Object-Oriented Programming and Software Development

Nwokoro Ifeanyi Stanly

IKWO, Ebonyi, Nwokoro Ifeanyi Stanly, Nigeria

Abstract
Object-Oriented Programming (OOP) Software Development Paradigm, birthed some few decades ago with provisions for varied understanding of programming languages than the primordial procedural programming parlance in software development (FORTRAN, Pascal, C and etcetera). It is obvious that the OOP ab-initio didn't provide some resounding assurances and attractive structured codes to the users without the use of "goto" provisions. OOP at this point in time has attained some feet as excellent programming practice are now guaranteed and be defined. OOP is attributed as a methodology or a tactic which focused on such ways objects work together to express data sharing. It has changed the procedural conception of programming where vast attention is saddled on procedures and logic. There are seven essential concepts in Object-Oriented Programming (OOP) Software Development. First is an Object. An Object is an encapsulated entity that provides sets of predetermined services at a given period of time. It is also looked upon as anything that has existence and with some value. Subsequently is Class; a class is the main body of a system. It creates the federating units of a system. We also have Generalization, Inheritance, Abstraction, Polymorphism and Encapsulation. Not all programmers believe that OOP is a methodology. This paper will also show how the concepts of OOP can be used in modeling of Geographic Information Systems (GIS) software and the use of OOP in the design of features of GIS software and it's production.

Keywords:
Object-Oriented Programming, Procedural Programming, Object, Software Development Class, Inheritance, Polymorphism, Encapsulation, Abstraction, Generalization, Geographic Information Systems.

1. Introduction

Software is termed a group of related programs that perform specific functions. The software development life cycle is a process of building good software and in life cycle stages provides quality and correctness of good software [1]. Object-Oriented Programming (OOP) is a programming language that is designed within objects instead of procedures and emphasis on data rather than logic. The Object Oriented paradigms suggest we should model instructions in a computer program with the data they manipulate and store these as components together. One advantage of doing this is we get reusable software components [2]. In OOP, data is seen as input which is processed for the resultant information.

Manuscript received March 5, 2026
Manuscript revised March 20, 2026
https://doi.org/10.22937/IJCSNS.2026.26.3.17

2. Literature Review

2.1 Review of Related Work

Computer programming has passed through a period of chaotic as it has made programmers to undergo an uneven learning and application processes. Codes are fashioned in some ways that a programmer may not decipher what another programmer may have presented with their codes. Worst it is that a programmer who wrote some codes may forget the essence of such codes after a while. Computer Programming commenced with some great works from John Von Neumann, Charles Babbage, and etcetera. The Assembly language, other procedural-based programming of 1950s, Structured programming techniques of late 1960s, Modular programming concepts 1970s and Object-Oriented programming of late 1980s have taken their turns to influence the computer programming and software development landscape.

2.2 Assembly Language:

Assembly language as acknowledged by Wikipedia, is a programming Language that can be used to directly tell the computer what to do. An assembly language is alike to the language of computer, apart from that it employs some words called *mnemonics* in replace for numbers. Computer does not understand an assembly program without an assembler. Assembler changes the program into machine readable code and substituting the mnemonic with the binary outlook which they represent. It is a programming language that is a close approximation of the binary machine code [3].

2.3 Procedure-Based programming:

Procedural programming uses subroutines as functions. It is built on procedures and functions other than objects and data. In procedural programming, codes are structured into smaller units called procedures. As in software development, these functions accept inputs process it and display output. The vital information at this point is that there is no correlation between the functions and the procedures with the data at which is being utilized. Function as well as the procedure carryout their different assignments and produce the exact expectations or results as long as the appropriate information is given. So in a procedural system our functions/procedures use data "given" to them (as parameters) but also directly access any

shared data they need [4]. It is trite to note that procedural programming gives some sets of commands by informing the computer of the exact tasks in a stepwise refinement from the foremost line of codes to the other lines. Examples of the procedural programming languages are Pascal, C, FORTRAN and BASIC.

