Ingegneria del Software 1: Design Pattern

Due on Martedi, Aprile 15, 2014

Contents

1 Introduction

2 Creational Patterns

Inside the class of the creational patterns we can distinguish between two main pattern:

- **class creational pattern** uses inheritance to change how the different classes are intantiated.
- **object creational pattern** delegate the instantiation of the object to another object.

The object creational pattern become important when it is necessary to compose a smaller set of composed behavior into complex one. Thus, creating an object with a particular behavior is more complex then simply instantiating a new class.

2.1 Prototype

2.2 Abstract Factory (Kit)

2.2.1 Goal

The goal is to create an interface which allows **to create families** of related or dependent objects, *without specifying* their concrete classes.

2.2.2 Structure

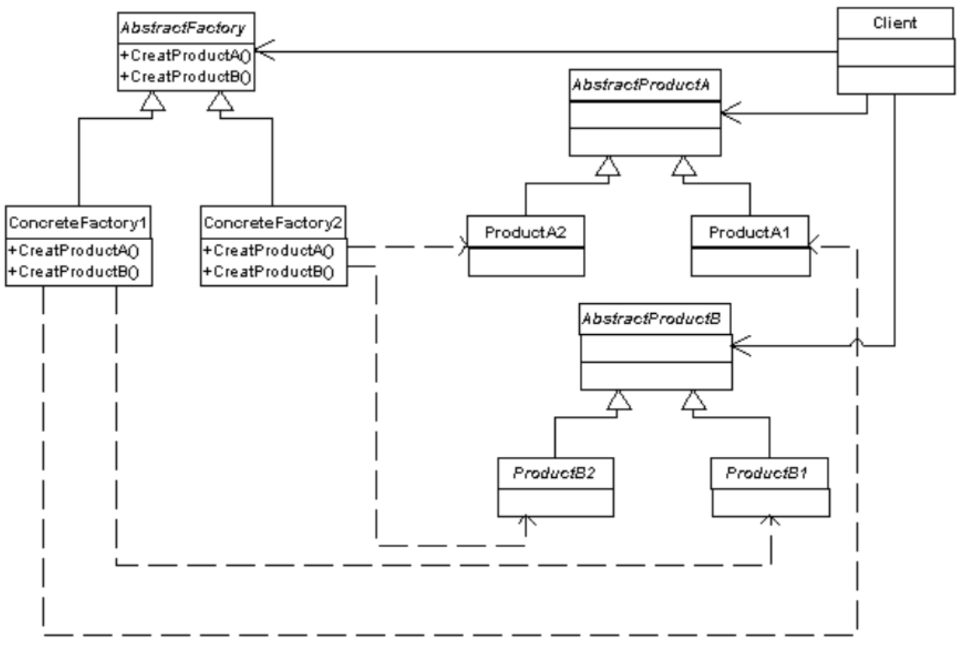

Figure 1: Abstract Factory UML.

The UML of Figure 1 is composed by the following classes:

- *AbstractFactory*: is an abstract class (see the italic style of the text), which declares the interface for the operations that allows to create *abstract* products

- *ConcreteFactory*: are the concrete classes that allows to create *concrete* products (Note the hierarchical relation between the AbstractFactory and the ConcreteFactory). Each product is a family of related dependent objects AbstractProductA and AbstractProductB, and in particular, for the ConcreteFactory1 the products are ProductA1 and ProductB1, while for the ConcreteFactory2 the products are ProductA2 and ProductB2 (Note the uses relation between the Factories and the products).
- *AbstractProduct*: declares an abstract product which can be aggregated to other products using appropriate factories
- *ConcreteProduct*: are concrete products which extends particular abstract products.

2.2.3 Examples

Using this pattern an abstract framework (factory) is defined, which allows to create objects that follow a general pattern. At run-time the abstract factory is paired with any concrete factory that allows to create the objects that follow the concrete pattern. In other words, the Abstract Factory is a super-factory which creates other factories (Factory of factories).
The classes that are involved in the Abstract factory pattern are:

- *Abstract Factory* is the interface the the ConcreteFactories must implement
- *Concrete Factories* implements the operations and allow to create concrete products

2.3 Builder

2.3.1 Goal

Separate the construction of a complex object from its representation, so that the same construction process can create different representations.

2.3.2 Motivation

2.4 Singleton

2.5 Factory Method (Virtual Constructor)

2.5.1 Goal

Define an interface for creating an object, but let sub-classes decide which class to instantiate.