Equipo: Cerebros de Pollo
Cruz González Irvin Javier
Ugalde Ubaldo Fernando
Ugalde Flores Jimena

Modelado y Programación

Factory, Abstract Factory y Builder
Practica 04

31716198-2
31714149-8
31816536-1
14-10-2022

1. Menciona los principios de diseño esenciales del patrones Factory, Abstract Factory y Builder. Menciona una desventaja de cada patrón

   Patrones de Creación que por lo general cumplen con el principio *Open-Closed.*

   a) Factory

   ✓ Define una interfaz para crear un objeto,pero deja que sean las subclases quienes decidan que clases instanciar. Permite que una clase delegue en sus subclases la creación de objetos.

   ✓ Factory elimina la necesidad de ligar clases especificas de la aplicación del código,este mismo sólo trata con la interfaz Producto;además puede funcionar con cualquier clase ProductoConcreto definida por el usuario.

   ✗ Una *desventaja* es que los clientes pueden heredar de la clase padre simplemente para crear un determinado objeto. La herencía esta bien cuando el cliente tiene que heredar de todos modos de la clase padre, pero si no es así estaría introduciendo una nueva vía de futuros cambios.

   b) Abstract Factory

   ✓ Proporciona una interfaz para crear familias de objetos relacionados o que dependen entre sí,sin especificar sus clases concretas.

   ✓ *Open-Closed.*
   " Objects or entities should be open for extension but closed for modification"
   Abstract Factory es la interfaz que provee un método para la obtención de cada objeto que pueda crear, de este modo si queremos crear estos objetos no implicaría estar modificando el código existente sino solo debemos extenderlo, en ningún momento se modifican las factorias concretas (familias de productos) cumpliendo así este principio.

   ✗ *Es dificíl dar cabida a nuevos tipos de productos*, es decir cuando se añaden nuevas familias de productos o cambian los existentes afecta a toda las familias creadas o subclases.

   c) Builder

   ✓ Separa la construcción de un objeto complejo de su representación,de forma que el mismo proceso de construcción puede crear diferentes representaciones.

   ✓ Proporciona una clara separación entre la construcción y la representación de un objeto además permite la representación interna de los objetos.

   ✎ *Dato:* Factory requiere que todo el objeto se construya en una sola llamada de método, mientras que Builder se enfoca en construir un objeto complejo paso a paso.

   ✗ Una desventaja es que *permite variar la representación interna de un producto* es decir la interfaz permite que el contructor oculte la representación y la estructura interna del producto,dado que se construye a través de una interfaz abstracta todo lo que hay que hacer para cambiar la representación interna del producto es definir un nuevo tipo de constructor.

Referencias

[1] Gamma, E., Helm, R., Johnson, R. Vlissides, J. (1994). Design Patterns:Elements of Reusable Object-Oriented Software.

[2] `https://es.wikipedia.org/wiki/Abstract_Factory`