Sara Cubero García-Conde

Introducción a la ingeniería del software

2023

Índice:

1. Introducción

1.1 Contexto y propósito del documento

El presente documento tiene como objetivo detallar el análisis y las soluciones propuestas respecto al diseño recibido para la programación de una aplicación. En este sentido, se presenta una revisión detallada del diseño, con el fin de identificar posibles errores o inconsistencias que puedan afectar la implementación de la aplicación.

Asimismo, se plantean soluciones a las posibles deficiencias identificadas. Finalmente, se describe la implementación realizada, destacando las principales características de la solución propuesta y su correspondencia con el diseño original.

El proyecto se ha realizado en Java con Eclipse, bajo aceptación de mi tutora.

2. Análisis del diseño recibido

El diseño está basado en una arquitectura orientada a objetos, que consta de varias clases interconectadas que modelan los diferentes componentes del sistema. Las clases más importantes son la clase GesFlota, que actúa como el controlador principal del sistema y realiza todas las operaciones relacionadas con la flota de buques, y las clases Puerto, Buque, Producto y Operación, que representan los componentes clave del sistema y proporcionan información sobre los puertos, los buques, los productos y las operaciones respectivamente.

El sistema permite llevar a cabo cinco operaciones básicas: editar puerto, editar buque, mostrar el estado actual de los buques, operar un buque y generar un resumen mensual de un buque en particular. Estas operaciones son llevadas a cabo por métodos definidos en la clase GesFlota, que a su vez utilizan los métodos y atributos de las demás clases.

2.1 Identificación de errores o inconsistencias en el diseño

En líneas generales he podido construir sin problema la estructura de la aplicación, pero no he podido completar la implementación sin algunos cambios. Algunas de las inconsistencias detectadas en el diseño son:

1) Falta de algunos métodos elementales para la implementación, no aparecen en el diagrama de clases, como el método estadoBuques y resumenMensual.
2) Dado que se define la lista de operaciones en la clase GesFlota y se realiza todo el manejo en esta, y no se guarda en memoria ningún objeto de la clase Calendario, por lo tanto, he optado por utilizar el listado de operaciones ya que me pareció mas eficiente para trabajar el sistema.
3) La clase Operación tiene demasiados atributos, algunos podrían haberse aprovechado para distintas funciones.
4) Los métodos anadirPuerto y anadirBuque no deben añadir sino editar basándonos en la lógica de la práctica por tanto los nombres no son lógicos.
5) Encuentro redundantes las clases Refinado y NoRefinado para la lógica de la aplicación.
6) No encuentro necesario declarar a nivel global la variable opcionSeleccionada ocupa espacio en memoria de forma irrelevante.

3. Soluciones propuestas

Para solucionar los errores o inconsistencias anteriormente descritas. He llevado a cabo los siguientes arreglos, cada número corresponde a la solución del número de inconsistencia proporcionado en el apartado 2.1:

1) He agregado los métodos necesarios a la implementación como son los constructores, los métodos getter y setter y los métodos mencionados de estadoBuques y resumenMensual.
2) He creado la clase Calendario para seguir la implementación del diagrama de clases propuesto, pero no ha sido utilizada.
3) He utilizado el diseño de Operación pese a su redundancia para basarme en el diseño proporcionado lo más posible.
4) He nombrado los métodos anadirBuque y anadirPuerto tal y como se menciona en el diseño, a pesar de la incoherencia de estos, para basarme en el diseño proporcionado.
5) He generado la arquitectura de Refinado y NoRefinado y aplicado su lógica de abstracción con la clase Producto, para basarme en el diseño proporcionado.
6) He declarado la variable por seguir el diseño proporcionado.