JOURNAL OF ADVANCEMENT IN ENGINEERING AND TECHNOLOGY

Journal homepage: www.scienceq.org/JAET.php

Research Article Open Access

Four Basic Concepts of Object Technology: A Structuring Method, a Reliability Discipline, an Epistemological Principle and a Classification Technique

Girma Yohannis Bade
Department of Computer Science, Wolaita Sodo University, Wolaita, Ethiopia
*Corresponding author: Girma Yohannis Bade, E-mail: girme2005@gmail.com

Received: January 15, 2019, Accepted: March 11, 2019, Published: March 11, 2019.

ABSTRACT
The rapid evolution of every data has created to store and transmit all data standards, concepts, and technologies. The problem is the analysis of object-oriented data base systems that would be flexible enough how to make the best benefit of this data for object oriented data base management systems (OODBMS) and quickly developing work with evolving data. Object technology is an ideal vehicle to discuss such concepts without going into messy technologies. The object oriented method has grown out a meeting between a problem and a set of powerful ideas that provide at least as path to a solution then problem is easy to state transforming software constructions into industry activities. This paper is going to examine object technology in view of reliability, classification, epistemological principle and structuring method. Finally, this paper also concludes these four ideas are the basis of object technology.

Keywords: *object technology, reliability, classification, epistemological structuring.*

1. INTRODUCTION
As the name suggests, the basis of object technology is an object. The revolution brought about by object technology can be largely attributed to the idea of objects. The basics of object technology can be traced back to the 1960s [1]. A language called as Simula (abbreviation for Simulation language) first used the ideas of object technology. By confirming to the physicists' new way of thinking about systems with discrete probabilistic events, SIMULA actually implemented the representation and simulation of Aristotelian causality in a programming language [11]. In the object oriented model, systems are viewed as cooperating objects that encapsulate structure and behavior and belong to hierarchically-constructed classes. But what is object technology? Different books define the term OT differently. OT is a tools and techniques that for creating software from reusable parts. It is a lot like the idea of interchangeable parts that revolutionized manufacturing [7].

2. METHODOLOGY
2.1 Reliability principle
In a formal manner, the probability an item (e.g., system, subsystem, component, part) will perform its intended function with no failures for a given time period under a given set of operating conditions (environment).In brief, the likelihood of failure free performance for stated conditions. An inherent design characteristic (defined in the next paragraph) for no repairable items. Since reliability is a prime "design for" operational outcome, the reliability requirement is an integral part of engineering specifications [3, 5]. As contained in the formal definition of reliability, the reliability requirement or specification in complete form addresses four elements, namely, the:

a) Intended function (mission) to be performed.
b) Mission time.
c) Operating environment.
d) Probability of success (or no failure) the item will perform its intended function.

2.2 Classification method
Object technology depends based on the five fearful ideas: Decentralization, contracts, selfishness, classification, and seamlessness. OT classifies the objects into similar concepts by looking for their common characteristics (or properties) and groups them into hierarchies as follows.

Objects - It can represent a person, a bank account or any item that a program can handle. When a program is executed, the objects interact by sending messages to one another. For example, if 'customer' and 'account' are two objects in a program, then the customer object may send message to account object requesting for a bank balance. Each object contains data and code to manipulate data. Objects can interact without having to know details of each other's data or code. It is sufficient to know the type of massage accepted and the type of response returned by the objects.

Classes - it have just been mentioned that objects contain data and function or code to manipulate that data. The entire set of data and code of an object can be made a user-defined data type with the help of a class. In fact objects are variables of type class. Once a class has been defined, we can create any number of objects associated with that class. For example, mango, apple and orange are members of class fruit. If fruit has been defined as a class, then the statement fruit mango, will create an object mango belonging to the class fruit.

Data abstraction - the describing of objects by defining their unique and relevant characteristics (properties). Abstraction refers to the act of representing essential features without including the background details. To understand this concept more clearly, take an example of 'switch board'. You only press particular switches as per your requirements. You need not know the internal working of these switches. What is happening inside is hidden from you. This is abstraction, where you only know the essential things to operate on switch board without knowing the background details of switch board.

Data encapsulation -Encapsulation is the most basic concept of OOP. It is the way of combining both data and the functions that operate on that data under a single unit. The only way to access the data is provided by the functions (that are combined along with the data). These functions are considered as member functions in C++. It is not possible to access the data directly. If you want to reach the data item in an object, you call a member function in the object. It will read the data item and return the value to you. The data is hidden, so it is considered as safe and far away from accidental alternation. Data and its functions are said to be encapsulated into a single entity. Encapsulation leads to more stable systems [6].

Modularity-The act of partitioning a program into individual components is called modularity. It gives the following benefits.

a) It reduces its complexity to some extent.
b) It creates a number of well-defined, documented boundaries within the program.

Module is a separate unit in itself. It can be compiled independently though it has links with other modules. Modules work quite closely in order to achieve the program's goal.

Inheritance - Inheritance is the ability to write one definition as an extension of another. Only the properties and behaviors which are different need be described in the child object... All the properties and behaviors which are unchanged are obtained by reference to the parent. The main benefit from inheritance is that we can build a generic base class, i.e., obtain a new class by adding some new features to an existing class and so on. Every new class defined in that way consists of features of both the classes. Inheritance allows existing classes to be adapted to new application without the need for modification.

