Raúl Elizalde Roldán

Practica 5A

El objetivo de esta práctica es analizar las diferentes métricas y aplicar refactorización en las diferentes clases y métodos.

Calculo Métricas pre-refactorizacion

Se ha realizado el calculo de todas las métricas que se ven reflejadas en la siguiente tabla. WMC y CCog están indicadas en el propio código.

METRICAS						
	WMC	WMCn	CBO	DTI	NOC	Ccog
Cliente	14	1,4	4	0	0	7
Credito	16	1,77777778	6	1	0	8
Cuenta	2	1	3	0	2	0
CuentaAhorro	18	1,63636364	4	1	0	7
CuentaValor	5	1,66666667	2	1	0	3
Debito	8	1,33333333	4	1	0	2
Movimiento	6	1	2	0	0	0
Tarjeta	3	1	5	0	2	0
Valor	7	1	1	0	0	0

Una vez calculadas las métricas se realiza refactorización.

Refactorización

Se han utilizado los diferentes métodos vistos en clase y se han ordenado en función de la aparición de ellos.

-Rename method: Este consiste en renombrar aquellos métodos que no dan la suficiente información sobre lo que tratan. En la clase movimiento podemos ver como setI, setF y setC no dan una idea correcta de lo que tratan. Se han sustituido por setImporte, setFecha y setConcepto respectivamente.

-Extract method: Cuando se repiten varias veces las mismas instrucciones se puede crear un método para ahorrar. En la clase cuentaAhorro se crea uno para realizar la comprobación de la cantidad ya que esta comprobación se repite varias veces y contiene 2 condiciones(2 ifs).

-Extract class: Se basa en crear una nueva clase para aquellos atributos y métodos que existen en otra pero no que no tratan lo mismo que la clase. Se ve que en la clase cliente se tiene información muy concreta sobre la dirección, por o que se ha creado una nueva clase Direccion con los atributos de ella(localidad, dirección y zip), su constructor y observadores.

-Pull up: Si existen métodos duplicados en clases heredadas se crea en la superclase el método que afecte a ambos. En este caso, tanto para la clase cuenta como tarjeta no existen métodos que se dupliquen en sus clases heredadas. No obstante, en la clase cliente se realiza

el calculo del saldo de los 2 tipos de cuentas por lo que se ha creado un método abstracto en la superclase cuenta que heredaran los 2 tipos de cuentas para calcular este saldo.

-Move method: Existe cuando en un método solo se utilizan atributos de métodos ajenos a este y no se utiliza este para nada.

-Parameter object: Esto se crea cuando para un método se exigen los mismos atributos en varios metodos

-Replace Magic Number with Symbolic Constant: Cuando existe un número que se repite en varias ocasiones se debe crear una constante para ese número. Vemos como aparece en crédito la comisión múltiples veces, por lo que creamos la constante comisión.

-Replace Conditional with Polymorphism: Se realizaría en caso de que existiesen atributos que dependiesen entre ellos creando una superclase con sus clases heredadas. En este programa no se cumplen están condiciones.

Tras realizar todos los cambios con refactorización se rehace la tabla:

Calculo métricas post-refactorización

METRICAS POST-REFACTORIZACIÓN						
	WMC	WMCn	CBO	DTI	NOC	Ccog
Cliente	7	1,16666667	4	0	0	1
Credito	16	1,77777778	6	1	0	8
Cuenta	3	1	3	0	2	0
CuentaAhorro	18	1,63636364	4	1	0	5
CuentaValor	5	1,25	2	1	0	4
Debito	8	1,33333333	4	1	0	2
Movimiento	6	1	2	0	0	0
Tarjeta	3	1	5	0	2	0
Valor	7	1	1	0	0	0
Direccion	5	1	0	0	0	0

Como podemos observar vemos que se ha creado una nueva fila(clase dirección) y que han disminuido otros valores. Sin embargo, estás métricas no reflejan la mejora completa del código, ya que este ha sufrido cambios en el nombre de los métodos o en la creación de constantes para números sueltos y estos cambios no se ven reflejados en las métricas. Además la manipulación de otros métodos o clases no se ven perfectamente reflejados en esta tabla ya que el mayor cambio es en la clase cliente cuyo wmc baja de 14 a 7 y en cuenta ahorro que baja de 7 a 2.