LEM

Trabajo Final: A la caza de las vinchucas
Asignatura: Programación con Objetos II
Institución: Universidad Nacional de Quilmes
Año: 2023

Integrantes:

Luciana Merlino – lmerlinomat@gmail.com
Ezequiel Rigo – rigoezequiel@gmail.com
Martín González Buitrón – martingonzalezbuitron@gmail.com

Decisiones de diseño

Para nuestro modelo de negocio del Trabajo Final (TF) *“A la caza de las vinchucas”* decidimos enfocarnos fuertemente en el modelado y diseño de clases usando UML para luego así dar lugar a las implementaciones a través de la metodología de desarrollo guiada por tests, TDD (en inglés, “*Test-driven development”*), teniendo presente los principios SOLID vistos en clase. Por último, esperamos lograr detectar instancias de refactoring para poner en práctica lo visto.

Inglés como lenguaje de comunicación
El inglés no es nuestro lenguaje nativo aunque decidimos optar por un diseño de UML y producción de software haciendo uso del mismo desde el día uno. Con esta decisión perseguimos la finalidad de poner en práctica y adquirir experiencia en la selección/asignación de nombres que representen y expresen el dominio del negocio en este lenguaje. Además, queremos mencionar que el inglés es la lengua mundialmente aceptada para la comunicación en el desarrollo de software y creemos que esta decisión ayuda y complementa la lectura de la bibliografía en inglés proporcionada por la cátedra como así la consulta en foros y sitios web durante el desarrollo de este TF. Por último, sabemos que quizás haya errores ortográficos, gramaticales o simplemente mal uso del vocabulario. Aprender de esto último es lo que buscamos.

Pasos en la construcción del UML
A continuación se intentan enumerar los pasos que consideramos haber atravesado en el proceso de diseño de la solución utilizando UML

1. Creación Objetos principales
2. Atributos de los objetos
3. Dependencias/composición entre objetos
4. Herencia de objetos
5. Detección de protocolos (Implementación de interfaces)
6. Decisión de tipos (Clases o Enumerativo)
7. Adaptaciones a PD diseño

Implementaciones

Nos parece importante mencionar que desde el inicio de nuestra solución de negocio optamos por implementar un objeto de una única existencia que tenga como responsabilidades almacenar los objetos creados para poder ser localizados, consultados y operados globalmente por otros objetos. Este objeto responde al patrón de diseño Singleton el cual estudiamos por nuestra cuenta para poder implementarlo.

También creemos relevante mencionar la utilización de interfaces para el manejo de tipos en lo que respecta a las opiniones dadas por los usuarios en las muestras como así cuando se

le pide el resultado a ésta. Utilizamos enumerativos que implementan una interfaz con las opiniones posibles y otra interfaz que otorgue la posibilidad de responder a la situación de indefinido cuando se le pregunta el resultado a una muestra.

Finalmente, queremos mencionar la implementación de 2 patrones de diseño Observer en el cual una misma Clase cumple el rol de Sujeto para uno de ellos y Observador para el otro. Esta implementación se torna confusa de interpretar enfocándose sólo en el código pero nos facilitó encontrar una solución para el diseño elegido haciendo uso del Singleton previamente mencionado.

Patrones de diseño

State

- User
  - User -> context
  - UserState -> State
  - BasicUser -> ConcreteStateA
  - ExpertUser -> ConcreteStateB
- Sample
  - Sample -> Context
  - SampleState -> State
  - UnverifiedSample -> ConcreteStateA
  - VerifiedPartialSample -> ConcreteStateB
  - VerifiedSample -> ConcreteStateC

Observer

- **Observer 1**
  - IZoneCoverage -> Sujeto
  - ZoneCoverage -> ConcreteSubject
  - IOrganization -> Observer
  - Organization -> ConcreteObserver
- **Observer 2**
  - System -> Sujeto
  - ZoneCoverage -> Observer

Singleton

- System -> singleton

Bibliografía

- GAMMA
- XUnit
- SOLID