International Journal of Computer Applications Technology and Research
Volume 5– Issue 8, 533 - 538, 2016, ISSN:- 2319–8656

Harnessing Object–Oriented Programming Techniques for Transaction Processing

Onu Fergus U. (PhD)
Department of Computer Science,
Ebonyi State University,
Abakaliki - Nigeria

Akpan Abasiama G.
Department of ICT,
Ritman University,
Ikot Ekpene, Nigeria

Nnanna Emmanuel E.
Department of Computer Science
Ebonyi State University,
Abakaliki- Nigeria

Abstract: This study focused on harnessing object-oriented programming techniques for Transaction Processing. In this way, object-oriented programming reintroduces techniques for managing software components. On the other land, Transaction processing are means of managing classic software structure for concurrent accesses to global data and for maintaining data consistency in the presence of failures. Hence, in this study, object-oriented programming techniques are examined in a view to structure units that encapsulate complex behaviour and embrace groups of objects and method calls. We collected data from both primary and secondary sources, to elicit information from stakeholders in software development industry. It was unraveled that there is a high positive relationship between object-oriented programming techniques and Transaction processing.

Keywords: Object based programming, Abstraction, Transaction Processing, Modularity, Concurrency.

1.0 INTRODUCTION

In the words of [1], software technology is making a transition from the view that programs are action sequences to the view that programs are collections of interacting software components. Object – oriented programming is a form of component – based software technology whose software components are objects and classes. In the way, programmers can create relationships between one object and another. For example, objects can inherit characteristics from other objects [2].

Kienzle [3] posits that complex systems often need more elaborate concurrency features than the ones offered by concurrent object – oriented programming languages. The existing single method approaches do not scale well, since they deal with each single operation separately, hence, there is a need for structuring units that encapsulate complex behavior and embrace groups of objects and method calls. These units should represent dynamic systems execution as opposed to the static declaration of objects inside objects. A Transaction processing is an interaction in the real world, usually between an enterprise and a person or another enterprise, where something is exchanged. It requires the execution of multiple operations, it must run in its entirety, it must be incrementally scalable, and records of transactions, once completed, must be permanent and authoritative. Object – oriented programming presents these collaborative techniques and hence poses the tendency to provide great success in Transaction processing.

Meyer [4] defined, object-oriented programming in a distinct manner, as a type of programming in which programmers define not only, the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure. Objects provide an ideal mechanism for implementing abstract data types, which includes stacks, trees and hash tables. It offers multi threading system in building control within applications developed as compared to conventional language; thereby solving real life problems by making everything (process included) an object and having control reside within each object, i.e. at anytime multiple objects could be executing an operation and communicating with other objects with the concept of message passing; which is simply an invocation of an operation in another object.

In the other hand, transaction processing has been an important software technology for 40 years. Large enterprises in transportation finance, retail, telecommunications, manufacturing, governments, and the military are utterly dependent on transaction processing applications for electronic reservations, banking, stock exchanges, order processing, music and video services, shipment tracking, government services, telephone switching, inventory control and command control [5]. Transaction processing systems have to handle high volumes efficiently, avoid errors to concurrent operation, and producing partial results, grow incrementally, avoid downtime, never lose results, offer geographical distribution, be customizable, scale up gracefully, and easy to manage.

Transaction processing requires the execution of multiple operations and if runs, it must run in its entirety and for high performance, transactions must execute concurrently. It is now interesting to focuses on the harnessing object-oriented processing techniques for Transaction processing.

The objectives of this study among others are:

i. To identify object-oriented programming techniques for transaction processing.
ii. To identify base principles and concepts that are related to object – oriented programming.
iii. To relate the object – orientation mechanisms to transaction processing.
iv. To focus on preserving and guaranteeing important properties of the data objects (resources) accessed during a transaction.

The relevance of this study lies in structuring dynamically created objects with object-oriented notions with extends to include temporary association of data and operations. This aid in handling high volumes of transactions efficiently, avoid errors, avoid partial results, downtime and ultimately enhance the ease with which we manage transaction operations.

2.0 REVIEW OF RELATED LITERATURE

2.1 Object-Oriented Programming (Definition and Evolution)

Many scholars have defined object-oriented programming in unique way, but we will consider its definition by [6] which said that "A programming language is said to be object-based if it supports objects as a language feature, and is said to be object-oriented if, additionally, objects are required to belong to classes that can be incrementally modified through inheritance"

The definition above clearly depicts Object-oriented programming being based on the principles of object – orientation and is based on the object-oriented concepts. It is a new way of organizing and developing programs and has nothing to do with any particular language. This study paper defines object – oriented programming has encapsulation of data as well as operations applicable to that data into objects.

