Desarrollo de Sistemas



Profesor:

* Saclier, Lucas

Alumnos:

* González Rouco, Lucas
* Gambacorta, Facundo
* Tabor, Joaquín
* Zalazar, Nahuel

Curso: 4°CSTC

Fecha de Entrega de 1ra instancia: 02/06/2020

Fecha de Entrega de 2da instancia: 09/06/2020

Fecha de Entrega de 3ra instancia: 16/06/2020

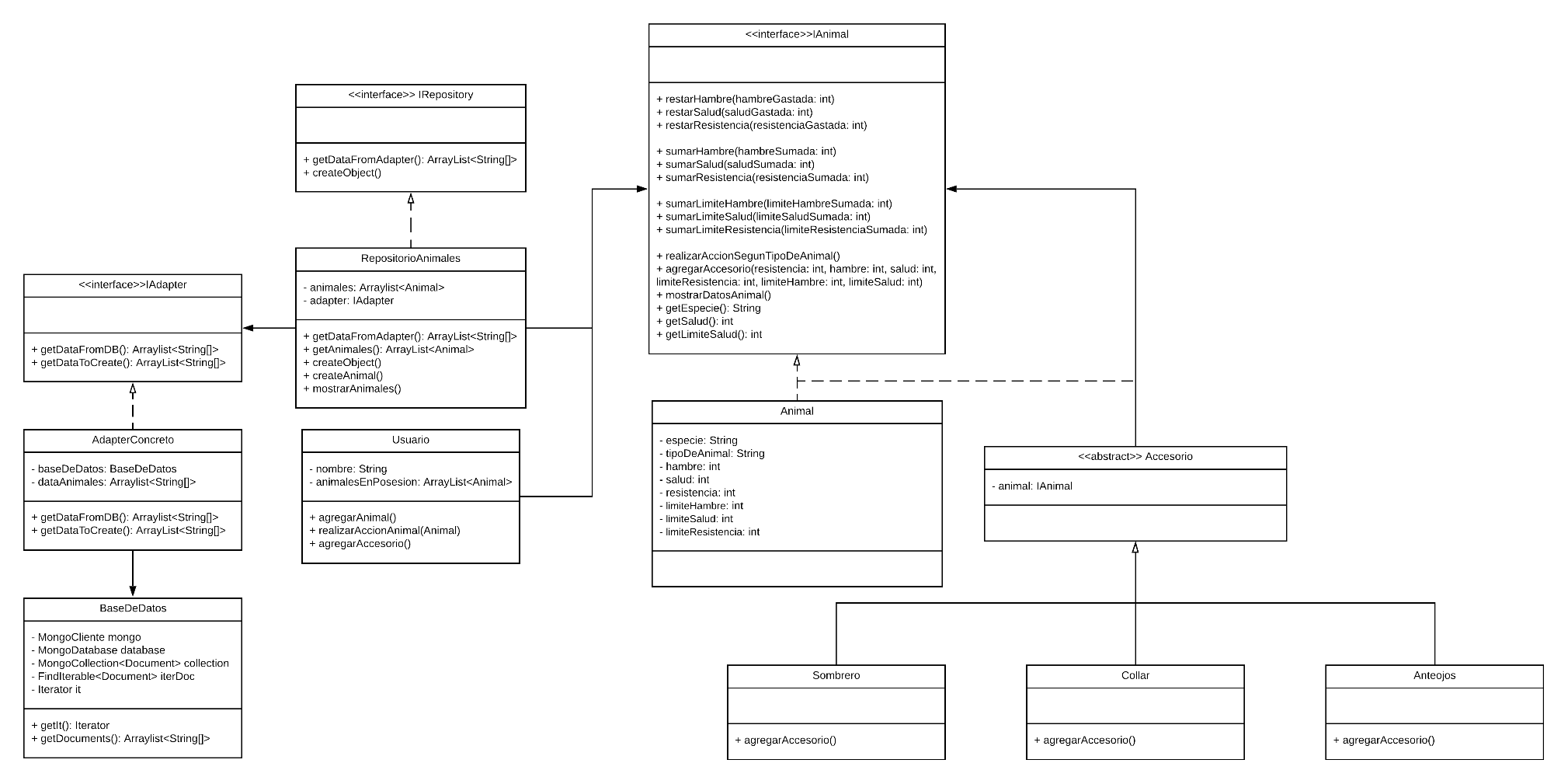
**Enunciado**

Somos contratados por Hasbro. Hasbro en los últimos tiempos (y más en la cuarentena) se dio cuenta que los juguetes ya no venden tanto. Nos contratan para desarrollar un sistema para crear un zoológico online. Nos proporcionan una base de datos con los datos de los animales que va a haber en el juego. El usuario va a poder agregar animales en su colección sin ningún tipo