International Journal of Computer Applications Technology and Research
Volume 5– Issue 8, 504 - 508, 2016, ISSN:- 2319–8656

# Object Oriented Programming (OOP) Approach to the Development of Student Information Management System

Onu, Fergus U.
Computer Science Department,
Ebonyi State University, Abakaliki – Nigeria

Umeakuka, Chinelo V.
Computer Science Department,
Ebonyi State University, Abakaliki – Nigeria

**Abstract:** It is not in doubt that good data management is essential to the success of every organization. Most instituions in Nigeria still adopts the use of Relational Database Management System (RDBMS) platform for the management of their students' information hence they grapple with the enormous and overwhelming challenges facing the RDBMS technique. This paper presents the Object Oriented programming concepts in the development of the system for student data management. The research used Object oriented analysis and design (OOA&D) and Agile methodology to realise a simple and easy to manage approach to data representation, storage and retrieval. The resulting model of Students Registration System at the departmental level drastically reduced maintenance cost and increased productivity. The system generally reduced the burden of data management, report generation and presentation and hence brought efficient resources utilisation to the institutions.

**Keywords:** Object Oriented Database, software maintenance cost, UML, data retrieval, Object Oriented Models.

## 1.0 INTRODUCTION

The measurement of success of any Institution is largely dependent on its record keeping capability and management process. To enhance the process, there is the need to deploy Object Orientation in student Information system in order to provide better speed and efficiency in the representation of students' data, at the development and maintenance stage. The use of the concept of object orientation (OO) in the analysis and design of the application would prove beneficial in terms of cost, energy and time [1]. For over a decade, Relational databases (RDB) have been the accepted model for storage and retrieval of huge volumes of data. This technique is faced with a lot of problems thus:

- Complexity of Application: Applications often require large amount of code to produce many varied reports, the level of complexity as measured by the interactions between modules is relatively low. So, too limited to handle large class of applications[15].
- Hierarchy and Relationship: Database model is less expressive and flexible in terms of network of connected pieces of information[15]
- Query languages (Invocation of Operations): Relational query languages are not computationally complete and programming environment can be less uniform. As a result, the bulk of the application code may not reside in the database and not managed by all of the database facilitates (e.g., concurrency, recovery, and version control). Hence, data needs to be copied into its virtual memory[15]
- Automatic type checking: Type failures are only detected at the end of a transaction when the new values are checked back in, so does not support automatic type checking. [15]

Object Oriented Database (OODB) model provided solution to many of the problems associated with RDB. Based on the concept of abstraction and generalization, object oriented models capture the semantics and complexity of the data.[3]. Many authors have stated that OODBs are optimized to provide support for object oriented applications, different types of data structures including trees, composite objects and complex data relationships. The OODB system utilizes the concepts of object-oriented languages and has the capability to handle complex databases efficiently and it can allow the users to define a database, with features for creating, altering, dropping tables and establishing constraints.

The principal strength of OODB is its ability to handle applications involving Complex and interrelated information [2], but in the current scenario, the existing Object-Oriented Database Management System (OODBMS) technologies are not competing in the market with their RDB counterparts [7]. Also, there are numerous applications built on existing relational database management systems (RDBMS). It is difficult, if not impossible, to move off those RDBs. Hence, [7] felt that there is need to incorporate the object-oriented concepts into the existing RDBMSs.

Student Management information system (SMIS) is a computer-based system used within an Institution of higher learning. It is designed to be a secure, confidential collection of data about students which help in proper administration and management of students at the departmental level in higher Institutions. It is a system to handling objects and object identity by deploying the concepts of encapsulation, classes, and inheritance in an efficient way.

The system became very desirable dut to the ever increasing need to manage the data of students with a more robust technique. That was the motivation for this work. So we aimed at transforming the Relational database systems of managing student data which are prevalent within the university systems in Nigeria into Object Oriented systems, and show efficiently how administration is made easier through this effort.

The identifying benefits of Object Oriented model (OOM) in Student Information System in addition to the analysis of how OOP concepts is applied in complex system to make it easier and safer to implement in information systems showed that the use of OOP in the system should be embraced to effectively check the excesses of students.

## 2.0 LITERATURE REVIEW

A lot of related work focused on features of Object Oriented Programming and systems relating it to Relational Database Systems while some worked on its application areas models

Through interrelated works on Object oriented analysis and modeling, [1] explored the basics and advancement of OOA, its utilization and implementation. The use of concepts of objects in analysis and application design to prove its benefits, identified the shortcoming and its solution at the project phase. In a reviewed by [4], the concepts of Class, Object and inheritance, based on extent of coupling and cohesion in OOS, and their effect on results produced, taking into consideration the run-time properties of programming Languages were presented.

