ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (23-28), Month: April - June 2015, Available at: www.researchpublish.com

Design Patterns and Its Evolution

[1]PRAGNA CHOWDARY KAJA, [2]TARIK ELTAEIB

***Abstract*: Design patterns in daily usage present a decently shaped dialect of a software design. In this paper we present the different design patterns, the outline designs which are improved and having reusable answers for the issues confronted by every developer in coding. The significance of design patterns should be understood by both the end user and programmer. We also discuss about the evolution of the different patterns and also the relationship between the mainly used patterns. We also present the design patterns with an example which helps us to understand the patterns more easily.**

***Keywords:* Design patterns, Programmer, Relationship, Adaptor Class, Chain of Responsibility, Evolution, Applicability, Object Oriented Programming, and Model.**

I. INTRODUCTION

Design pattern development is intriguing subject. Yet, it would be a basic assignment in programming advancement since adjustment to quick change of necessities and working situations is the aggressive edge in today's business atmosphere. In this article, we propose a strategy for taking care of complex issues with developmental examples of plan in terms of configuration examples.

The arrangement acknowledged generally for repeating configuration issue is "Design Patterns" which speaks to how to structure the classes to get the final solution. It controls the programmer orderly how to actualize the code in simple route by concentrating on relationship among the classes. Then again, it doesn't clarify us with the particular algorithms.

This research paper will explain about three main categories of design patterns such as structural, creational and behavioral and also in-depth concepts of Factory method, Adapter and Chain of Responsibility. The every pattern has the diverse utilization while resolving the problem as per the given prerequisite. This examination paper serves to discover the principle functionalities, structures and the execution methodology of different designs patterns.

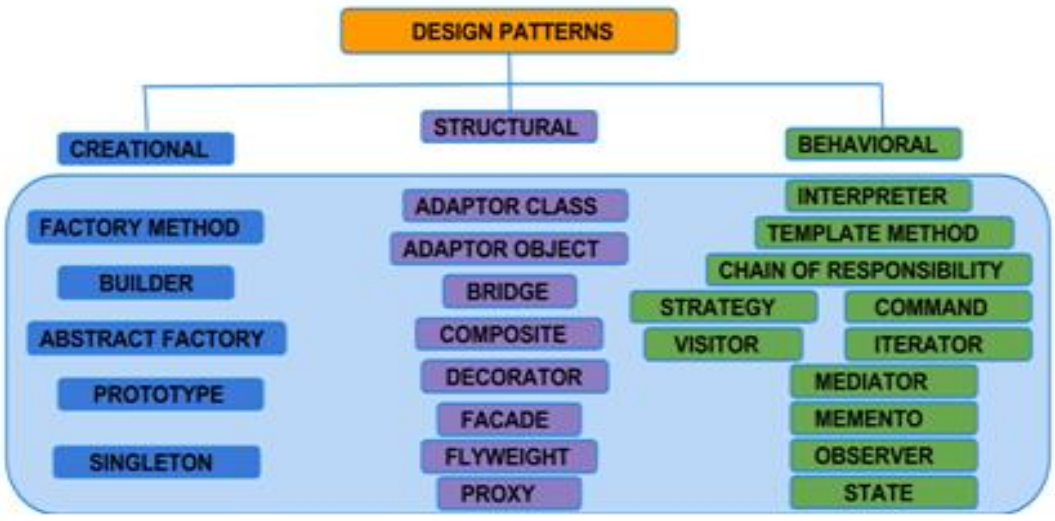

Fig: Classification of design patterns

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (23-28), Month: April - June 2015, Available at: www.researchpublish.com

II. EVOLUTION OF DESIGN PATTERNS

Design evolution is a sorted out aggregate changes to design. Since design patterns speak to plan, we can see the relationship between two configuration designs as a design advancement. To speak to the development relationship among the configuration patterns, we have to speak to heading of advancement and structure of aggregate changes. Therefore, we utilize a coordinated chart to speak to the heading of advancement, and a requested arrangement of PEOs to speak to the struc