[3] argued that the early programmers thought of programs as instruction sequences. Procedure-oriented languages introduced procedural abstractions that encapsulate sequences of actions into procedures. The procedure-oriented paradigm has strong organizing principles for managing actions and algorithms, but has weak organizing principles for managing shared data.

Typed languages introduced the notion of a data type. A type characterizes a group of values, and a set of operations applicable to those values. During that period, objects provided define interface and hiding the internal implementation in what is termed abstract data types - ADT.

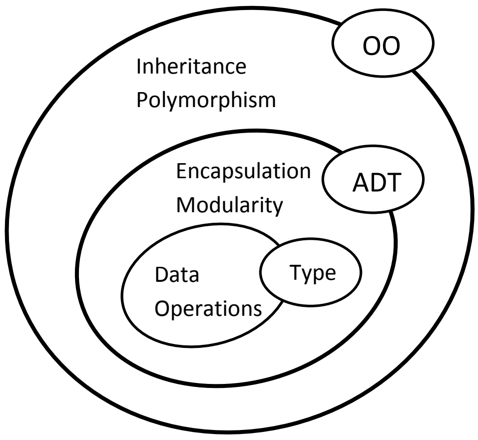


Fig 1: Evolution of Programming Language Concept.

Table 1: Evolution of Object – Oriented Programming Languages [7]

S/n	OOP Languages	Year	Description
1.	Simula	1967	Developed for simulating discrete Systems
2.	Smallalk	1970s	Developed for simulating graphics-oriented systems.
3	Ada	1980	A structured, statically typed, imperative, wide spectrum, and object oriented high-level computer programming language built to support extremely strong typing, explicit concurrency, offering tasks, and synchronism easy eg. Passing, protected objects and non-determinism.
4	C++	Early 198	A simpler version of c++ developed by sun Java. It is meant to be a programming language for video-on-demand applications and internet applications development.

Features and Concepts of Object Oriented Programming

The features of object – oriented programming offers robust techniques that give adequate support to Transaction processing. It creates a lot of ease in Transaction processing. It includes the concept of:

- **Abstraction:** The process of picking at (abstractly) common features of objects and procedures.
- **Class:** A category of objects. The class defines all the common properties of the different objects that belong to it.
- **Encapsulation:** The process of combining elements to create a new entity. A procedure is a type of encapsulation because it combines a series of computer instructions.
- **Information Hiding:** The process of hiding details of an object or function. Information undying is a powerful programming technique because it reduces complexity.
- **Inheritance:** A feature that represents the relationship between different classes.
- **Interface:** The languages and codes that the applications use to communicate with each other and with the hindrance.
- **Messaging:** Message passing it a form of communication used in parallel programming and object – oriented programming.
- **Object:** A self contained entity that consists of both data and procedures to manipulate the data.
- **Polymorphism:** A programming language's ability to process objects differently depending on their data type or class.
- **Procedure:** A section of a program that performs a specific task.

2.2 Transaction Processing (TP)

Definition and Evolution

Gray [8] defined a Transaction processing system (TPS) is an information processing system for business transactions involving the collection, modification and retrieval of all transaction data. Characteristics of a TPS include performance, reliability and consistency. TPS is also known as transaction processing or real-time processing. [5] x-rayed the origin of Transaction processing as shown in table 2.

Table 2: Evolution Of Transaction Processing – TP [5],

Year	Name	Manufacturers	Purpose
1960	IBM transaction Processing Facility (TPF)	IBM	Airtime control program (ACP)
1966	IBM information Management System (IMS)	IBM	Joint hierarchical data base with extensive capabilities
1969	IBM customer in frustration control system (CKS)	IBM	used for rapid, high volume online processing
1980s	Tuxedo Transactions for Unix, extended for distributed operative.	Oracle TPS	A Cross-platform
1970S	UNIVAC Transaction Interface package (TIP)	UNIVAC	A Transaction processing monitor
1980s	Burroughs corporation MCP	Burroughs	generalized message control system
1985	(DEC) Application control and management system (ACMS)	DEC	Creating and Controlling online transaction processing (OLTP).
1991	Transare Encina	IBM	A Transaction system

2.3 Features of Transaction Processing

The following features are considered important in evaluating transaction processing systems.

i. **Performance:** Fast performance with a rapid response time is critical. Transaction processing systems are usually measured by the number of transaction they can process in a given period of time.

ii. **Continuous Availability:** The system must be available during the period when the users are entering transactions. Many organizations rely heavily on their TPS; a breakdown will disrupt operations or even stop the business.

iii. **Data Integrity:** The system must be able to handle hardware or software problems without corrupting data. Multiple users must be protected from attempting to change the same example two operators cannot sell the same seat on an airplane.

iv. **Ease of User:** Often users of transaction processing systems are casual users. The system should be simple for them to understand, protect them from data entry errors as much as possible, and allow them to easily correct their errors.

v. **Modular Growth:** The system should be capable of growth at incremental costs, rather than requiring a complete replacement. It should be possible to add, replace, or update hardware and software components without shutting down the system.