[5] Surveyed the discretional access control issues and mechanism of both structural and behavioural aspects of subject to Object, Inter-object and Intra-Object and their effects on object oriented design (OOD). They also explored other authorization mechanisms beneficial to OODBMS. Object Oriented modeling using Inheritance and propagation properties for complex systems was analysed by [6]. They highlighted how OO approach has powerful tools for data structuring in terms of generalization, classification, Aggregation and Association. The importance of inheritance and propagation to model dependencies of property operations and values as well as in the implementation were seriously considered.

It was the focus of [7] to design an object-oriented database, through incorporation of object-oriented programming concepts into existing relational databases. The presented approach of the Object oriented programming concepts such as inheritance and polymorphism aids showed the efficiency in data mining. The experimental results demonstrated the effectiveness of the presented OO approach in successful reduction of implementation overhead. There was a considerable reduction in the amount of money paid for memory space required for storing databases that grow in size in the design of an OODB when compared to the traditional databases.

### 2.1 Concept of Objects, Classes and Inheritance in Object Oriented Systems

In the modern computing world, the amount of data generated and stored in databases of organizations is vast and continuing to grow at a rapid pace [8], the OODMS which employs the use of OOPL should be incorporated to be able to handle the system efficiently. The concepts of OOP are Object, Class and Inheritance is reviewed.

***Concept of Object and Class***

Object is a central abstraction that models a real world entity. Every object encapsulates some state and is further uniquely identified by an object-identifier.[5]. The word object is used for a single occurrence (instantiation) of data describing something that has some individuality and some observable behaviour. The terms object type, sort, type, abstract data type, or module refers to types of objects, depending on the context [6]. In designing an application, objects should conceptualize the design using the real world components as objects. The state of an object is made of the values of its attributes (that describe the real world entity modelled). In behaviourally object-oriented database, the object state is accessible only through the operations (methods) supported by its interface(s). Every operation (method) is associated with a method body that contains some piece of executable code that models the behaviour of the corresponding real world entity. Every object belongs to a type that is determined by its class, and is thus considered to be an instance of the class [5].

A class is thus akin to an abstract data type definition. Classes can be organized into class hierarchies enabling the sharing of structure and behaviour through the mechanism of inheritance [5].

In OO design, the coupling of a class means the measurement of the interdependence of class with the other classes. In a design of reasonable size design size is ten classes normally classes do not exist in absolute isolation [4].

***Concept of Inheritance***

Inheritance is the transitive transmission of the properties from one super class to all related subclasses, and to their subclasses [6]. Inheritance is beneficial in terms of high reduction rate of data redundancy and maintains high integrity of data, consistence and modularity.

The most important object-oriented concept employed in an OODB model includes the inheritance mechanism and composite object modelling [13].

An inherited class is the base class or super class or parent class, whereas derived class is the subclass or child class. Defined operations on super class apply to other objects of its subclass. Defined operations on subclass are not related to the super class objects.

We can have single or multiple inheritances, while single inheritance restricts relations to a strict hierarchical structure, multiple inheritances allows properties defined on several super classes to be accessed by one or more

subclass thereby, should be considered to achieve desired goal. Good design decision creates better relations among interacting objects and their properties.

## 2.2 Unified Modelling Language (UML)

A **UML** is a standard modeling Language to model the real world in the field of software engineering. A UML diagram is a partial graphical view of a model of a system under design, implementation, or already in existence. UML diagram is made up of graphical elements, UML nodes connected with edges ( flows) that represent elements system model. The UML model of the system might also contain other documentation such as use cases written as texts.

The kind of the diagram is defined by the primary graphical symbols shown on the diagram. Many UML diagrams exists but we look at the properties of Class diagram as we will be using it in the model (Table 1).

Table 1: UML Class and Object diagram properties

| Diagram | Purpose | Elements |
|---|---|---|
| Class diagram | Shows structure of the designed system, subsystem or component as related classes and interfaces, with their features, constraints and relationships - associations, generalizations, dependencies, etc. | class, interface, feature, constraint, association, generalization, dependency. |
| Object diagram | Instance level class diagram which shows instance specifications of classes and interfaces (objects), slots with value specifications, and links (instances of association). | instance, specification, object, slot, link. |
| Use case diagram | Describes a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors) to provide some observable and valuable results to the actors or other stakeholders of the system(s). | use case, actor, subject, extend, include, association. |

Source: http://www.uml-diagrams.org/uml-25-diagrams.html

## 3.0 OBJECT ORIENTED STUDENT INFORMATION MANAGEMENT SYSTEM (OOSIMS) SYSTEM METHODOLOGY

The Student Information system is a server-based system that uses an Object-Oriented approach to manage student registration at the department level. It consists of good data integrations features, good GUI to enhance user experience and flexible reporting features to deliver value to users.

