Research Paper on Object Oriented Software Engineering

[1]Iqbaldeep Kaur, [2]Navneet Kaur, [3]Amandeep Ummat, [4]Jaspreet Kaur, [5]Navjot Kaur
[1,2,3,4,5]Dept. of CSE, Chandigarh Engineering College, Landran, Punjab, India

Abstract
By the development of the software industry and the advances of the software engineering, the use of Object Oriented Software Engineering (OOSE) has increased in the software complex real world. The origin of the OOSE in evaluation and design of the software has expanded much and is now considered as one of the software integration processes. The OOSE is combination of Object Oriented Analysis (OOA) models, Object Oriented Design (OOD) and the Object Oriented Programming (OOP) which provide a powerful way for development of the software. The OOSE provides the possibility of OOP on the development and production of the software after the analysis and designing the software. In the paper, we study the general terms and issues which effects software development in industry.

Keywords
Software Development, Object Oriented Software Engineering, Object Oriented Analysis, Object Oriented Design, Object Oriented Programming.

I. Introduction

The concept of object oriented programming comes from the programming languages.The first programming language Simula that come with the feature of object oriented paradigm and developed into the language Simula 67. It was introduced in 1960's. In the 1970's this language then used as a platform for development of programming language Smalltalk. In the 1980s the C++ was introduces which put the concept of Simula into it. In the 1990s the object oriented programming languages become famous and complex enough through which it used to handle large complexities of interacting components. After this the object orientation also applied to object oriented design (OOD) and object oriented analysis (OOA) of software system. The languages use the feature to combine methods and data into single unit. Object oriented used to solve the real world problems.

Fundamentals of OO are often explained in terms of features provided by OOP languages. It helps us to design higher level of abstraction. It helps to perform manipulation by the programmer into the system. It is acceptable for designing and implementing software system in area range from client server to real time. Procedural approaches were not best fit to adapt new requirement and does not contain the potential of reuse. Most of the performance failure is due to the issues in the development process and in the architecture design phase. It becomes engineer to check the performance goal effectively.

II. Object Oriented Software Engineering

By the development of the software industry, the OOSE tries to create the trends which make the development of the software fast and with low costs. So, the OOSE could be described as combination of activities and object oriented methods on development of the software. OOSE makes a structure in which the methods,processes and tools are combined for development of software.

The key point in OOSE is the design and analysis phases which play the role and the relation dependency. The object oriented programs generally include objects which use classes to get in relationship. So, the object oriented programming software engineering relies on theobjects.

The object oriented models is a set of the activities for development of the software and they used develop any activity according to a set of the goals. So, OOSE leads to reuse of the software and enters the integration capability into the software. The integration process is a struggle to reach the best specifications of the models of the software process. The most important key role of the OOP is the reuse of the codes in software development process. The OOP for reuse will lead to reduction of the costs and the reduction of coding of the software. All methods and member functions are defined in description of a class. Now a daysmost of the software are object oriented because they make software development easier and fast.

Object oriented software engineering play key role in development of software, because the object oriented features are very effective in program structure, program sectioning and its complexity. Also, the Component Based Development is more advantageous in increasing the reusability capability, reduction of costs and the time of production and is a very important point in software development.

III. The Object Oriented Software Engineering Models

In OOSE, the software systems develop by models which use requirements, designing and OOPs. So, OOSE is composed of a set of models which create a framework for software development. The fast development in software engineering and the changes of technology in this field are the causes which lead to use of OOSE as a framework for integrity and development.

A. Mapping

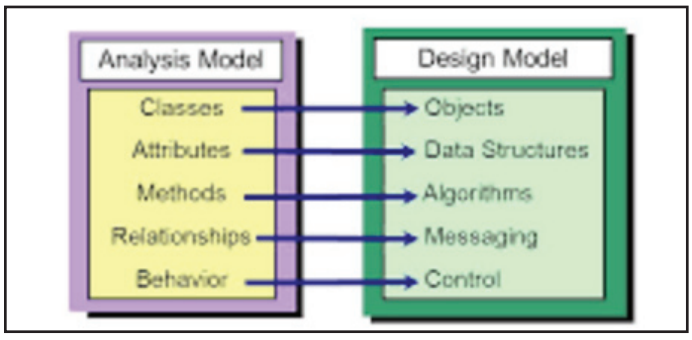

Fig. 1: Mapping View of Analysis and Design Model

A. Object Oriented Analysis

The analysis provides the description and checkpoint in software development because software development takes look to the findings of the analysis. The OOA consists set of software engineering activities to the software development requirement .When software point view is consider, the OOA finds the software requirements in software development activities. The OOA process the main requirements of the users, studies the feasibilities, validates the specifications and manages the requirement. The OOA identify all classes and the relationship between the classes and their behavior. The goal of OOA is development of a model which describes the requirements of the users.. So, OOA in OOSE

makes the main activities, the classes and their relationship, the system behavior and the data in relationship to the classes become definite.