de restricción. Hasbro nos dice que a los animales se les van a poder agregar accesorios, estos accesorios deben considerarse como mejoras al animal, modificando aumentando la salud, el hambre, la resistencia y los límites de estos 3 atributos ya que todos deben inicializarse en 100 por defecto. Además, también nos piden que el usuario puede jugar con los animales, teniendo en cuenta la lógica (no se puede pasear a un pez, por ejemplo). Para jugar con los animales, el animal debe tener cierta salud, cierta resistencia y cierta hambre. Si no se cumple con estas condiciones, no se puede jugar con el animal.

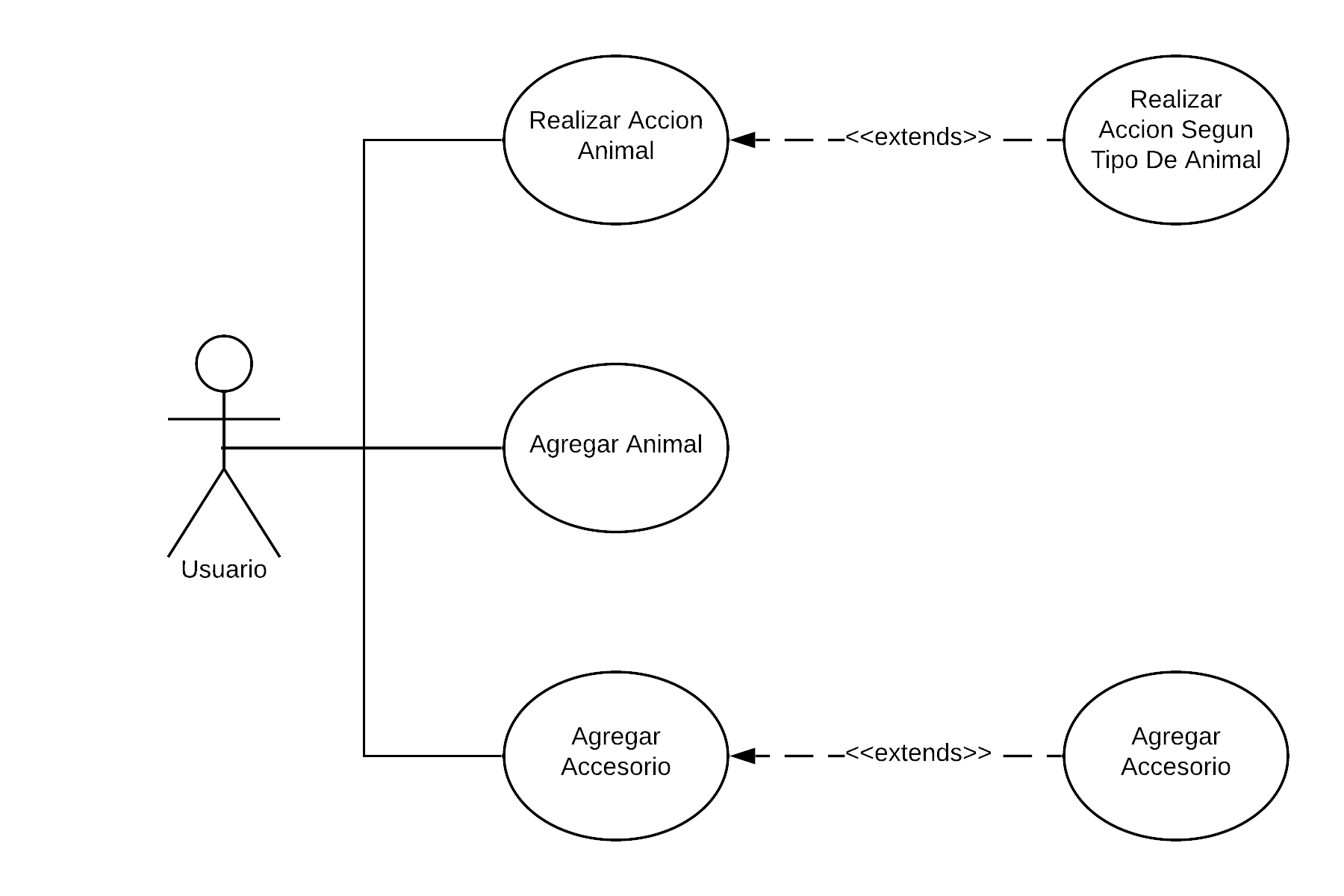
Para resolver esto, a simple vista, nos daríamos cuenta que tendríamos que usar

