Patrones de diseño

Sistema de Gestión de Cobros Andi Asistencia

Revisión 2

Noviembre de 2016

Ficha del documento

Fecha	Revisión	Autor	Verificado dep. calidad
Noviembre 5 de 2016	# 2	Karen Aldana, Giovanni Galvis, Pablo Vallejo	Aprobado

Documento validado por las partes en fecha: [05/11/2016]

Por el cliente	Por la empresa suministradora
Mapfre - Asistencia	Kenai Inc
Fdo. Giovanni Galvis	Fdo. Pablo Vallejo

Contenido

Índice de Ilustraciones

1. INTRODUCCIÓN

Template Method es un patrón de comportamiento. Este tipo patrones ayudan a resolver problemas de interacción entre clases y objetos. Este patrón nace de la necesidad de extender determinados comportamientos dentro de un mismo algoritmo por parte de diferentes entidades. Es decir, diferentes entidades tienen un comportamiento similar pero que difiere en determinados aspectos puntuales en función de la entidad concreta.

La solución que propone el patrón Template Method es abstraer todo el comportamiento que comparten las entidades en una clase (abstracta) de la que, posteriormente, extenderán dichas entidades. Esta superclase definirá un método que contendrá el esqueleto de ese algoritmo común (método plantilla o template method) y delegará determinada responsabilidad en las clases hijas, mediante uno o varios métodos abstractos que deberán implementar.

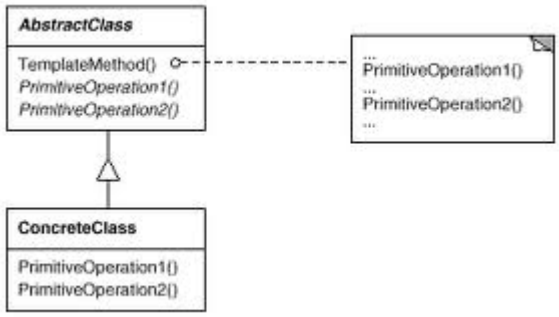

Ilustración 1 Template Method

Como puede verse en el anterior diagrama, la superclase contiene el método plantilla, ese método con el algoritmo que comparten las entidades concretas (subclases). Como se puede apreciar, define una o varias operaciones concretas en forma de métodos abstractos que son usados por el método plantilla y que deben ser implementadas por las clases hijas. Dichos métodos abstractos representan los comportamientos concretos de las entidades.

1.1 Ventajas

- La reutilización de código
- La superclase base invoca los métodos de las subclases.

1.2 Desventajas

- Ambigüedad si no se escribe bien

- Si el método plantilla llama demasiados métodos abstractos, se cansará pronto de utilizar AbstractClass como superclase.

2. APLICACIÓN DEL PATRON EN EL PROYECTO

El sistema presenta comportamientos similares en varias funcionalidades, estás son: carga de ventas y carga de pagos, igualmente sucede en generación de archivos de cobro y generación de reporte de clientes activos.

Como el Template Method busca eliminar la duplicidad de código, y, por lo tanto, disminuir el tiempo requerido al modificar funcionalidades compartidas. Esto se logra extendiendo las funcionalidades comunes desde una clase principal a varias clases derivadas.

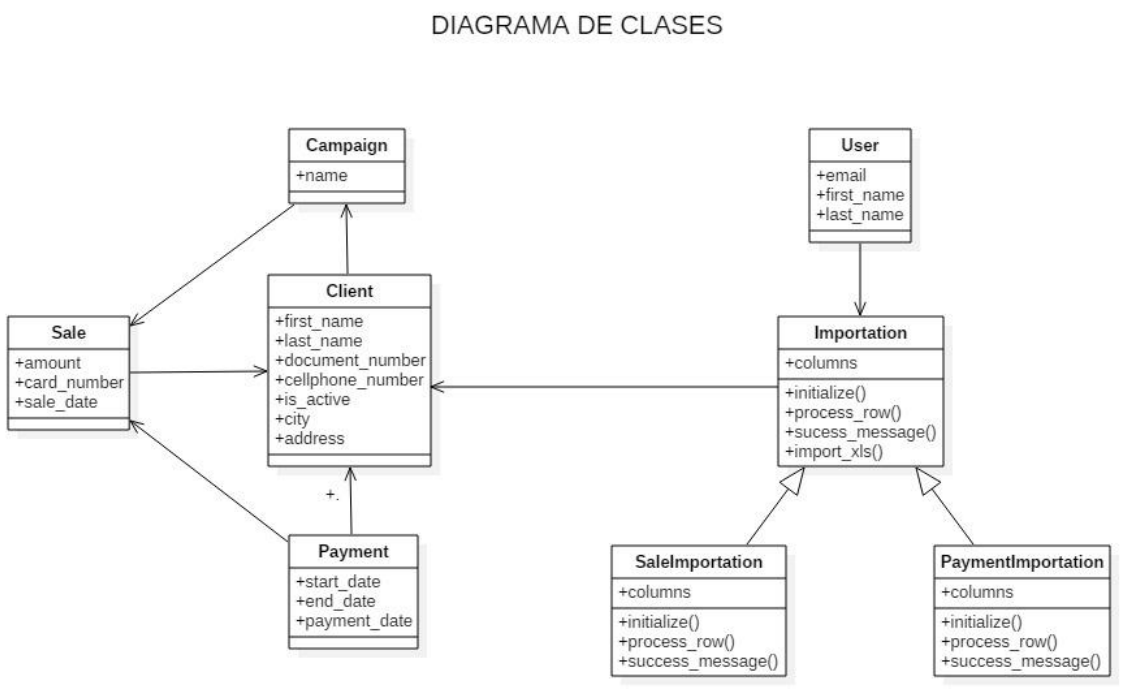

Ilustración 2 Diagrama de clases

En la imagen se observa una clase principal Importation que da origen a dos clases derivadas SaleImportation y PaymentImportation, estas corresponden a la carga de ventas y a la carga de pagos respectivamente. El atributo columns (llave, valor) permite identificar al cliente en el

momento de cargarse en el sistema e igualmente lo identifica posteriormente a medida que se registran los pagos efectuados, y se requiere información histórica.

Al tratarse de ingreso de información relacionada al sistema, se evidencia que el método es idóneo para su implementación. En la siguiente iteración se observará este mismo tipo de diseño, en la solución de los casos de generación de cobros y generación de reportes con clientes activos.

3. BIBLIOGRAFIA

- https://www.ecured.cu/Template_Method
- https://www.adictosaltrabajo.com/tutoriales/patron-template-method/
- https://es.wikipedia.org/wiki/Template_Method_(patr%C3%B3n_de_dise%C3%B1o)