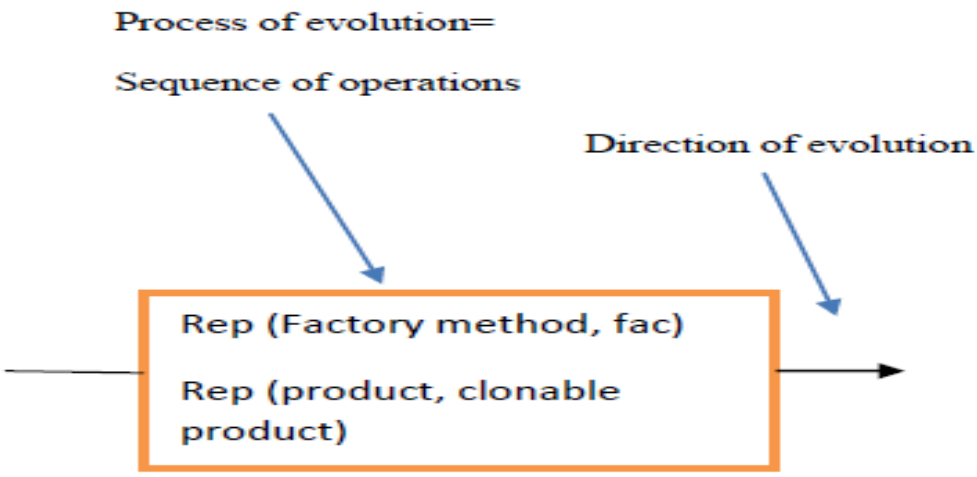

Fig: Pattern Evolution Concept

III. MODEL OF DESIGN PATTERNS

All the design patterns depend on the way of representation. Any patterns is represented by using the UML diagram which is Unified Modeling Language which can be more easily understood and all the diagrams contain a name, problem, pattern architecture and protocol

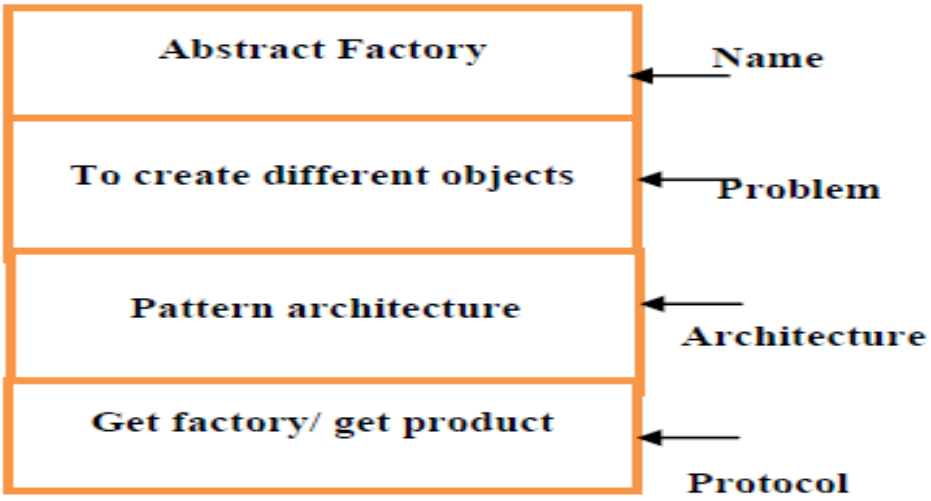

Fig: Representation of a pattern

Creational Design Patterns:

Creational design patterns are design patterns that deal with object creation mechanisms, trying to create objects in a manner suitable to the situation.

As per the necessity, objects will be made by utilizing object creational mechanisms. On account of the new instance creation systems the creational outline examples could undoubtedly take care of the many-sided complexity related issues.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (23-28), Month: April - June 2015, Available at: www.researchpublish.com

The most helpful example fits in with this classification is "Factory method" which permits a class to concede instantiation to subclasses.[2]

Factory method is creating an object without uncovering the creational logic to the clients and just referring to the object that is created using common interface. This makes the design more customizable. [3]

Representation of Factory Method:

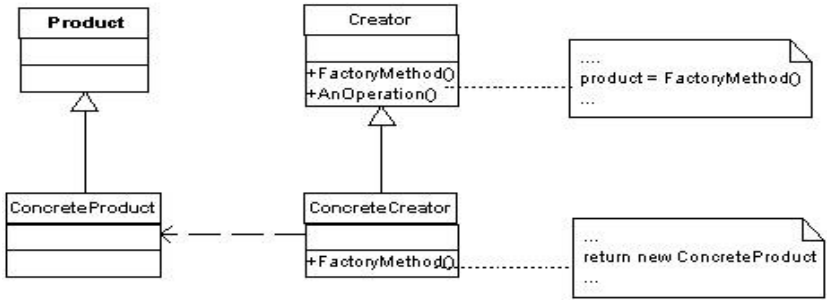

Fig: Factory Method

UML Representation of Factory method:

Here we take an example to explain the Factory method with the help of an example 'representation of shapes of circle, rectangle and square'. This example will explain the factory method where interface is defined by creating an object and the sub classes in the problem will decide which one to instantiate. This is represented in UML diagram