2.4 Properties of Transaction

In conclusion, [5] said that a transaction is a unit of work that has the following properties:

i. **Atomicity:** A transaction should be done or undone completely and unambiguously.

ii. **Consistency:** A transaction should preserve all the invariant properties (such as integrity constraints) defined on the data.

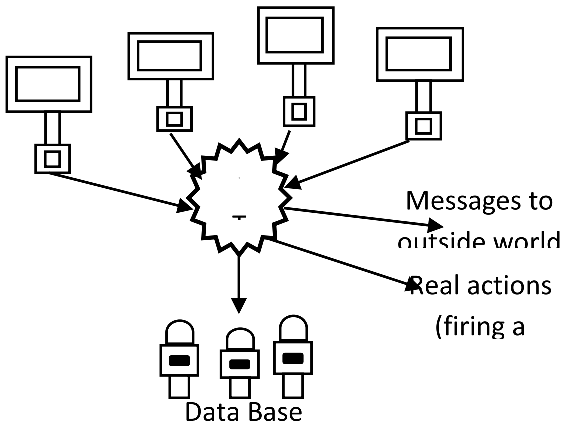
iii. **Isolation:** Each transaction should appear to execute independently of other transactions that may be executing concurrently in the same environment.

iv. **Durability:** The effects of a completed transaction should always be persistent.

2.5 Types of Transaction Processing

i. **Processing in batch:** Transactions may be collected and processing. Transactions will be collected and later updated as a batch when it is convenient or economical to process them. Historically, this was the most common method as the information technology did not exist to allow real-time processing.

ii. **Processing in real-time**: This is the immediate processing of data. It provides instant confirmation of a transaction. It may involve a large number of users who are simultaneously performing transactions which change data. Because of advances in technology (such as the increase in the speed of data transmission and larger



band-with), real-time update in possible.

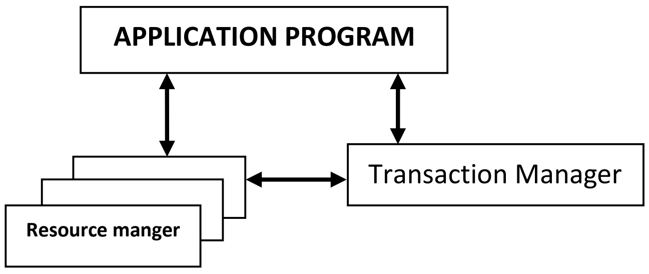


Fig. 2: Simplified explanation of TPS

Fig.3: Issues in building transaction applications:

To understand the issues involved in building Transaction applications, consider an order process application with the architecture shown in figure 3.

2.5 Standard Operations in Transaction Processing:

Transaction processing scheme is supported by the following standard primitive operations: Begin, Comit, and Abort.

After beginning a new transaction, all update operations on transaction objects are done on behalf of that transaction. At any point during the execution of the transaction if can abort and once a transaction has completed successfully is committed, the effects become permanent and visible to the outside [8].

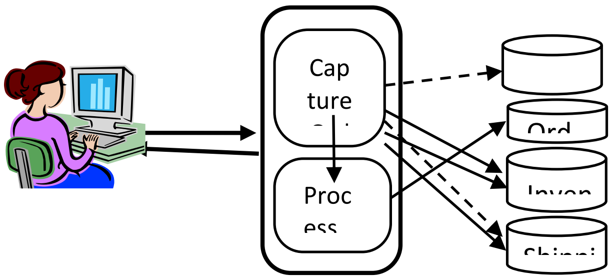


Fig. 4: ***X/Open Transaction model (XA).***

The transaction manager process starts, commit, and abort. It talks to resource managers to run Two Phases commit [5].

