International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 8, Issue 6, June 2019)

Object Oriented Application Paradigms and Software Development Processes

Girma Yohannis Bade[1], Fetenech Meskele[2]
[1,2]*Department of Computer Science, Wolaita Sodo University, Wolaita, Ethiopia*

***Abstracts*-- 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 Object Oriented Software Development is a set of the Object Oriented Analysis (OOA) models, Object Oriented Design (OOD) and the Object Oriented Programming (OOP) which provide powerful framework for development of the software. The main goal of the OOSE is the increasing the reusability capacity of the software, reducing the complexity and the software systems maintenance costs. This paper discuss Object oriented application paradigms and software development process. The application paradigms are specification, analysis and design, programming, and user interfaces which also consists software developmental processes like requirement specification, system analysis, design, implementation, deployment and maintenances. In both case class and object play a central role to ensure that software development is object oriented.**

***Key terms:* Application paradigm, software, object oriented, software development process**

I. INTRODUCTION

Late in three decades the creation and application of the software has progressed a lot. Now, the software industry is so widely extended that the structured methods can no more analyze and design the complex software systems. The object oriented models make us defeat the software design and analysis in short time. OOSE is a relatively new method for designing and implementing the software systems. The main goal of the OOSE is the increasing the reusability capacity of the software, reducing the complexity and the software systems maintenance costs[1]. Object oriented was first suggested for the development of the software in 1960 and developers of the software focused on object oriented in 1980s, and the method got to be used vastly in the software society [7].

II. METHODOLOGY

A. Object oriented thinking

An object

An object is an abstraction of something which exists in the real world or in our minds which belongs to the system we want to model and which we want to store information about.

Classes

A class represents an abstraction of a class of objects in the real world; it is the set of all Class objects sharing certain common properties. And in fact a class is a type in most programming languages. Once a class has been abstracted from the real world, the attributes and methods belonging to it are identified. Method is a common name for a procedure which belongs to an object, another Method name popular with the C++ community is member function.

B. key features of the object oriented approach

Encapsulation

The data is grouped together with the procedures working on them. Ideally, there is only an interface through which the data can be accessed comprising several procedures, but no direct access to the data. This is called data hiding. The procedures should not reveal the implementation used to manipulate the data. The whole concept is called information hiding.

Inheritance

Inheritance means that one class stands lower in a hierarchy than another, having all the attributes and methods of the class standing higher, thereby inheriting them. The class that stands higher is called the superclass or the base class, whilst the class inheriting is called subclass or derived class. The two classes stand in a is-a relationship to each other, i. e. we say the subclass is a (kind of the) super class.

Polymorphism

Polymorphism is supported when we can use one class where another is expected. This is sensible when the class which is actually provided has at least the one feature which is used in the context that the expected class did have. Polymorphism is achieved when it is possible to use a subclass of a class instead of that Polymorphism class itself.

III. APPLICATION PARADIGM OF OBJECT ORIENTED SOFTWARE DEVELOPMENT

A. Programming

In the late 1960s the first OOPL was developed: Simula 67.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 8, Issue 6, June 2019)

An OOPL supports all the key concepts enumerated in the previous section. If a language does not support them, it is not object-oriented[2].

B. Analysis and design

Even later than object-oriented programming object-oriented analysis and design were developed. Whilst object-oriented programming is accepted and widely used, analysis is often carried out in more conventional ways.

C. Specification

Object-orientation is used to master the complexity, to break the problem into smaller units and to tackle them one after another.

D. User interfaces

One of the things that has most often been claimed to be object-oriented is the user interface. Most are not even aware of what object-orientation means. But object-oriented user interfaces can, in fact, make sense.

IV. Object Oriented Software Development Processes

Requirement specification

A formal process that seeks to understand the problem and document in detail what the software system needs to do is known as requirement specification. This phase involves close interaction between users and designers. Most of the examples in this paper are simple, and their requirements are clearly stated. In the real world, however, problems are not well defined. You need to study a problem carefully to identify its requirements.

System analysis

It Seeks to analyze the business process in terms of data flow, and to identify the system's input and output. Part of the analysis entails modeling the system's behavior. The model is intended to capture the essential elements of the system and to define services to the system.

System design

The process of designing the system's components. This phase involves the use of many levels of abstraction to decompose the problem into manageable components, identify classes and interfaces, and establish relationships among the classes and interfaces [3].

Implementation

The process of translating the system design into programs. Separate programs are written for each component and put to work together. This phase requires the use of a programming language like Java. The implementation involves coding, testing, and debugging.

Testing:-Ensures that the code meets the requirements specification and weeds out bugs. An independent team of software engineers not involved in the design and implementation of the project usually conducts such testing.

Deployment

Deployment makes the project available for use. For a Java applet, this means installing it on a Web server; for a Java application, installing it on the client's computer.

Maintenance

Maintenance is concerned with changing and improving the product. A software product must continue to perform and improve in a changing environment. This requires periodic upgrades of the product to fix newly discovered bugs and incorporate changes.

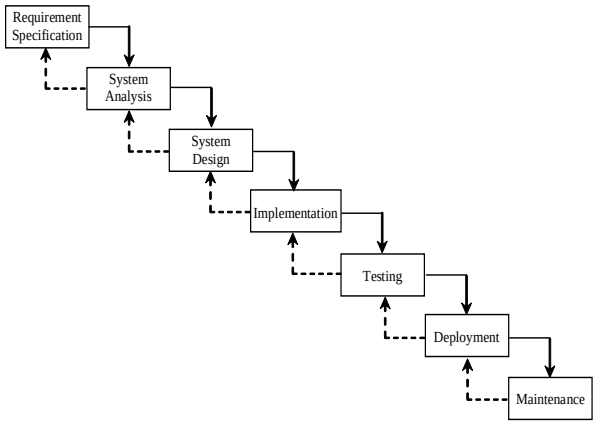

Figure 1: Iterative based OOSD processes [5]

V. Conclusion

The object-oriented approach is nearer to the systems it has to model. The structure of the problem domain can be directly represented. This has the advantage of increased understanding and is therefore less error-prone. Due to the strong encapsulation the object oriented approach proposes, software reuse is facilitated. Reuse means to use something again which has been used before. The simplest form is to 'copy &paste' code fragments from other programs. Its simplicity at the beginner level, enabling students of programming to easily grasp the concepts; modularity for objects, providing software programmers with the opportunity to code without too much trial and error; and efficiency[4]. The disadvantages include more complicated programming for back-end development; the programming is less efficient when not used in conjunction with procedural programming, and the fact that the coding itself can be repetitive and cumbersome.

Software engineers still have to learn a new, and likewise managers face considerable problems since the methodology used is often not suitable for a strictly managed process [6].

REFERENCES

[1] Michaelpiefel, coursework 'information engineering'1996/97
[2] Systems Analysis and Design Kendall and Kendall Fifth Edition
[3] An Introduction to Object-Oriented Analysis and Design and Iterative Development by C. Larman. 3rd edition. Prentice Hall/Pearson, 2005.
[4] https://www.bestcomputersciencedegrees.com/faq/what-is-object-oriented-software-development/
[5] https://cs.nyu.edu/courses/spring07/V22.0101-002/11
[6] Bernd Bruegge, Allen H. Dutoit Object-Oriented Software Engineering: Using UML, Patterns and Java, 3rd Edition
[7] https://www.academia.edu/9790955/Object_Oriented_Software_Engineering_Models_in_Software_Industry