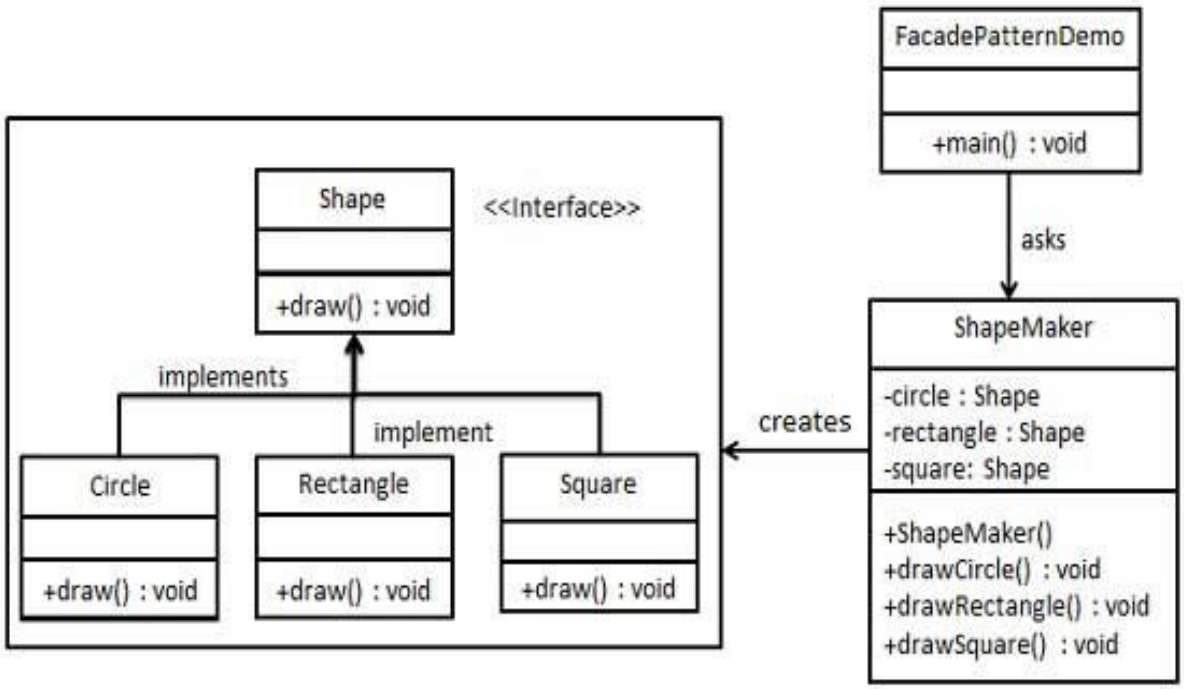

Fig: UML Representation of Factory Method

Structural Design Patterns:

Structural Design Patterns are Design Patterns that ease the design by identifying a simple way to realize relationships between entities. The most famous and useful pattern in structural design pattern is adaptor pattern

In general terms **Adaptor** can be explained as a problem of inserting a new three point electrical plug to the two point wall outlet which can be done by using an adapter or some intermediate device. Adaptor is about creating an intermediate abstraction that maps the old components to new system which is implemented using inheritance or aggregation.[5]

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (23-28), Month: April - June 2015, Available at: www.researchpublish.com

Adaptor Structure:

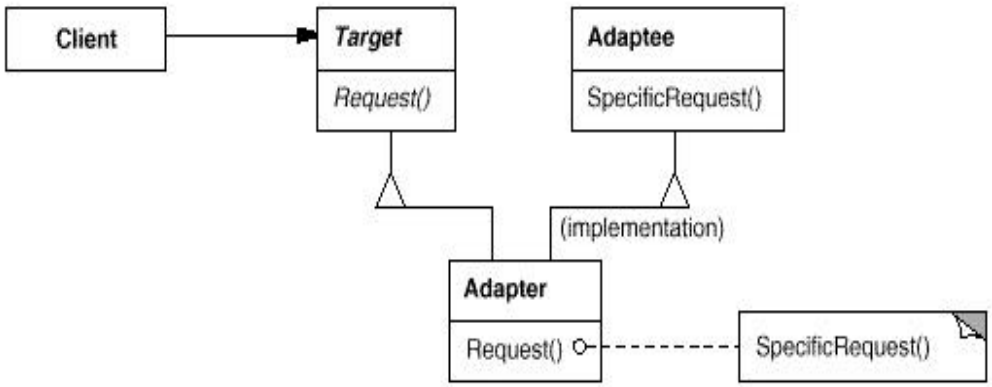

Fig: Adaptor Method

UML Representation of Adaptor method:

Here we take an example to explain the Adaptor method with the help of an UML representation.