B. Object Oriented Design

The main goal of OOSE is the efficiency, reliability, reusability and sharing of resources in software systems. Hence, increase the reusability factor in software development. The OOD is focused on organizing the objects in the classes and all methods and the functions in a class are defined in OOD phase. It is the next step to object oriented analysis. OOD is a process in which the user requirements are transformed to a design for software creation. In OOD all requirements in an analysis model must be as the requirements of the users. OOD transforms the data objects to the data classes which perform the software development. So, OOD leads to a structure in which the class implementation takes place after requirements analysis. OOD in OOSE includes two main factors.

- Software should have no errors violating the operation of it.
- The relations between the data objects and other information related to each other must be defined.

C. Object Oriented Programming

It is basic techniques of coding is OOP. It is done by some programming languages which has the features and capabilities of OOPS. It is the implementation phase. So, when software is implemented by OOP, it is possible to say that software includes a set of commands and in relation classes, and if the classes are organized, reusable codes are created which reduce the time and the costs of production. So, the complexity of the operations is reduced and the large software could be manages better. The use of OOP makes the producers hide the complexity of the software systems in classes. The OOP models the relationship between the objects of the classes. And the objects used to send and receive message by the message passing feature it contains. The OOP uses the data and the functions and provides the reusability and creation make software development takes place fast. OOP model uses some features for software development systems. Three main features and instance of OOP are:-

1. Objects:

An object has the following characteristics.

- state

For recording the history of an object

- Behaviour

The observable effects based on its state and the relations with other objects.

- Identity

As known by other objects, either by name or by reference.

- Relations

Relations between objects are expressed by interactions in the form of message passing.

2. Encapsulation:

In OOP classes encapsulates the data along with member function (methods) for the purpose of hiding. So, encapsulation organizes the data and the methods better. Also, encapsulation is often called hiding information. Encapsulation creates limits in accessing the internal data of the classes. So, encapsulation creates a packet which protects the internal instances from the users. Capsulation is the completeness and integrity that shows the relation of the data and the methods of the classes in integration. Figure (2) shows encapsulation modeling.

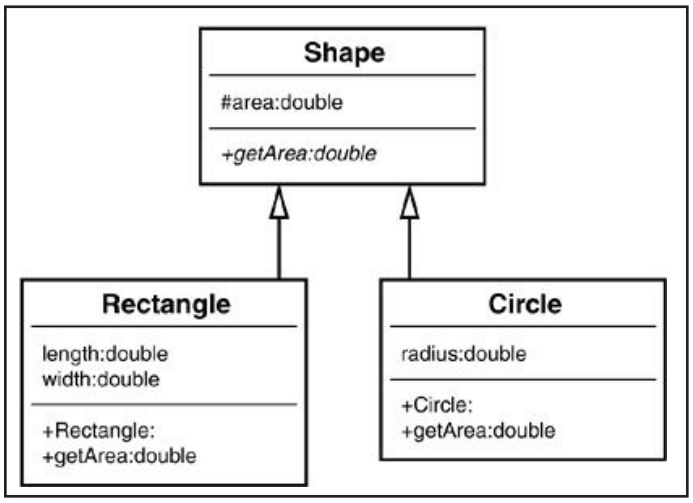

Fig. 2: Encapsulation of Data and Methods

As you can see in fig. 2, using the encapsulation, it is possible to packet all variables and the functions in classes.

3. Polymorphism

Using polymorphism in OOP it is possible to define more than one same name functions with different structure. In other word polymorphism term means one name different forms. Polymorphism makes a function to get implemented in different ways in classes and sub classes and get different forms. Polymorphism creates a common interface for different implementation of a function for the programmer which operates different for different objects. Polymorphism means "one class, some functions with different structure.

4. Inheritance

Inheritance is an operation in which an object can get traits from another object. So, using the inheritance we can inherit traits which are needed from the super class and inherit down to the base class. There can be chances where properties are inherited from more than two classes. SO, if changes take place in data and the functions of the inherited classes, they will be operated in other inheriting classes. It is the power full feature of object oriented language like java. Reusability of the codes is one of the main advantages of inheritance.

