Práctica 7: Refactorizaciones

Manuel Caballero Rábago

Estado del sistema antes y después de refactorizar:

Diagrama de clases inicial:

Captura de pantalla de un celular con letras

Descripción generada automáticamente

Diagrama de clases final:

Valores de las métricas antes y después de refactorizar:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WMC | | WMCn | | DIT | | NOC | | CBO | | CCOG | |
|  | Ini | Fin | Ini | Fin | Ini | Fin | Ini | Fin | Ini | Fin | Ini | Fin |
| Tienda | 35 |  | 3.89 |  | 0 |  | 0 |  | 2 |  | 33 |  |
| Vendedor | 8 |  | 1.14 |  | 0 |  | 2 |  | 3 |  | 1 |  |
| VendedorEnPlantilla | 2 |  | 1 |  | 1 |  | 0 |  | 2 |  | 0 |  |
| VendedorEnPracticas | 1 |  | 1 |  | 1 |  | 0 |  | 1 |  | 0 |  |

En cuanto a los valores de CBO inicial se obtienen de la siguiente forma:

* Tienda: Tiene un valor 2 ya que depende de las clases Vendedor y TipoVendedor, mientras que ninguna clase depende de ella.
* Vendedor: Tiene un valor de 3 ya que dependen de ella las clases Tienda, VendedorEnPlantilla y VendedorEnPracticas. No depende de nadie.
* VendedorEnPlantilla: Tiene un valor de 2 ya que depende de las clases Vendedor y TipoVendedor, mientras que ninguna clase depende de ella.
* VendedorEnPracticas: Tiene un valor de 1 ya que depende de la clase Vendedor, mientras que ninguna clase depende de ella.

Refactorizaciones posibles:

1. Extract Method
2. Extract Class
3. Pull Up Method
4. Move Method
5. Rename Method
6. Introduce parameter Object
7. Replace magic Number with symbolic constant
8. Replace conditional with polymorphism

Lo primero que hice fue fijarme en que el método vendedores() estaba mal implementado, ya que asignaba incorrectamente los roles a los vendedores. Los primeros vendedores que aparecen en el txt son los seniors, seguidos de los junios, y por último, los de prácticas. Es por ello que lo que hice fue arreglar esa parte del código y eliminar el código del método buscaVendedor(ya que era el mismo que acababa de arreglar) sustituyéndolo por una llamada a vendedores() (Se puede considerar como una refactorización ***Extract Method***). Por lo que el método buscaVendedor() llama a vendedores y así nos evitamos el tener duplicada la lectura del txt (reduciendo así el valor de WMC y CCOG).

Tras esto me fijé en el método anhadeVenta. El switch añadía complejidad al método y esto se podía solucionar extrayendo el código a su clase correspondiente (***Move method***). Es por ello por lo que esta implementación la añadí en la clase VendedorEnPlantilla, ya que ahí es donde pinta más. Además, al ejecutar el método desde esa clase, podemos aplicar ***Replace magic number with symbolic constant*** (ya que la prima por venta la podemos poner como constante en la clase VendedorEnPlantilla, mientras que en tienda no quedaba bien hacerlo).

Reflejo de las mejoras de calidad del código en las métricas:

install-file -Dfile=fundamentos.jar -DgroupId=fundamentos -DartifactId=fundamentos -Dversion=1.0 -Dpackaging=package

<https://examples.javacodegeeks.com/enterprise-java/maven/maven-resolve-missing-artifact-error-example/>