Polymorphism - the ability to make one object able to be coupled with a wide variety of others, relying on their all having a common external interface. Polymorphism is a key to the power of OOP. It is the concept that supports the capability of data to be processed in more than one form. For example, an operation may exhibit different behavior in different instances. The behavior depends upon the types of data used in the operation. Let us consider the operation of addition. For two numbers, the operation will generate a sum. If the operands are strings then the operation would produce a third string by concatenation. Polymorphism separates the different uses of an object. New uses can be added with no impact on the existing uses of the object. So from this point I can conclude that classification technique is one of the intellectual tool which is the main component of object technology.

2.3 Epistemological principle

Simulation as a research method has a long history in numerous disciplines and in many research communities. Physics, electronics, robotics, economies, logistics and production planning and control apply simulation, for instance. Furthermore, simulation is used worldwide as a method for explanation and prediction in different research and practitioner groups. In this context, several epistemological questions arise: What are the assumptions made when researches from different academic disciplines, communities and countries apply simulation? How does knowledge researches already gained about the "real world" influences the conduction of the research and the interpretation of the results? Are the basic assumptions made always the same? The epistemological principle addresses the question of how we should describe the classes. In object technology, the objects described by a class are only defined by what we can do with them: operations (also known as features) and formal properties of these operations (the contracts).

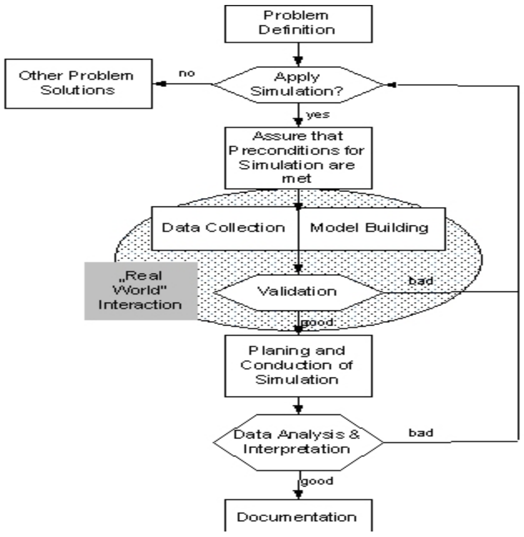

Figure 1: Simulation Process

2.4 Structuring method

Applies to software decomposition and reuse. Software systems perform certain actions on objects of certain types; to obtain flexible and reusable systems, it is better to base their structure on the object types than on the actions. The resulting concept is a remarkably powerful and versatile mechanism called the class, which in object-oriented software construction serves as the basis for both the modular structure and the type system [8,9]. It is also consists of breaking big problems into smaller problems, then further breaking those into still smaller problems, and so on, until a level of such simplicity is reached that the implementation is obvious to the programmer expected to do the coding [10].

3. Discussion

Object-oriented technology is both immense and far-reaching. End users of computer systems and computer-based systems notice the effects of object-oriented technology in the form of increasingly easy-to-use software applications and operating systems and in more flexible services being provided by such industries as banking, telecommunications, and cable television. For the software engineer, object-oriented technology encompasses object-oriented programming languages, object-oriented development methodologies, management of object-oriented projects, object-oriented computer hardware, and object-oriented computer aided software engineering, among others [4]. Object technology is an ideal vehicle to discuss such concepts without going into messy technologies. The object oriented method has grown out a meeting between a problem and a set of powerful ideas that provide, if not a solution, at least as path to a solution then problem is easy to state: transforming software constructions into industry activities. But this is not a new idea. So the main concern of any object enthusiast should not be whether object technology will be around in the future, but whether the OO concepts can avoid the kind of dilution that have plagued structured techniques. If everything is advertised as object-oriented, the burden is on the buyer to ascertain what OO is and what is not.

4. CONCLUSION

Areas that had traditionally resisted the influx of OO ideas are no longer immune. Even the scientific computing field is becoming increasingly OO, both with the spread of OO languages and libraries and with the increased influence of object ideas on new versions of traditional languages, most notably FORTRAN. Since high immunity is attained through object and object itself the building block of object technology &its core the combination of four ideas: a structuring method, a reliability discipline, an epistemological principle and a classification technique. Even if it is the combination of these four basic concept ,one strips away all of the confusing acronyms and jargon, the object technology approach is nothing more than a method, an approach to systems design which can be implemented without any changes to existing software technology.

REFERENCES

1. Rodney Collins, available <online> http://www.sewo.biz/object_technology_today/object_technology_today.php
2. Edward V;"object oriented concepts",available <online> http://www.ipipan.gda.pl/~marek/objects/TOA/oobasics/oobasics.html
3. Mayer B;Object technology conceptual perspective,1996
4. Salvatore T;et al" Research Frontiers in Object Technology"
5. http://www.managingchange.com/object/principl.htm
6. BECKER, J. and Niehaves, B. "Epistemological Perspectives on IS Research – Analyzing and Systematizing Epistemological Assumptions." Information Systems Journal (forthcoming), 2005.
7. William S,"Object technology for executives"Cambridge University press,1999
8. www.cs.dal.ca/smedley/vl98
9. George Coulouris, Jean Dollimore, Tim Kindberg: Distributed Systems, Concepts and Design. AddisonWesley 2001
10. Wolfgang Emmerich: Engineering distributed objects. John Wiley & Sons, Inc. 2000
11. Adam Reed" *Object-Oriented Programming and Objectivist Epistemology Parallels and Implications* "The Journal of Ayn Rand Studies 4, no. 2 (Spring 2003): 251–84.

Citation: Girma Yohannis Bade e*t al* (2019). Four Basic Concepts of Object Technology: A Structuring Method, a Reliability Discipline, an Epistemological Principle and a Classification Technique, J. of Advancement in Engineering and Technology, V7I3.03. DOI: 10.5281/zenodo.2595853.