2.4 Object-Oriented programming:

Object Oriented Design paradigm is a dynamic and vast area for research purposes in software development. This paper will expose application developers and thrill seekers to an initiative which deals on pattern and object specifications. Object Oriented Programming paradigm is centered on Classes and in Objects. For an instance is a class of Animals. Classes also have subclasses and could inherit properties. Example of sub-class could be a dog called Bingo. Bingo eats meat as all male dogs in Nigeria eat meat. This paper would give details on the importance of OOP and software development which would inculcate some specific understanding of dependency and collaboration benefits between the attributing patterns and paradigm shift.

3.0 Concepts of Object-Oriented Programming

Object-Oriented Programming concepts are exceedingly important in program development. The fundamental concepts of OOP are as follows;

- **Objects**
- **Classes**
- **Abstraction**
- **Encapsulation**
- **Inheritance**
- **Polymorphism**

3.1 Objects: Objects are the states and behaviors of entities in an Object Oriented structure. Object is an encapsulated entity that provides a set of services at a given time in software development. In programming, problems are examined with regards to objects and their nature of communication amongst them. Objects are bound to interact with one another at the point of program execution by sending messages and other communications within without their precise knowledge of individual details to data and code sequence.

3.2 Classes: A 'class' is a software design which describes the general properties of something which the software is modeling [5]. Class is the main body of a program as it illustrates some specifics during program initiation. A class is seen also as a set of objects with related types. If a class is defined, it provides the creation of objects that belong to that particular class. Class diagrams are vital in the description of system architecture.

3.3 Data Abstraction: Abstraction is attributed in software development as an act of presenting important characteristics without the need to instill some surrounding clarifications. Data Abstraction is a design technique that focuses on the essential attributes and behavior. It is a collection of essential attributes and behavior relevant to programming a given entity for specific problem domain, relative to perspective of the user [6]. The principle of abstraction is wholly used to define some attributes in classes and storing data in Objet Oriented Programming. Abstraction allows us to consider complex ideas while ignoring irrelevant detail that would confuse us [7]

3.4 Encapsulation: In OOP classes do encapsulation and the variables, methods and functions are hidden inside them and manage coding in software systems development [8]. The knowledge of an object is no longer necessary once the object is created unlike in structural programming. The object has some capabilities to conceal some vital parts of it from nosy and thrill seekers in order to avert interference of its contents. Encapsulation allows us to focus on what something does without considering the complexities of how it works [9]. In encapsulation, codes are combined and operated as a distinct measure. This ensures security within the data from misuse and abuse.

3.5 Inheritance: Inheritance is a concept in Object Oriented Programming by which objects can acquire the characteristics of objects in another class and ensures reusability of codes without the stress of creating them again. Inheritance gives any user the provision to utilize the same code in his program just by changing the definition of its variables [10]. Inheritance allows reusability of codes between classes.

3.6 Polymorphism: In polymorphism, objects could be processed in a different way with the help of their data types and classes. It is the aptitude at which dissimilar objects react to a message in different patterns by taking more than a composition. It allows single name or operator to be associated with different operations [11]. Polymorphism gives the way to represent variables as a group or a common way to represent the variables [12]

4. Object Oriented Programming Paradigm

Object-Oriented programming paradigm is constructed beyond the outlook of the structured programming paradigm. It is centered on assisting programmers to cushion effects of structured programming with the analysis and design undertakings during software development by making certain that the software is stout and easy to maintain after a long time.

OOP is an enhanced technique of resolving systems dilemma and harm as compared to the procedural programming languages. OOP utilizes classes that contain variables and functions as it is called a modular kind of programming structure.

OOP is a type of programming in which programmers define not only the data type of a data structure, but also the types of operations that can be applied to the data structure. In this way, the data structure becomes an object that includes both data and functions. In addition, programmers can create relationships between one object and another. For example, objects can inherit characteristics from other objects [13].

