IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 3, Ver. II (May-Jun. 2016), PP 36-37
www.iosrjournals.org

Factory Pattern in Model of Object Oriented Application Development

Gaurav Pradip Pande[1],Umesh Kulkarni[2]
[1](Department Of Computer Engineering, VIT University Of Mumbai, India)
[2](Department Of Computer Engineering, VIT University Of Mumbai, India)

Abstract: *Design pattern is a salient milestone in the development of coding techniques. Code reusability is the prime motive of design patterns and it promotes code development by finding the existing solution which could be as close as it could be to the new design problem. Generally small scale applications ignore the design patterns and coding is done in traditional way, patterns are considered to develop large scale applications like IDEs. Proposed work suggests use of Factory pattern in small scale application development in object oriented languages. Factory pattern creates hierarchical abstraction of objects.*
Keywords : *design pattern, factory pattern, code reusability*

I. Introduction

There has been a remarkable improvement in application development techniques. 'Spaghetti' code development method is long gone, in which all focus used to be on solution for now. Traditional way of coding has given great importance to creation and calling of routines. Rewriting of code should be avoided as much as possible. It has given rise to object oriented programming in which each object represents a particular part of a certain project.

Different functionalities are fulfilled in the form of methods of respective objects and they are collaborated together to meet business requirement. [1]. Building a code which could be reusable for future developments is always a crucial task. Design pattern let us not to start a project from the scratch; it rather promotes the idea of finding similarity with previously faced design problems and then reusing the solution with necessary modifications and customized functionality.[2]

II. Factory Pattern

Factory pattern does not define concrete classes, in fact it tries to keep classes abstract and use similarity of objects on the course of creation and creates group of similar objects. Factory pattern plays role in the way objects are getting created, that is the reason it belongs to the group of creational pattern. Factory pattern uses interfaces for the creation of 'base objects' which could be used to create requirement specific concrete object.

III. Abstract Factory Pattern

Abstract factory provides scalability to the factory design pattern. As we know, application development is seldom a linear task. Most of the times, objects are created and called in a hierarchy and it too underlines a pattern in which they are created. Abstract factory is that factory object which creates other factories which would further create concrete objects.

IV. Proposed Work

IDE and code development frameworks have been using design patterns recently, since the concept was put forward by Erich Gama in 1994. [1] Disciplined coding and consistency in nomenclature of classes is heart of design patterns. Factory class will take input of class name and will return concrete object of the class.

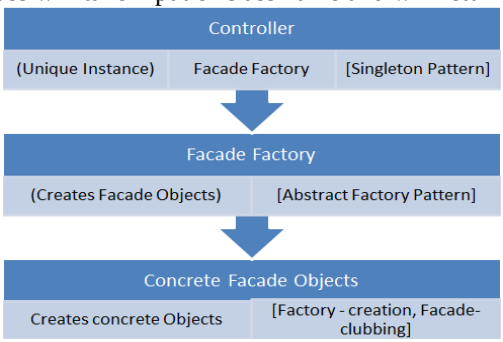

Fig 1. Suggested Model

In this paper, role of factory pattern in the model will be discussed.

V. Factory Pattern In The Model

Factory Pattern is mainly used here to create objects of concrete classes on the fly, without exposure of creational logic to the end user. Factory pattern can be seen as a classical object oriented design pattern which makes use of typical object oriented functionality like 'constructor' to create the objects. The most useful feature of this design pattern is use of 'common interface' to create different concrete object instances. It enables factory pattern to largely simplify object creational logic and proves out crucial in terms of code efficiency and line of code metric.

VI. Example And Pseudo Code

If we consider example of an engineering college, factory pattern can be used in hierarchical form where a common interface *stream* can create different branch objects.

Pseudo code :-

interface *Stream { //Basic Stream Logic.}*

AbstractFactory object is a factory of factories. For example, 'Stream' interface could be one of the interfaces used in the application.

There could be another interface say 'xyz' which could be created using AbstractFactory object with string input to object returning method. [3]

Psuedo Code :

public abstract class *AbstractFactory*
{
public abstract *Stream getStream(String stream);*
}

Using the interface stream, a *streamfactory* object is made which will then create concrete stream objects.

class StreamFactory { //use getStream method to get object of type Streampublic Stream getStream(String StreamType){
if StreamType = 'abc'
then
Return new abc();

In the proposed model Factory pattern will play role of creational design pattern in which AbstractFactory object will create factory objects as per requirement, these factory objects will then make use of common interfaces to create different concrete objects so as to have coherence in application development and naming consistency. [5]

VII. Conclusion

Code reusability should be one of the prime factors to be considered while developing an object oriented application. Design patterns promote code reusability by means of naming consistency and object creation method. Given model uses multiple design patterns together to propose a methodology of programming. It explains how factory pattern can be used to create a common interface for creation of objects and also, how they can be created dynamically using constructor input in object oriented language.

Acknowledgements

We acknowledge Mr. Akshay Mendki for his support and help in development of this research.

References

[1]. Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of reusable object oriented software Addison-Wesley, Second ISE reprint edition, 1999.
[2]. Rachel Cardell-Oliver, Evaluating the Application and Understanding of Elementary programming patterns 22nd Australian Conference on Software Engineering, pg 61-66, 2013.
[3]. Design Patterns, http://www.tutorialspoint.com/design_pattern, (Accessed: 28 Dec 2015)
[4]. Gaurav Pradip Pande ,Akshay Janardan Mendki, Design Patterns' Model for Application Development in Object Oriented Languages
[5]. International Journal Of Engineering And Computer Science Volume 4 Issue 12 Dec 2015 , Page No. 15114-15116
[6]. Freeman, Eric T, Elisabeth Robson, Bert Bates, Kathy Sierra (2004). Head First Design Pattern, Second Edition, Oct 2004.