* El patrón Adapter para elegir que datos se traen de la base de datos.
* El patrón Decorator para agregar accesorios a los animales y que estos modifiquen los atributos del animal.
* El patrón Repository para guardar la información de los animales en el sistema.

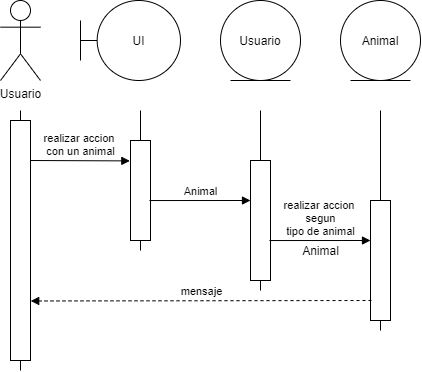
**Diagrama de clases**



**Diagrama de Casos de Uso:**



**Diagrama de Secuencia:**



**¿Qué es un patrón de diseño?**

Son soluciones generales y reutilizables para resolver un problema de la forma más eficiente posible.

**¿Qué tipos existen?**

Hay 3 tipos:

* Creacionales
* Estructurales
* De Comportamiento

**¿Cuál es su utilidad?**

Se utilizan para que haya ciertos estándares a la hora de escribir un código para que otros desarrolladores puedan entender más fácilmente que se quiso hacer y cómo. Además, son útiles para crear soluciones a problemas que se puedan volver a utilizar y evitar perder el tiempo.

* Los creacionales ayudan a encapsular y abstraer la creación de un nuevo objeto.
* Los estructurales sirven para especificar como una clase se relaciona con otra.
* Los de comportamiento sirven para gestionar los métodos en las clases y las relaciones y responsabilidades entre objetos.

**¿Cuáles utilizamos en este trabajo practico?**

Utilizamos:

* Adapter
* Repository
* Decorator

**¿Qué es el testeo unitario?**

Son una forma de comprobar que un fragmento de código funciona correctamente. Consisten más que nada en aislar una parte del código y comprobar que funciona a la perfección. Son pequeños test que validan el comportamiento de un objeto y la lógica.

**¿Qué es el refactoring?**

El refactoring es reestructurar un código, alterando su estructura interna sin cambiar su comportamiento externo.

La idea es que un código que está “sucio”, mejorarlo para conseguir que sea un código más “limpio”, que tenga un diseño más simple, que sea más fácil de arreglar, modificar y leer.

**Link del repositorio**

<https://github.com/lucasrouco/Tamagochi>