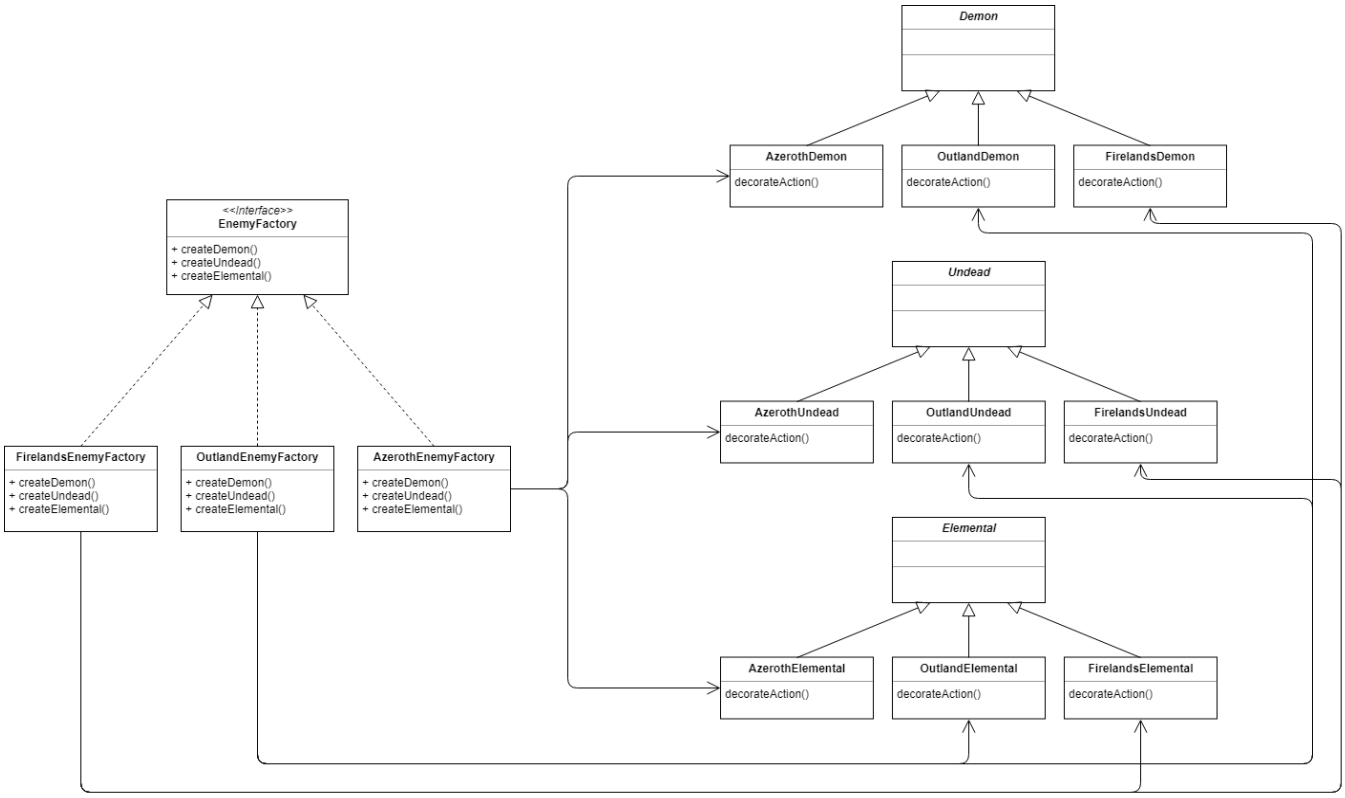
Se ha empleado un patrón decorator para gestionar las acciones que puede realizar cada personaje. Existen varios tipos de acciones que pueden ser potenciadas haciendo uso de uno o más de sus decoradores. Cada personaje cuenta con tres acciones distintas y estas pueden ser personalizadas de distinta manera con los decoradores. Los decoradores ofrecen funcionalidad extra a la acción, añadiendo más daño a esa acción y/o añadiendo un efecto especial a la acción, que hará cambiar de estado al atacante o al atacado añadiéndole un efecto como quemadura, que resta vida en cada turno. Solo se puede realizar un efecto especial por acción, por ejemplo una acción con efecto especial decorada con uno o más decoradores con efectos especiales solo activaría el efecto del decorador más externo.

Patrón State:



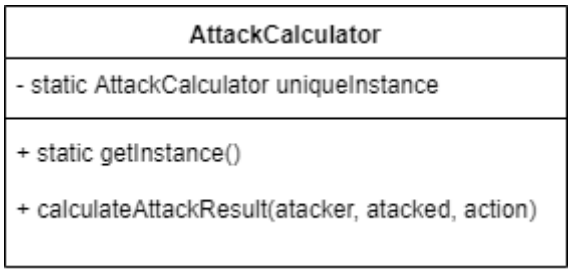
Se ha empleado un patrón state para gestionar los distintos estados por los que puede pasar un personaje. Todos los personajes tienen en todo momento un estado que puede producir varios efectos. En cada turno de ataque se llama al estado para que produzca su efecto. Los eventos que hacen cambiar de estado a los personajes son los ataques o la disipación de otro estado.