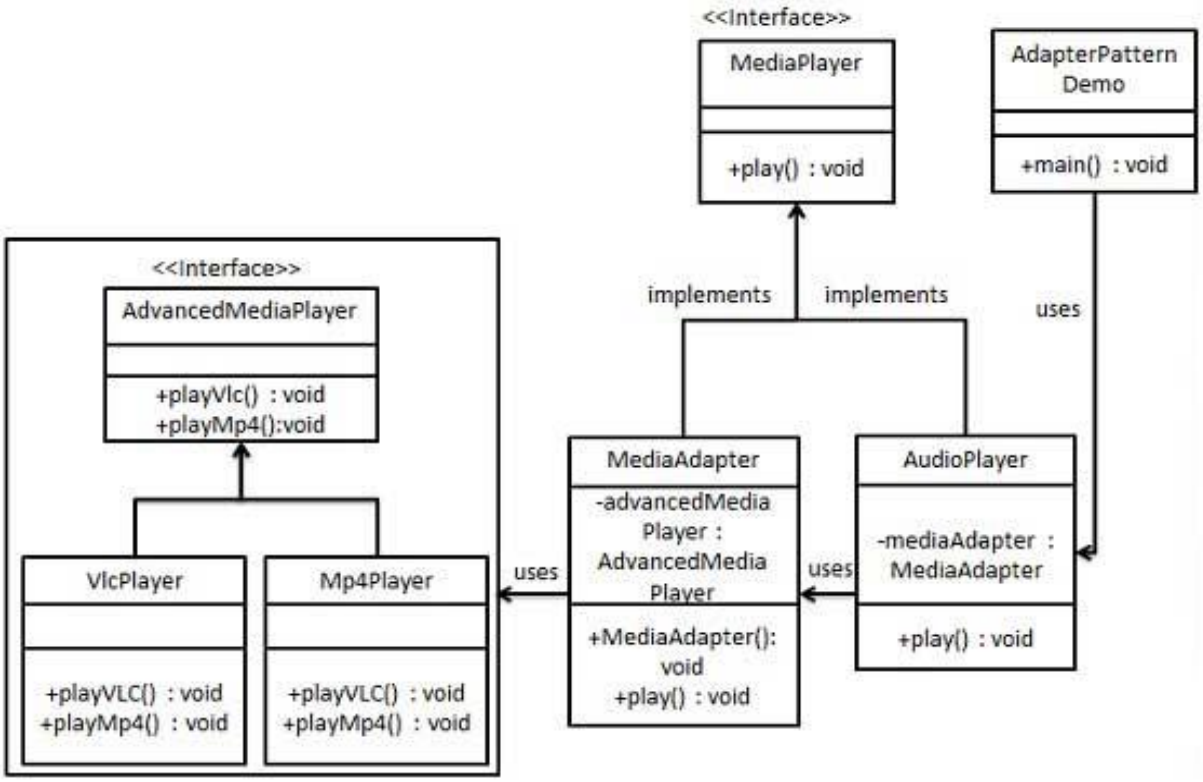

Fig: UML Representation of Adaptor Method

Behavioral Design Pattern:

Behavioral design patterns are configuration designs that distinguish normal correspondence designs in the middle of objects and these examples expand flexibility in communication

Chain of responsibility is Encapsulate the processing elements inside a pipeline abstraction and have clients leave their requests at the entrance to the pipeline. This will simplify the object interconnections.[6]

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (23-28), Month: April - June 2015, Available at: www.researchpublish.com

Chain of Responsibility Structure:

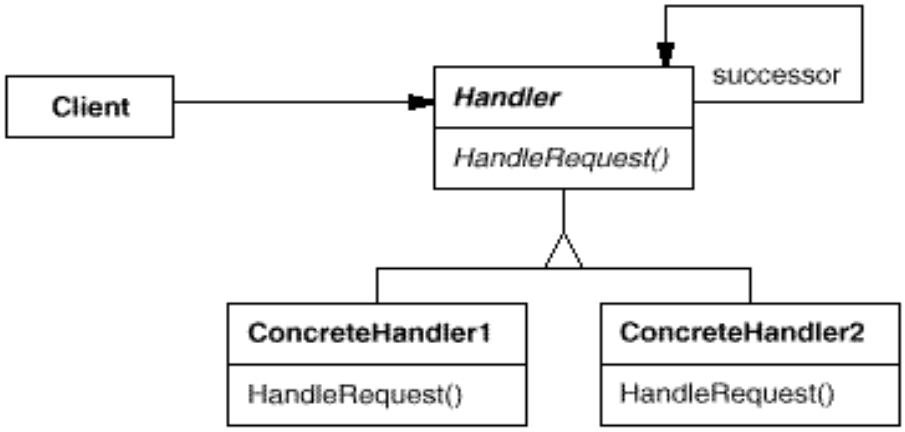

Fig: Chain of Responsibility

UML Representation for the Chain of responsibility:

Here we take an example to explain the chain of responsibility with the help of an UML representation.