In our design, we present the approach that extends the relational database system of managing student registration to an Object Database Management System (ODBMS) incorporated by utilizing the OOP concepts like Objects, classes, inheritances, and encapsulations. The modeled system makes use of the OOP relationship feature to show interaction among objects and classes, and these include Association, inheritance and Generalization This ability to represent classes in hierarchy is one of the OOP beauties.

Object Oriented Students Information Management System (OOSIMS) designed and modeled is for yearly record of new students and update of existing student information at the departmental level in higher institutions through the interfaces provided. A sample of the interface for the capturing of student bio data information is shown in Figure 1. It facilitates access to the information about a student at anytime, registered courses, and an id card is generated for every registered student to check impersonation during an examination

The System serves as a repository in the department, showing information about students

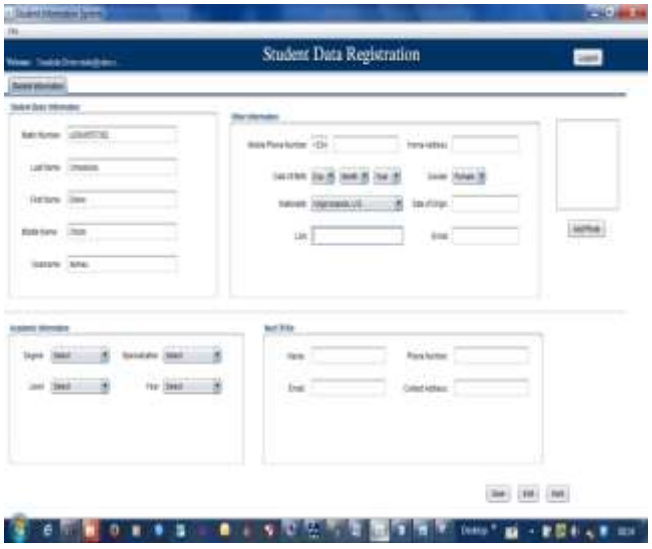


**Fig. 1: Interface of Student Data Registration**

## 3.1 Agile Methodology

The Agile Methodology employed, with the use of UML Class Diagram tool shows the structure of the Object Oriented Student Information Management system (OOSIMS). Its component as relates to entity type and responsibility, classes. The featured relationships include Inheritance, associations, generalizations.

UML 2 class diagram is one of the tools for representing object-oriented analysis and design. UML class diagrams in figure 2 shows the classes of the system, their interrelationships (including inheritance, association and Generalization, and the operations and attributes of the classes.

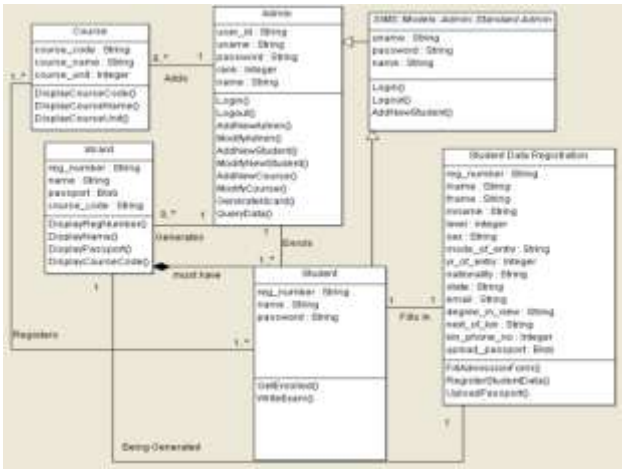

**Fig. 2: Class Diagram of the OOSIMS Structure**

## 3.2 Generalization and Inheritance Analysis

Classes and objects often show some similarities in attributes and methods. To maximize re-use of attributes and methods, Inheritance mechanism which "is a" and "is like" relationships, is deployed to avoid repetition of codes. These relations show high dependency nature to achieve a desired goal. These dependencies include generalization and association relationship types to reveal effective administration in student registration. Relationship Analysis among Classes is shown in table 2.

