International Journal of Research
Available at https://edupediapublications.org/journals

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue-01
January 2018

Comparison Between Traditional and Object -Oriented Approach in Software Engineering

Anju Bala

Research Scholar, Deptt. of Computer Science and Applications, M.D. University, Rohtak, Haryana

Abstract

Testing is one of the most important and challenging part of software development life cycle. It takes a lot of time and cost of total software development life cycle but unavoidable, executing this phase with the intent to find errors or fault to make a software error free and satisfies all the requirement of end users. This paper highlights the comparison between Traditional approaches and Object-Oriented approach to software testing. Traditional approach has a lot of model, is used to deal with different type of projects - such as waterfall, prototyping, spiral, iterative and v-shaped, however, there is dilemma as how best fit for object oriented approaches i.e. is widely used. In recent years, most of the developers surge their interest in object-oriented methodology. A lots of researches have been made in the field of software testing are carried out in academic as well as in industry. There is a noticeable differences between conventional software development process to object oriented process as well as conventional testing and object oriented testing with testing process is same for all software development paradigms, the approach of testing may differ widely. The main objective of this paper is to overview on comparison between traditional and object oriented approach.

Keywords- Software Testing, Traditional Approach, Object-Oriented Approach, Comparison between Traditional Approach and Object-Oriented Approach.

1. INTRODUCTION

The main goal of software testing is to find errors in a systematic and effective way rather that random experimentation. All software, especially large pieces of software produced by many people, should be produced using some kind of methodology. Even small pieces of software developed by one person can be improved by keeping a methodology in mind. A methodology is a systematic way of doing things.

Testing may be defined as the process of validating and verifying the software product. Verification is the process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase i.e. process oriented whereas validation is the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements i.e. product oriented.
Executing the program using the generated test cases, and Evaluating the test result

A good methodology will address at least the following issues: Planning, Scheduling, Resourcing, Workflows, Activities, Roles, Artifacts, Education.

With the traditional approach, you‟re expected to move forward gracefully from one phase to the other. With the modern approach, on the other hand, you‟re allowed to flow freely around the system, encountered procedural approach.

2. TESTING APPROACH

Here, explanation of these two approaches as follow:

2.1 TRADITIONAL APPROACH

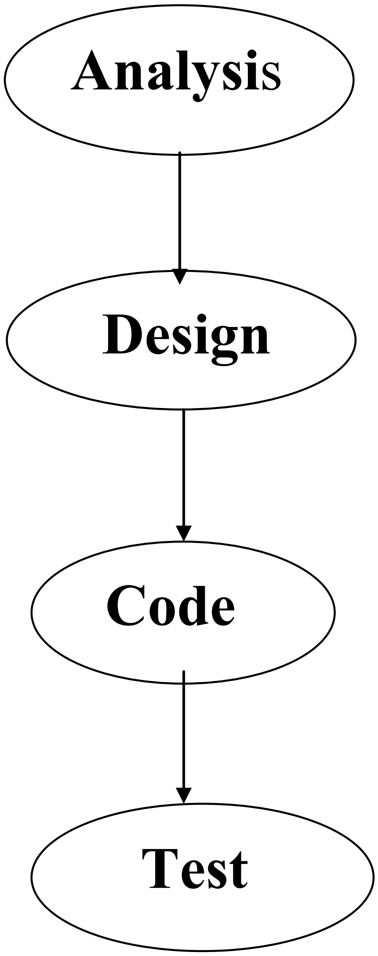
There are a number of phases common to every development, regardless of methodology, starting with requirements capture and ending with maintenance. With the traditional approach, it emphasis on procedure and top down strategy to solve the complexity of things and intented to move forward gracefully from one phase to the other.

Requirements: A system requirement modeling (or functional specification) means deciding what capabilities the new software will have and writing down those capabilities.

Analysis: Analysis means understanding what are dealing with. Before designing a solution, it needs to be clear about the relevant entities, their properties and their interrelationships.