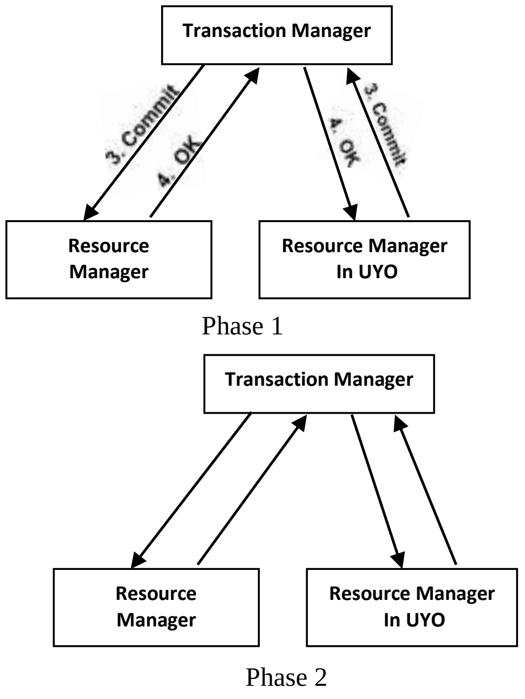


Fig. 5: ***The Two-phase commit protocol***.

In phase 1, every resource manager durably saves the transaction's updates before replying "I am prepared". Thus, all resources managers have durably stored the transaction's updates before any of the commits in phase 2 [5].

3.0 METHODOLOGY

3.1 Data Collection

We studied harnessing object – Oriented programming techniques for Transaction processing with data from two main sources thus:

a. **Primary Source:** We carried out a study using a questionnaire with a 10- point items. The 10-point items were structured to achieve the operational objectives of the study using the modified 5 Likert Scale of Strongly Agreed (SA), Agreed (A), Undecided (UD), Disagreed (D), Strongly Dis agreed. A total of 125 respondents out of which 10 were lecturers, 15 were freelance programmers in Port Harcourt and 100 were final year students of computer science. These respondents were selected from three different universities namely: University of Portharcourt, Portharcourt, University of Calabar, Calabar, University of Uyo, Uyo all in Nigeria, and the questions sought the views of the above named groups of persons' on harnessing object – oriented programming techniques for Transaction processing.

b. **Secondary Source:** We extract information from existing from existing computer science journals, text books, laboratory manuals and manuscripts, etc. The internet was a major source of the secondary data. People views were equally considered.

3.2 Data Analysis and Results Presentation

Table 3 shows the occupational distribution of the interviewee. The opinion of 125 respondents were sampled and responses collected and analyzed on a 5 – point likert type scale as shown in table 4.

Table 3: Occupation distribution of interviewed respondents

S/n	Respondents Occupation	No.	Percentage (%)
1.	Software Developers	15	12%
2.	Lecturers	10	8%
3.	Final Year Students	100	80%
	Total:	**125**	**100%**

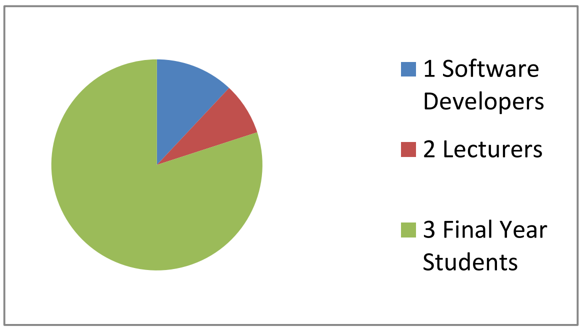


Fig. 6: Pie Chart showing the occupational distribution of interviewed respondents.

Table 4: Shows the opinion of the respondent to the questions presented by the interviewer. These questions were presented to freelance programmers, Lecturers and final year students of computer science department selected from the three Nigerian Universities mentioned in section 3.1.