**Table 2. Class - Relationship Analysis.**

| Relationship / Class (Modifier) | Inheritance type | Inheritance Property | Generalization | Association |
|---|---|---|---|---|
| **Admin** (Root) | Super class | - | - | Must create 'Student' with 0..1 multiplicity |
| ***Standard Admin*** *(Abstract)* | Sub class | all* of 'Admin' except rank, all # of 'Admin' except create, modify Admin | Can be both 'Admin' and 'Student' | - |
| **Student** | Super class | - | - | -Must be created by 'admin' to register, -Register 'Course with 1..* multiplicity |
| **Student Data Registration** *( Active)* | Sub class | all* of 'Student' except password, | - | Must have 'Id card' to write exam |
| **Course** | Super class | - | - | Each 'Course' must be registered by many 'Student' with with 0..1 multiplicity city |
| **Id Card** *(Leaf)* | Sub Class | Multiple- some * and # of 'Course', 'Student Data Registration', 'Student'. | | must be generated for 'Student' with 1..* multiplicity |

* denotes Attribute, # denotes Method

## 3.3 Process design in OOSIM System

The Functionality of the system is described using a Use Case Diagram illustrating the sequence of actions / interaction between the agents/actors and the database. Figure 4 illustrates the activities of actors in the system. The Actors are Admin, Database, and Student.. The generalization link indicates that the abstract '*Standard Admin'* can be an 'Admin' and a 'Student'. The actor 'Student' has direct use cases to the database indicating his actions in the system. The abstract use case '*include*' indicates that the student will always update the course as he moves from one level to another. The 'admin' creates student using reg number, student name and password. The 'Student' then login to fill the registration form, uploads passport, and register his courses

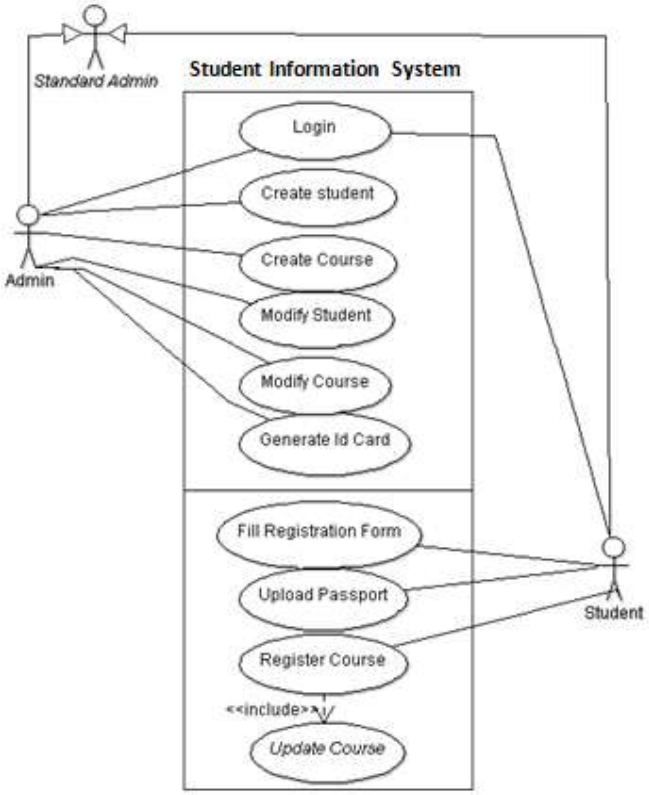


**Fig.4: Use Case Diagram of OOSIMS Behaviour**

## 4 DISCUSSION AND EVALUATION

Introducing object orientation through the OOSIMS in student information management described the main features of OOP in a database. Adopting Object Oriented Paradigm in student information management brings out the system functionality. It also showed objects and classes with their properties and operations. With OODB, the research has Identified the relationships and associations, as well as the coordination constraints among interacting objects and functional classes. All these have helped in the following:

1. Reduction of codes for developing an application and modification of similar functionality if already existed.
2. Enable the re-use of design and code function with minimal modification to suite a need (i.e. inheritance).
3. Improved Maintainability of OOSIMS by allowing complex systems broken into smaller manageable units.
4. Reduced cost and time of developing Student Information Management System due to the encouragement of team work.
5. Maintains data integrity by hiding access from unauthorized objects and users (i.e. encapsulation).

## 5.0 CONCLUSION

The OODB platform of student information system has been shown to be the most secured and flexible in the development of Student Information system. The object

oriented concepts impact in the areas of resource reuse, extensibility, maintenance, and scalability cannot be ignored. Inheritance and other relationship types employed in the structural definition makes the system concise. Its beneficial features like redundancy reduction, improved cost effective, data integrity and easy maintenance are to be considered by authority to embrace OODMS in administration of students to improve productivity.