Design: In the design phase, will work out, how to solve the problem. System design breaks the system down into logical subsystems (processes) and physical subsystems (computers and networks)

Specification: Specification is an often-ignored, or at least often-neglected, phase. The term specification is used in different ways by different developers. For example, the output of the requirements phase is a specification of what the system must be able to do



Implementation: The type of the task is carried out during the implementation phase is Write the method or code for the Inventory class, in such a way that they conform to their specification.

Testing: When the software is complete, it must be tested against the system requirements to see if it fits the original goals. It is a good idea for programmers to perform small tests as they go along, to improve the quality of the code that they deliver.

Deployment

In the deployment phase, providing the ready to use software to end user with all documentation or manual with training material.

Figure 1: The Linear Sequential Model

2.2 OBJECT-ORIENTED APPROACH

Object-oriented approach was developed as the next great advance of software engineering to eliminate the flaws of procedural approach.. Software developed using object-oriented concept has a different structure and behavior , it revolve around the objects. Object-oriented approach is a data centric approach rather than algorithmic and it is a method based on hierarchy of classes and well-defined and cooperating objects.

In object-oriented approach, a system is viewed as a set of objects. All object-orientation experts agree that a good methodology is essential for software development, especially when working in teams. Thus, quite a few methodologies have been invented over the last decade.

Everything in object-oriented programming is grouped as self sustainable objects, thereby gaining reusability by means of four important object-oriented features makes it very popular such as, Abstraction, Inheritance, Encapsulation and Polymorphism. Encapsulation may be defined as wrapping up of data and functions into a single unit. Objects are described as implementations of abstract data types (ADTs). Usually an ADT definition is called class, while an object is a runtime entity i.e. instance of a class. Inheritance is the process by which objects of one class(child class) inherit the properties of another class(parent class). Polymorphism means more than one form enables a number of different operations to have the same name i.e. program entities should be allowed to refer to objects of more than one class, when a hierarchical relationship among these classes exits

The advantages of Object-Oriented Development are less time and cost reduced, greater product flexibility, and schedule predictability and the risks of them are performance and start-up costs.

Object-Oriented Unit Testing

The primary aim of unit testing is to uncover errors within a given unit. In the context of object-orientation, the smallest unit may be a method or a class. Testing a method which is considered as a single operation of a class is termed as intra-method testing and testing the integrity of the class as a whole is termed as intra-class testing. A class is a combination of data members and member functions. Testing a unit may involve more than one class because a class can contain a number of different operations and a particular operation may exist as part of a number of different classes.

Object-Oriented Integration Testing

Integration testing is performed to combine all modules and units to ensure all units of system work together or not, verifying the interaction of class, are minimized. class interaction, polymorphic calls and exception handling is termed as integration testing in object-oriented concept.

There are two important strategies for integrated testing of object-oriented system such as Thread based testing and Use based testing. Thread-based testing, combine all the sets of classes necessary to respond to one input or event for the system. Thread is tested and integrating individually. Regression testing is applied to ensure that no side effects occur.

Use-based testing, construction of the system by testing those classes (called independent classes) that use very few of server classes.

A. Analysis: The aim of the analysis process is to analyze, specify, and define the system which is to be design. In this phase, we build models that will make it easier for us to understand the system. The models that are developed during analysis are oriented fully to the application and not the implementation environment; they are "essential" models that are independent of such things as programming language, operating system DBMS, processor distribution, or hardware configuration etc.

Two different models are developed in analysis; the Requirements Model and the Analysis Model. The first model, the Requirements Model, should make it to meet the expectation of the customer and delimit the system and to define what functionality should take place within it fulfill.

The Analysis Model is an architectural model used for analysis of robustness. It gives a conceptual view of the system, consisting of various object classes: domain entities, active controllers, and interface objects. The purpose of this model is to find a robust and extensible structure for the system as a base for construction.

B. Construction

The construction process lasts until the coding is completed and the included units have been tested. There are three main reasons for a construction process:

1) The Analysis Model is not sufficiently formal.