Table 4: Questions and responses by respondents

S/N	QUESTIONS	X	F	FX	*X* (MEAN)	%
1.	Objects oriented programming languages encourage programming in modules?					
	• Strongly Agree	5	100	500	4.80	80.00
	• Agree	4	25	100		20.00
	• Undecided	3	0	0		0.00
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0	0		0.00
2.	Object oriented programming languages have concept of objects, class, data abstraction, inheritance, and polymorphism?				4.62	
	• Strongly Agree	5	83	417		66.67
	• Agree	4	34	136		26.67
	• Undecided	3	8	25		6.67
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0	0		0.00
03.	Program can be divided into modules of task and tested separately in OOP?					
	• Strongly Agree	5	100	500		80.00
	• Agree	4	17	68	4.74	13.33
	• Undecided	3	8	24		6.67
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0			0.00
4.	OOP has Internals Process Concurrency?					
	• Strongly Agree	5	83	415		66.67
	• Agree	4	25	100	4.53	20.00
	• Undecided	3	17	51		13.33
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0	0		0.00
5.	OOP has Inter-Process Communication?					
	• Strongly Agree	5	17	85		13.33
	• Agree	4	17	68	3.41	13.33
	• Undecided	3	91	273		73.33
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0	0		0.00
6.	OOP supports shared data structure?					
	• Strongly Agree	5	25	125		20.00
	• Agree	4	42	168	3.34	33.33
	• Undecided	3	17	51		13.33
	• Disagree	2	33	66		26.67
	• Strongly Disagree	1	8	8		6.67
7.	Transactions allow shared data structures?					
	• Strongly Agree	5	67	335		53.33
	• Agree	4	58	232	4.54	46.67
	• Undecided	3	0	0		0.00
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	0	0		0.00
8.	Conventional objects bind data structures to operations in a permanent union?					
	• Strongly Agree	5	67	335	4.34	53.33
	• Agree	4	42	168		33.33
	• Undecided	3	8	24		6.67
	• Disagree	2	8	16		6.67
	• Strongly Disagree	1	0	0		0.00
9.	Are transactions object-based?					
	• Strongly Agree	5	58	290		46.67
	• Agree	4	42	168	4.14	33.33
	• Undecided	3	17	51		13.33
	• Disagree	2	0	0		0.00
	• Strongly Disagree	1	8	8		6.67

10.	Transactions may be viewed as extending traditional object-oriented notions to include dynamically created objects?					
	• Strongly Agree	5	50	250	3.86	40.00
	• Agree	4	33	132		26.67
	• Undecided	3	17	51		13.33
	• Disagree	2	25	50		20.00
	• Strongly Disagree	1	0	0		0.00

4.0 DISCUSSION AND EVALUATION

Most of the questions (8 – in number) listed for the opinion of the respondents had high arithmetic mean as shown in Table 4. This showed that greater number of the respondents agreed that OOP has a positive impact on Transaction processing. Questions five (5) and six (6) have arithmetic mean of 3.41 and 3.34 respectively which depicts the fact that the respondents do not see the tested parameters to have a positive impact that harness OOP techniques for Transaction processing.

About 80% of the respondents opined that Transaction processing is object-based hence the use of OOP has brought an increase in the production of millions of Transaction processing packages. Harnessing object – oriented programming techniques for Transaction processing would ensure the development of better and more efficiet transaction processing packages. In a nutshell, object – oriented programming has brought the dawn of a new epoch in Transaction processing. The key techniques of OOP (namely: abstraction, classes, encapsulation, information hiding, inheritance, interface, messaging, objects, polymorphism and procedure) has been shown to enhance transaction processing.

5.0 CONCLUSION

The discourse in this study centered on harnessing object – oriented programming techniques for transaction processing. In doing this, we espouse different object – oriented programming techniques in line with the objectives of the study. Data from respondents were presented and analyzed with statistical tools and charts. The analysis showed that object-oriented processing techniques are very suitable for transaction processing. The base principles and concepts as it is related to object – oriented programming, relate the object – orientation mechanisms to transaction processing and focus on preserving, and guaranteeing important properties of the data objects (resources) accessed during a transaction.

6.0 REFERENCES

[1] Wegner, P. (1991): Perspectives on Object – Oriented Design. *Technical Report No. CS -91 – 01*; Rhode Island, USA, 1991.

[2] Booch, G. (1994): *Object-oriented Analysis- 2nd Edition*. Cummings Publishers, Redwood city, USA.

[3] Kienzle, J. (2001): Open Multi-threaded transaction: A Transaction Model for Concurrent Object Oriented Programming. Cannes, France.

[4] Meyer, B.: *Object – oriented software construction, 2nd edition*. ISE, Santa Barbara, California, USA.

[5] Bernstein, P.A; Newcomer, E. (2009): *Principles of transaction processing,* 2nd edition. Morgan Kanfamann Publishers, USA.

[6] Wegner, P. (1990): Concepts and paradigms Expansion of Oct 4 OOPSLA – *89 Keynote Talk*, USA.

[7] Onu, F. U, Osagie, S.U, John- Otumu. M.A, Igboke M. E (2015): OOP and its calculated measures in Programming Interactivity. *Journal of Mobile Computing & Application (IOSR- JMCA), vol.2, pp.26-34*

[8] Gray, J. and Reuter, A. (1993): *Transaction processing: concepts and Techniques*. Morgan Kanfamann Publishers, USA.