## 6.0 REFERENCES

[1] Clara Kanmani A, S. Mohan Kumar, and Abhishek Y S S,. "Interrelated Research Works And Importance Of Object Oriented Analysis And Modeling", *ISOR Vol.5 No. 2, .785, ISSN: 2277-965, 2016.*

[2] Mansaf Alam, Siri Krishan Wasan, "Migration from Relational Database into Object Oriented Database," *Journal of Computer Science, Vol. 2, No. 10, pp. 781784,* 2006.

[3] Joseph Fong, "Converting Relational to Object Oriented Databases," *SIGMOD Record 1997, Vol. 26, No. 1,* 1997.

[4] Narendra Pal Singh Rathore1, Ravindra Gupta (). "A Novel Class, Object and Inheritance based Coupling Measure (COICM) to Find Better OOP Paradigm using JAVA", *International Journal of Advanced Computer Research Vol 1 , No1, ISSN : 2249-7277, 2011.*

[5] Roshan .K.Thomas and Ravi .S.Sandhu , " Discretionary access Control in object-Oriented databases: Ssues and research direction", *Proc.ofthe16thNIST-NCSC National Computer Security Conference, Baltimore, pages63-74, September 1993.*

*[6]* Max J. Egenhofer and Andrew U. Frank, " Object-Oriented Modeling in GIS: Inheritance and Propagation", *Res. National Center for Geographic Information and Analysis and Department of Surveying Engineering University of Maine USA.*

[7] Ajita Satheesh and Ravindra Patel, "Use Of Object-Oriented Concepts In Databases For Effective Mining", *International Journal on Computer Science and Engineering ,Vol.1, No. 3, pp. 206-216, 2009.*

[8] Satchidananda Dehuri, "Genetic Algorithms For MultiCriterion Classification And Clustering In Data Mining", *International journal of computing and information sciences, Vol. 4, No. 3, pp. 143-154, 2006.*

[9] Shermann Sze-Man Chan, and Qing Li, "Supporting Spatio-Temporal Reasoning in an object-Oriented Video Database System", *1999.*

[10] Satchidananda Dehuri, "Genetic Algorithms For MultiCriterion Classification And Clustering In Data Mining", *International journal of computing and information sciences, Vol. 4, No. 3, pp. 143-154, 2006.*

[11] Urban, S.D. and S.W. Dietrich, "Using UML Class Diagrams for a Comparative Analysis of Relational, Object-Oriented, and Object-Relational Database Mappings." *ACM SIGCSE Bulletin. 35(1):21-25, 2003.*

[12] Kelly Nunn-Clark, Lachlan Hunt, Teo Meng Hooi and Balachandran Gnanasekaraiyer, "Problems of Storing Advanced Data Abstraction in Databases," *In Proceedings of the First Australian Undergraduate Students' Computing Conference, pp. 59-64, 2003.*

[13] Cristian Seech et. Al. Object Oriented Modeling of Complex Mechatronic Components for the manufacturing Industry", IEEE/ASME Transactions on Mechatronics Vol. 12 2007, Pg. No. 696

[14] Jun Zhu et. Al. "Application of Design Patterns for Object Oriented Modeling of Power Systems", *IEEE Transactions on Power Systems, Pg. No. 532, 1999.*

[15] *K*aren E. Smith, Stanley B. Zdonii, "Intermedia: A Case Study of the Differences Between Relational and Object-Oriented Database Systems" *OOPSLA '87 Proceedings, ACM O-8979 l-247-0/87/00 lo-0452 $ I .50, 1987.*

*[16]* Victor T Sarinho et. Al. "Combine Feature Modeling with Object Oriented concepts to manage software viability", *IEEE IRI, Pg. No. 344, 2010.*

[17] Ulrich Frank, "Delegation: An Important Concept for the Appropriate Design of Object Models," *Journal of Object-Oriented Programming, Vol. 13, No. 3, pp. 1318, 2000.*

*[18]* Joseph Fong, "Converting Relational to ObjectOriented Databases," *SIGMOD Record, Vol. 26, No. 1, 1997.*

[19] Kitsana Waiyamai, Chidchanok Songsiri and Thanawin Rakthanmanon, "Object-Oriented Database Mining: Use of Object Oriented Concepts for Improving Data Classification Technique", *Lecture Notes in Computer Science, Vol: 3036, pp: 303-309, 2004.*

[20] J. Blakeley, "Object-oriented database management systems," *Tutorial at SIGMOD, Minneapolis, MN, May 1994.*