2) Adaptation must be made to the actual implementation environment.

3) We want to do internal validation of the analysis results.

The construction activity produces two models, the Design Model and the Implementation Model. Construction is thus divided into two phases; design and implementation, each of which develops a model. The Design Model is a further refinement and formalization of the Analysis Model where consequences of the implementation environment have been taken into account. The Implementation model is the actual implementation (code) of the system.

C. Testing

Testing is an activity to verify that a correct system is being built. Testing is traditionally an expensive activity, primarily because many faults are not detected until late in the development. To do effective testing we must have as a goal that every test should detect a fault.

TABLE 1. COMPARISON BETWEEN TRADITIONAL APPROACH AND OBJECT-ORIENTED APPROACH

Traditional approach	Object-Oriented Approach
Used to develop theTraditional Projects that uses procedural programming.	Used to develop Object-oriented Projects that depends on Object-Oriented programming.
Uses common processes likes: analysis, design, implementation, and testing	Uses UML notations likes: use case,classdiagram,communication diagram, development diagram and sequence diagram
Depends on the size of projects and type of projects[7][8]	Depends on the experience of the team and complexity of projects through the numbers of objects
Needs to large duration sometimes to development to the large projects.	Need to more time than Traditional approach and leads that to more cost.

3. LEVEL OF TESTING

- Requirement testing
- Design testing
- Module testing
- Unit testing
- Integration testing
- System testing
- Acceptance testing

Comparison between traditional and OO testing

Unit testing: testing process on isolated function or method.

Testing process on isolated objects or class.

Integration testing: In traditional testing module by module integrate together and test their interface.

In OOT, module can share their components and classes inherited from base class.

System testing: The sub-system are integrated together to make asystem.

In OOT, testing can be made at class level, also called partition testing.

Traditional testing oriented on decomposition and functional, follow sequential approach.

OOT focuses on composition, and follow incremental approach.

4. CONCLUSION

In this era, object oriented technology is emerging field in software development. This paper presented mainly two approach of software testing, traditional and object oriented. Both have some pros and cons. Traditional approach rely on sequential and decomposition, larger algorithm into smaller ones like C. OO Approach rely on objects and classes, makes object – oriented projects using OO languages like C++, Java etc.Software testing is very important phase for software development . Testing can be done at any level as unit or module, integration, system testing. After taking overview of paper, will be able to taking a view, how many types of testing and what level.

REFERENCES

[1] Mike O‟Docherty, "Object-Oriented Analysis and Design Understanding System Development with UML 2.0", John Wiley & Sons Ltd, England, 2005.

[2] Magnus Christerson and Larry L. Constantine, “Object-Oriented Software Engineering- A Use Case Driven Approach “, Objective Systems, Sweden, 2009.

[3] Ian sommerville, “Software Engineering”, Addison Wesley, 7th edition, 2004.

[4] Pankaj Jalote, “An Integrated Approach to Software Engineering”, Springer Science Business Media, Inc, Third Edition, 2005.

[5] Grady Booch, “Object-Oriented Analysis and Design with applications”, Addison Wesley Longman, Inc, second Edition, 1998.

[6] Roger S. Pressman, “Software Engineering a practitioner‟s approach”, McGraw-Hill, 5th edition, 2001.

[7] M M Lehman,”Process Models, Process Programs, Programming Support”, ACM, 1987.

[8] Tim Korson and John D. McGregor,” Understanding Object-Oriented: A Unifying Paradigm”, ACM, Vol. 33, No. 9, 1990.

[9] Li Jiang and Armin Eberlein,” Towards A Framework for Understanding the Relationships between Classical Software Engineering and Agile Methodologies“, ACM, 2008.

[10] Luciano Rodrigues Guimarães and Dr. Plínio Roberto Souza Vilela,” Comparing Software Development Models Using CDM”, ACM, 2005.

[11] Alan M. Davis and Pradip Sitaram, “A Concurrent Process Model of Software Development”, ACM, Software Engineering Notes Vol. 19 No. 2, 1994.