In OOP, programmers are enabled to create some modules which are bound not to be changed nor edited when a new type of object is integrated as it supports inheritance. This gives object-oriented programs an ease to modification. For programmers are able to make do with C++, Python, Java, and Smalltalk in Object Oriented paradigm.

The structured programming paradigm proposed that programs could be developed in sensible blocks that make the program more understandable and easier to maintain. The Object Oriented paradigms suggest we should model instructions in a computer program with the data they manipulate and store these as components together. One advantage of doing this is we get reusable software components [14].

5. Object Oriented Modeling in Software Development

Object-oriented modeling (OOM) is a universal standard modeling technique in software development of applications, businesses and system realms through the application of object-oriented paradigm in the developmental process. OOM is a foremost method that is profoundly applied in the Object-Oriented Analysis (OOA) and the Object-Oriented Design (OOD) actions in software development.

Object-oriented modeling typically divides into two aspects of work: the modeling of dynamic behaviors like business processes and use cases, and the modeling of static structures like classes and components. OOA and OOD are the two distinct abstract levels (i.e. the analysis level and the design level) during OOM. The Unified Modeling Language (UML) and SysML are the two popular international standard languages used for object-oriented modeling [15]

OOM are Proficient and effectual in communication as users do not find it difficult to understand spec documents and codes. Visual model diagrams can be more understandable and can allow users and stakeholders to give developers feedback on the appropriate requirements and structure of the system. A key goal of the object-oriented approach is to decrease the "semantic gap" between the system and the real world, and to have the system be constructed using terminology that is almost the same as the stakeholders use in everyday business. Object-oriented modeling is an essential tool to facilitate this [16].

OOM has functional and established abstraction as modeling supports coding. Object-oriented modeling enables this by producing abstract and accessible descriptions of both system requirements and designs, i.e. models that define their essential structures and behaviors like processes and objects, which are important and valuable development assets with higher abstraction levels above concrete and complex source code [17].

6. Object-Oriented Analysis and Design (OOAD)

This is an acceptable software development tool for analyzing and designing of a computer application using the object-oriented programming to ensure a good product quality in a development life cycle. OOAD in modern software engineering is best conducted in an iterative and incremental way. Iteration by iteration, the outputs of OOAD activities, analysis models for OOA and design models for OOD respectively, will be refined and evolve continuously driven by key factors like risks and business value [18].

In the early days of object-oriented technology before the mid-1990s, there were many different competing methodologies for software development and object-oriented modeling, often tied to specific Computer Aided Software Engineering (CASE) tool vendors. No standard notations, consistent terms and process guides were the major concerns at the time, which degraded communication efficiency and lengthened learning curves [19]. In 1994, the Three Amigos of Rational Software started working together to develop the Unified Modeling Language (UML).

Later, together with Philippe Kruchtenand Walker Royce (eldest son of Winston Royce), they have led a successful mission to merge their own methodologies, OMT, OOSE and Booch method, with various insights and experiences from other industry leaders into the Rational Unified Process (RUP), a comprehensive iterative and incremental process guide and framework for learning industry best practices of software development and project management. Since then, the Unified Process family has become probably the most popular

methodology and reference model for object-oriented analysis and design [20].

The software life cycle is usually alienated in several stages from the planning to analysis to design and next to integration after which the system is produced. The most basic stages of the process are analysis and design. The analysis stage is also known as the "requirements acquisition stage"

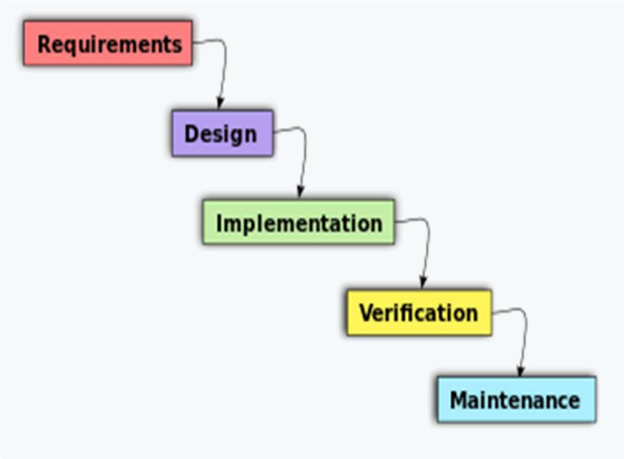

Figure1: Waterfall Model for Software Development.

