Integrantes
Sebastian Ramirez
Manuela Pacheco
Santiago Castro

Taller 5: Análisis de patrones de diseño

Diciembre 02, 2022

OBJETIVOS

Para este análisis de patrones de diseño, se optó por escoger una parte del proyecto de DiscLib, el cual es utilizado en distintos proyectos de la universidad, el cual consiste en la implementación de distintos tipos de estructuras de datos, por otro lado, este no utiliza el paradigma de la programación orientada a objetos y no está diseñado en java, los patrones de diseño se pueden identificar y se va a hacer énfasis en partes específicas del proyecto.

URLl del proyecto: https://github.com/DPOO2022-2SMS3/Taller-.5.git

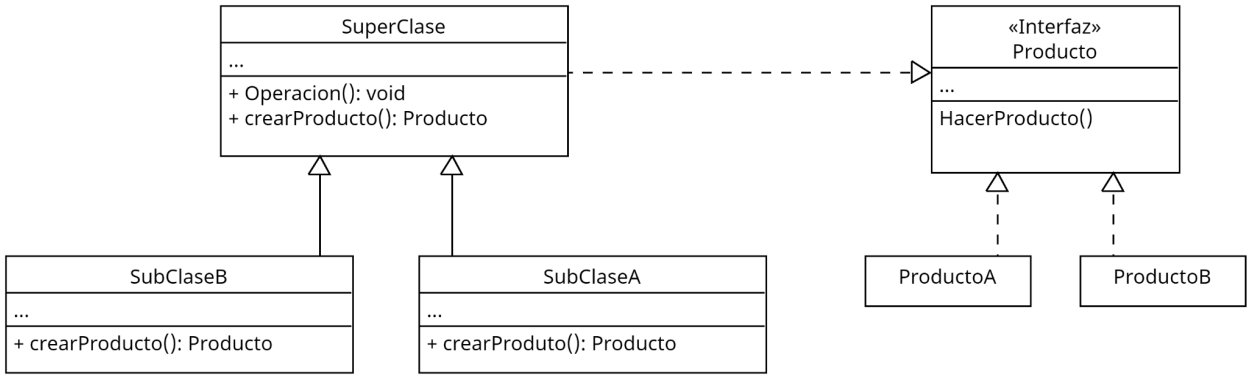
IDENTIFICACIÓN DEL PATRÓN

El proyecto consta de 4 archivos, ADT, Algorithms, DataStructure y Utils, en donde únicamente nos vamos a centrar en ADT, el cual para la implementación del patrón es una superclase, y en Data Structure, en donde se encuentran todas las subclases del ADT. Teniendo lo anterior en cuenta, se puede ver como en el ADT hay una serie de clases con el nombre correspondiente a las estructuras que se están implementando en el proyecto, dentro de ellas están los respectivos métodos de cada uno, mientras que en DataStructure es un poco más complejo, gracias a que tiene 2 subclases para cada una de las implementaciones de las estructuras de datos que tiene las superclases que corresponden a el archivo ADT

PATRÓN: FACTORY METHOD

Teniendo en cuenta lo anterior el patrón de diseño que se encontró es el factory method, que consiste en proporcionar una interfaz para crear objetos en una superclase, mientras permite a

las subclases alterar el tipo de objetos que se crearán, este patrón consta de; la superclase y sus n subclases, la interfaz y los objetos que hacer parte de ella.

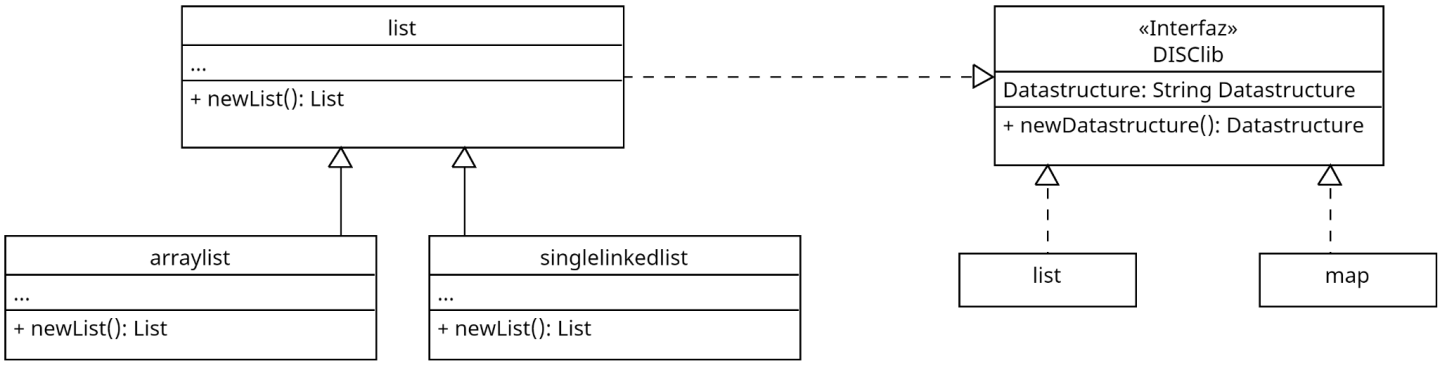


La clase Creadora declara el método fábrica que devuelve nuevos objetos de producto, es importante que el tipo de retorno de este método coincida con la interfaz de producto, las subclases sobrescriben el método base de creación, de modo que devuelva un tipo diferente de producto. En el lado izquierdo se encuentran Producto el cual declara la interfaz, que es común a todos los objetos que puede producir la clase creadora y sus subclases, mientras que ProductoA y ProductoB son los Productos Concretos con distintas implementaciones de la interfaz de producto.

PATRÓN APLICADO AL PROYECTO

Teniendo lo anterior en cuenta, podemos observar a DiscLib como la interfaz, la cual permite crear distintos tipos de estructura de datos, en donde cada una de ellas tiene sus respectivos métodos y son productos abstractos, el ADT como la superclase de los productos, el cual contiene una serie de clases que corresponden a cada estructura con distintos métodos de operación. Como este proyecto consta de varias clases, se va a identificar el patrón descrito anteriormente de una de ellas la cual es List, esta clase permite al usuario crear una lista encadenada o un arreglo, en donde permite realizar varios métodos que corresponden a la creación de la misma, de modo que list juega el papel de superClase, para la creación del producto, las clases están dentro de DataStructure el cual es el encargado de almacenar las subclases de las distintas superclases, por lo que dentro de él se encuentran las clases arraylist y singlelinkedlist que corresponden a las subclases de list, lo interesante de estas subclases, es que dentro de ellas se encuentran todos los métodos que crean la estructura de datos correspondientes, y a su vez se encuentran los métodos para modificarla.

De esta manera, se puede ver como entra un parámetro de creación específico en la superclase y pasa por las subclases para modificar o crear el producto requerido por el usuario mediante DiscLib.



VENTAJAS

- Bajo acoplamiento
- Facilita el mantenimiento, pues las clases mantienen un alto grado de autonomía
- Ahorra recursos del sistema, en lugar de reconstruir objetos, utiliza los ya existentes
- Facilita las pruebas, pues las funcionalidades de sus clases pueden ser testeadas individualmente

DESVENTAJAS

- Requiere un numero elevado de clases
- La extensión es complicada, pues se deben modificar varias clases solo para ampliar una familia de productos

Al investigar más a fondo pudimos observar que las ventajas superaban las desventajas para las necesidades especificas del proyecto. Además, notamos que la desventaja de la extensión es reducible con un buen diseño pre implementación, teniendo en cuenta que el proyecto siempre tendría una cantidad de estructuras de datos constante.