5. Information Hiding

Hiding of information about an object can be done by deliberately making this information inaccessible. Objects composed in this way are vertical related with the objects they are composed of.

IV. Issues With Object-Orientation

We describe some of important issues related to OO that lead to a lot of confusion and problems in this area. These are described below:-

A. Modelling Versus Reuse

During the last 10 years of change our focus is toward the object-oriented community. From being mainly interested in reuse of code, the focus has shifted to analysis and design. We argue that there should be a balance between these extremes to fully realize the benefits of the potential for object orientation to integrate analysis, design and implementation.

B. Explicit Conceptual Framework

The conceptual framework underlying object oriented modelling should be used independently of the languages. This separation is necessary to avoid limiting the developer by the expressive power of the languages and to produce new requirements for language design.

C. Abstraction Mechanisms

Although there is some procedure about the core abstraction mechanisms, there are still a number of differences between the common languages. We discuss alternative mechanisms for supporting classification and composition, single and multiple inheritance, inner versus super, generosity versus virtual classes, encapsulation, dynamic versus static typing and types versus classes.

D. Class-based Versus Prototype-Based Languages

Prototype-based languages are one of the most interesting developments within object-orientation. Prototype-based languages makes possible to program without classes, in general they do not support programming using a class-like style. We argue that the class-like style should be supported.

E. Concurrency

Simula67 uses a notation of quasi-parallel processes in the form of active objects that are used for representing concurrent processes. They have also been the basis for the design of concurrency in BETA. No other object-oriented language seems to have adapted the Simula67 notation of active object. There are a number of proposals for concurrency in object-oriented languages, but no single model has been widely accepted. We argue that the Simula67 approach is considered a good approach to concurrency.

F. Features

When people are questioned with the fundamentals of OO, they reply with a list of features provided by OOP languages instead of what OO is truly about. The features mentioned mostly are inheritance, polymorphism, encapsulation, and abstraction. For OO these features are irrelevant, apart from abstraction but the term is wrongly used here. To describe OO in terms of features provided by OOP languages that support OO leads to the conclusion that for a programming language to be OO, it has to support these features. This circular reasoning is certainly not helpful for a good understanding of what OO is truly about.

G. Abstraction and Generalization

There is a lot of confusion over abstraction and generalization, or rather they are interchanged. But Abstraction and generalization are not the same. With abstraction some detail is left out that is considered not important in a description on a higher level of abstraction. With generalization that detail is not left out, but described in a general way on the same level of abstraction.

V. Conclusion

Software engineering has the robust representation to measure the real world entities with the help of object oriented software engineering. It emphasizes on the reuse and complex development of software product with the help of object oriented analysis, object oriented design and object oriented programming. With the feature like objects, encapsulations, polymorphism, inheritance, information hiding and abstraction it can be seen that production of software is better by using these capabilities.

References

[1] Martin, James, James J. Odell.,"Object-oriented methods", Prentice hall PTR, 1994.

[2] Diertens, Bob.,"On Object-Orientation," arXiv preprint arXiv:1010.3100(2010).

[3] Soleimanian, Farhad, Saman Jodati Gourabi, Isa Maleki. "Object Oriented Software Engineering Models in Software Industry."

[4] Aksit, Mehmet, Louis Marie Johannes Bergmans, "Obstacles in object-oriented software development," ACM Sigplan Notices 27.10, pp. 341-358, 1992.

[5] Madsen, Ole Lehrmann,"Open issues in object-oriented programming—A scandinavian perspective." Software: Practice and Experience 25.S4 (1995): 3-43.

[6] Amit Verma et al.,"Mobile Agent and IP: Hurdles and Protecting Agent", National Conference Cum Workshop on Information Security and Networks (ISAN-2009), pp. 188-191, Held at CIET-RAJPURA-PUNJAB on 19th -20th, June 2009.

[7] Amit Verma et al.,"Artificial Neural Network Technology: A Technology Revolution", National Conference on Recent Trends in Communication and Broad casting held at SUSCET-TANGORI (MOHALI) - PUNJAB, Vol. 1, pp. 14-20, April, 2009. (Sponsored by The Institute of Electronics and Telecommunication Engineering, IETE).

[8] Amit Verma, Navdeep Kaur Gill,"Image Processing and Watermark", International Journal of Computer Science and Technology (IJCST), Vol. 7, Issue 1, pp. 143-147, Jan – March 2016.

[9] Amit Verma, Navdeep Kaur Gill, "Analysis of Watermarking Techniques", International Journal of Computer Science and Technology (IJCST), Vol. 7, Issue 1, pp. 153-156, Jan – March 2016.