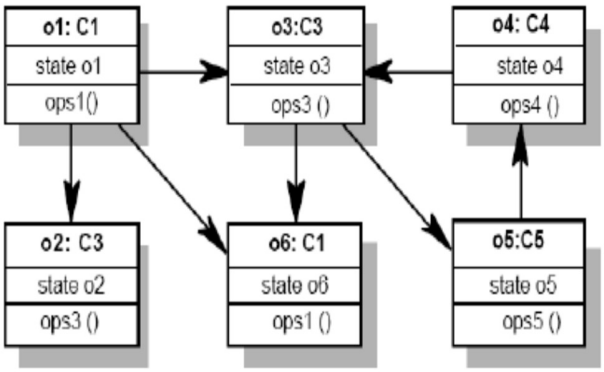

Figure 2: System with Interacting Objects.

7. Role of Object-Oriented Programming in the Implementation of Geographic Information Systems (GIS)

Geographic Information System (GIS) has gained a wide publicity in software development as developers rely extensively to it in program creation. The concept of Object-Oriented Programming (OOP) and its concepts have changed the perception in software development and GIS in common. Geographic Information Systems (GIS) are unique information management systems that keep tab on activities and happenings and their locations. Since things are bound to happen, so there is an importance that those things ought to be kept in tabs so that users would be cognizance to those happenings. For instance the Google weather forecast allows us to know when the weather is cloudy or when there would be slight rain or a heavy rain. It is also important for one to easily locate hospitals in the time of emergency and other services as they are vital to human. These services are attributed to the Geographic Information System GIS is a collection of computer hardware, software, geo-data, etc, for capturing, managing, analyzing, and displaying all forms of geographically referenced information. Of course, everything involves the people who will manage the system, not just the computers and procedures [21].

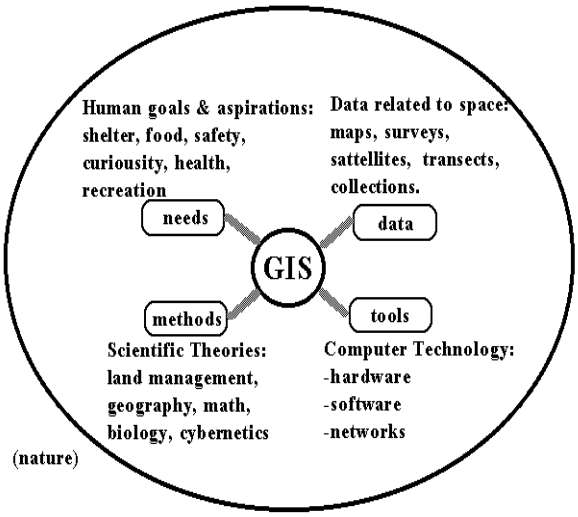

Figure 3: Different segments in a Geographic Information System

The concepts of Object-Oriented Programming Languages (OOPLs) are incorporated for the modeling of GIS software. The GIS models have some appreciable features of classes, objects, inheritance, polymorphism and etcetera. The GIS model is based on the geometry of Points, Lines, Surfaces and the Complexities that are contained in all the attributes of GIS. When we identify surfaces like the river, road, sea, mountain and trees we invariably have identified those remarkable classes of geometry as in the Points, Lines, Surfaces and the Complexities as enumerated.