Patrón Abstract Factory:



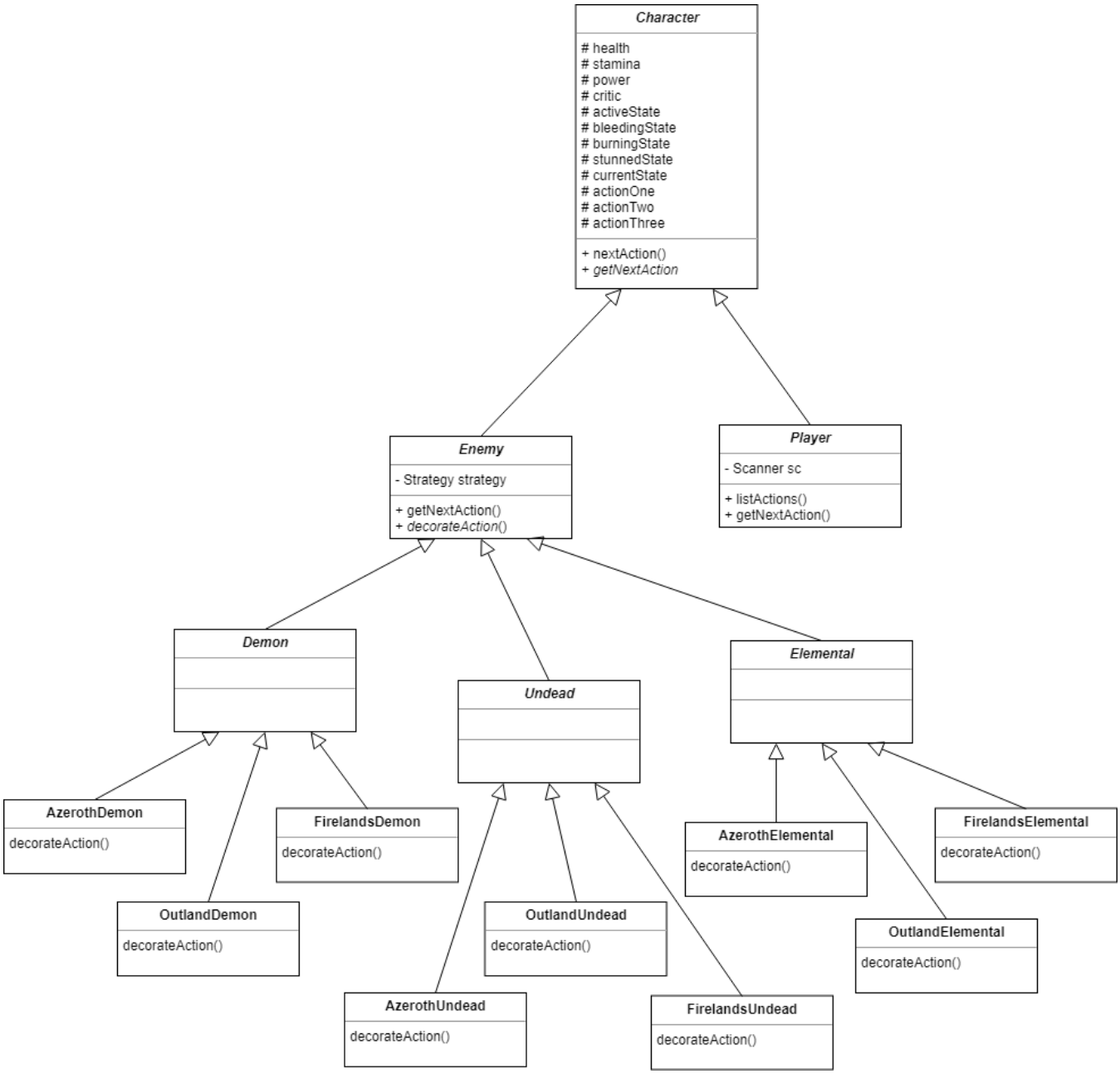
Se ha empleado un patrón abstract factory para gestionar la creación de los enemigos. Existen tres tipos de enemigo y por cada uno existe una variación según el mundo en el que se encuentre. El controlador del juego crea un tipo de fábrica aleatoriamente, cada fábrica crea enemigos de un mundo en concreto. Después el controlador le pide un tipo de enemigo a la fábrica y ésta devuelve el enemigo del mundo en concreto.

Patrón Singleton:



Se ha empleado un patrón singleton en la clase que calcula el daño de los ataques. Esta clase solo puede tener una instancia que usará el controlador del juego en cada ataque para calcular los resultados.

Patrón Template Method:



Se ha empleado un patrón template method en la elección del siguiente ataque de cada personaje. La elección del ataque se produce en el método declarado en la clase Character. En este algoritmo uno de sus pasos es la llamada al método getNextAction(), que se implementa de manera diferente en la clase Enemy o en la clase Player. Se ha implementado otro patrón template method en el método getNextAction() de Enemy, ya que uno de los pasos de este método es la llamada a decorateAction() que se implementa de manera distinta en cada clase hija.