Trabajo Práctico Integrador

Número de grupo: 19
Alumnos: Oreskovic, Franco Juan – Reynoso, Elías – Piñero, Martín
Materia: Programación con Objetos 2
Universidad: Universidad Nacional de Quilmes
Año: 2025 (primer cuatrimestre)
Comisión: 2

Información de contacto:
-Oreskovic, Franco: kitofran15@gmail.com
-Piñero, Martín: //Poner mail
-Reynoso, Elías: ereynoso2040@gmail.com

1. Decisión de Diseño

Contamos con una clase App destinada a guardar las samples y las regionar en un ArrayList respectivamente, que luego son pasados por parámetro a los métodos que lo requieren en vez de hacer uso de una clase Singleton y acceder de manera global a los objetos almacenadas, ya que, en este nivel de capa de diseño, el acceso a instancias globales puede ser evitado entonces, las funcionalidades que requieren del acceso a la totalidad de objetos del sistema almacenados (Regions y Samples), reciben las mismas por parámetro luego de pedírselas a la clase de almacenamiento App.
Ya con las listas de objetos a su disposición, las distintas clases hacen los procesamientos necesarios para obtener los resultados que buscan
- Los usuarios de la clase ChangeableUser poseen StatePattern, un estado que va variando segun la cantidad de muestras que opinan y suben, este estado puede ser Basic o Expert.
- La clase Sample posee un comportamiento similar en su cambio de esto, pudiendo ser Open, ExpertOnly y Closed, este estado varia segun las Reviews (Opiniones) que contenga esta Sample y son almacenadas en un ArrayList.
- La clase Region cuenta con un Patron Observer para notificar a las Organizaciones sobre muestras que se cargan o se validan en dicha region. Al tener dos eventos que notificar, optamos por la implementacion de un EventManeger, una clase abstracta que tiene una lista por evento, para los subscriptores de los mismos (Al tener solo dos eventos nos parecio mas conveniente hacelos asi, si fueran necesarios más eventos se cambiaria la un HashMap o algun otro tipo de almacenamieto).
El evento de notificar una Carga de una muestra comienza en la instancia de App a la que se suba, cuando se carga una muestra, se busca la region en base a la Posicion de la muestra subida y se notifica a la Region, que después notifica a las Organizaciones.
En cambio, el evento de Validacion de una Muestra, empieza en el cambio de estado de dicha muestra, cuando en esta Opinan dos expertos los mismo, el estado cambia a Closed (no puede opinar más nadie y se cierra), si la muestra pertenece a alguna Region se le notifica que esta cambiando el estado a Closed (Verificado)

2. Detalles de la Implementación

*Patrones de diseño utilizados:

-Patron Observer
EventManager => ConcreteSubject
IObserver => Observer
Organizacion => ConcreteObserver

-Patron State (2 casos)
Usuario
ChangeableUser => Context
IUserState => State
Basic, Expert => ConcreteState

Sample
Sample => Context
ISampleState => State
Open, ExpertOnly, Closed => ConcreteState

-Patron Strategy
Organizacion => Context
FuncionalidadExterna => Strategy
//A Implementar => ConcreteStrategy