**Patrones de diseño.**

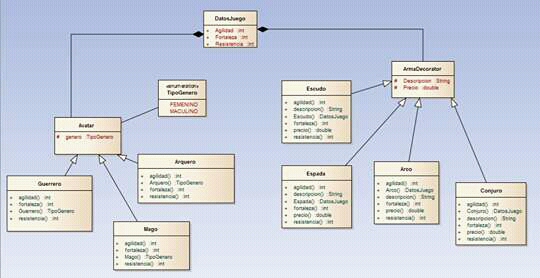
**Integrantes: Walter Mauricio Cuervo Barahona.**

**Lorena Lucia Maldonado.**

**Alba Ximena Ruiz Moreno.**

**Decorator:** Añade dinámicamente las responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender una funcionalidad.

**Ejemplo:** Se requiere crear un juego donde existan diferentes tipos de personajes (Arquero, Guerrero, Mago, etc.) que cuentan habilidades como Agilidad, fortaleza y resistencia, además de esto se cuenta con una serie de armas que permiten aumentar o disminuir las habilidades correspondientes a cada personaje.



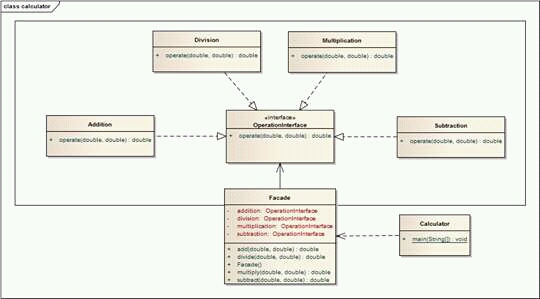
Se puede observar en el diagrama de clases planteado que cada uno de los Avatar hereda de la clase Datos los atributos agilidad, fortaleza y resistencia. Cada uno de tipo numérico que indica la cantidad de cada uno de estos, además de esto las armas también tienen un diseño que permite heredar los mismos atributos de la clase Datos con lo cual se podrá manejar las habilidades que cada una de estas otorga o disminuye.

Es decir, si se tiene un Avatar de tipo Guerrero con una agilidad de 80, una fortaleza de 95 y una resistencia de 20. Y se tiene por otro lado un arma tipo espada que tiene una Agilidad de +10 , fortaleza de +30 y una resistencia de -20 , cada una de las dos representa los mismos atributos por lo tanto facilita modificar los atributos del avatar de acuerdo al arma comprada.

Además, el patrón decorator permite manejar una sola clase que contiene estos datos garantizando menor gasto de tiempo en un futuro mantenimiento del sistema, es decir para este ejemplo facilitaría la creación de una nueva habilidad en el sistema.

**Facade:** Proporciona una interfaz unificada para un conjunto de interfaces de un subsistema. Define una interfaz de alto nivel que sea más fácil de usar.

**Ejemplo:** Fabricar una calculadora sencilla la cual reciba las operaciones del usuario y retorne los resultados deseados.



Se puede ver que en el diagrama existen objetos que representan una serie de operaciones básicas existentes dentro del sistema y cada una de estas están disponibles para el usuario.

La interface Facade permite controlar el sistema con el fin de manejar los datos entrantes y enviarlos a la opción deseada por el usuario, es decir que todas las instancias de las clases Operaciones se encuentran dentro de esta para así ejecutar las instrucciones recibidas por el usuario.

El patrón facade nos permite tener de forma más ordenada los datos y permite una mejor usabilidad y mantenimiento del sistema.