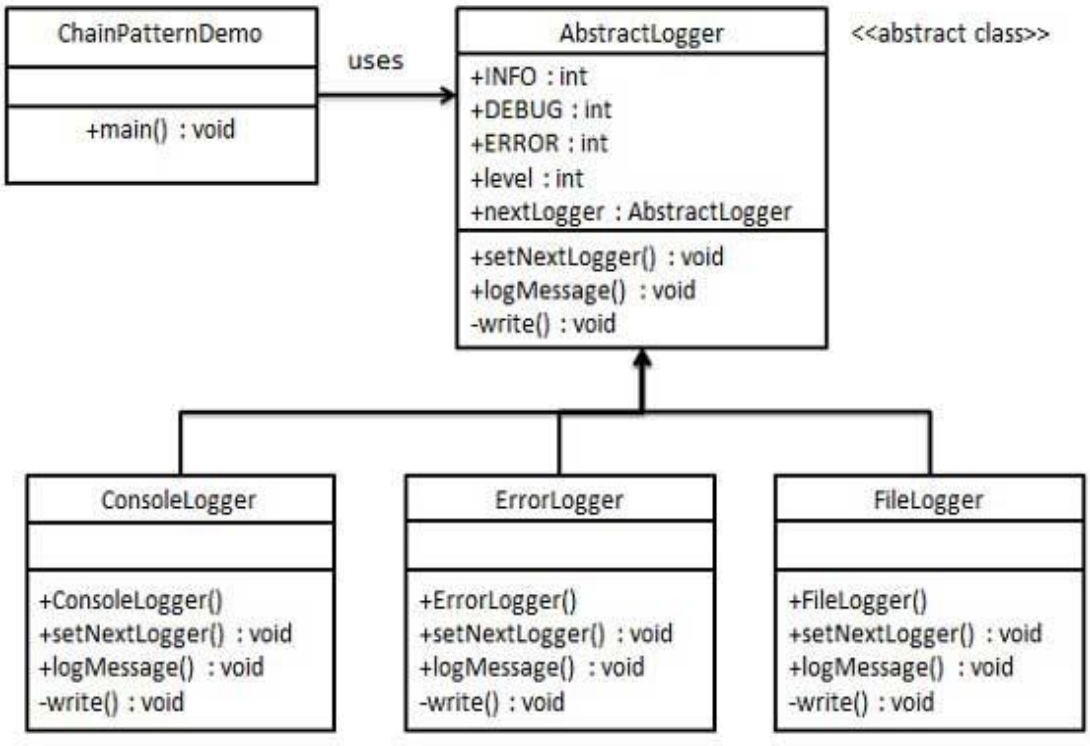

Fig: UML Representation of chain of responsibility

IV. CONCLUSION

Bringing design patterns designs into the programming scene permits both the developers and the clients more agreeable and adaptable. For making new activities as well as aides in support extends in a wide range. It permits the code designers to see vastly improved by evading the procedure of executing every last step. Error detection and Error Correction which set an extreme part for any undertaking can be explained with more precision and less compass of time by utilizing the outline designs particularly as a part of object oriented programming.

REFERENCES

[1] Aoyama, M. Evolutionary patterns of design and design patterns. in Principles of Software Evolution, 2000. Proceedings. International Symposium on. 2000.

[2] Qiu, W., W. Zou, and Y. Sun. Design Patterns Applied in Power System Analysis Software Package. in Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on. 2012.

[3] Phek Lan, T., et al. Improving a web application using design patterns: A case study. in Information Technology (ITSim), 2010 International Symposium in. 2010.

[4] Palma, F., et al. Recommendation system for design patterns in software development: An DPR overview. in Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop on. 2012.

[5] Zhang, H. and S. Liu. Java Source Code Static Check Eclipse Plug-In Based on Common Design Pattern. in Software Engineering (WCSE), 2013 Fourth World Congress on. 2013.

[6] Aminzadeh, N. and S.S. Salim. Detecting and visualizing web design patterns. in Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on. 2010.