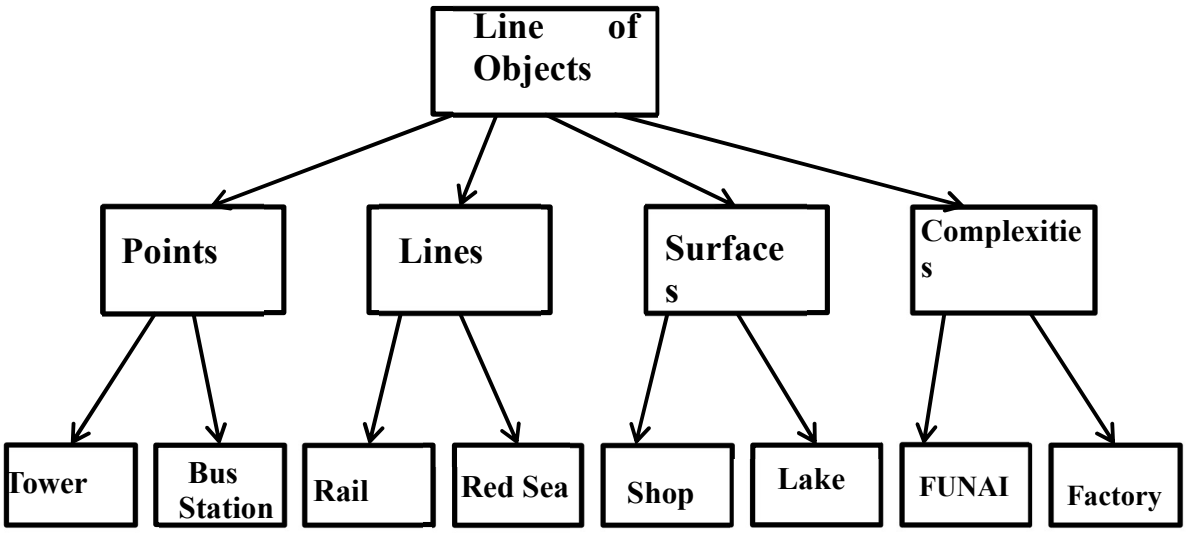

Figure 4: Feature classification

Furthermore, buildings can be specialized into post office, school, town hall and hotel. Some of the feature classes with the same geometric properties can be combined. For example buildings, parks, lake, land can be combined to super-class called surface. Road, river, railroads can be combined to a super-class called line. Train station and electric polls and trees can be combined to a super-class called points. Several features might be aggregated to a complex objects, for instance, some buildings and other features can be grouped to a factory. Each spatial features class is declared as belonging to a super-class of four geometry classes. For instance, the class BUILDING has attributes like building number, owner, date built, street etc. Specialized subclasses of a feature class like HOTEL can have additional attributes and still inherit the common attributes from the super-class (parent) class (BUILDING). For example, HOTEL class might have manger, staff, tax, bed number, and room number as its attribute and still inherit attributes like owner, built date, and building number from the super-class BUILDING [22].

Existent attributes in the real world could be modeled using GIS and its concepts of Object-oriented programming like the classes, objects, aggregation, encapsulation, classification, inheritance, polymorphism and generalization. Illustration of these concepts is shown below; The Object classification: This classification could be described as mapping some objects to a unique class. The GIS model of a Hospital is bound to comprise of the classes BUILDING, DOCTOR, and PATIENT. A particular illustration, such as the building: with building Identification number "Maternity Hospital 1A," is an object of the resultant object type, that is, the exacting object is an instance of the class BUILDING.

■ Object: The object type is assigned operations and attributes. For instance, the class BUILDING could have some attributes as the YearBuilt which is particular for any building. Another example is the hospital DOCTOR may have some attributes to establish the TotalNumberOfOperationMade, and the PATIENT could have an attribute to decide the ReportOfHealingProgress.

Figure 5: Graphical representation for *classes* BUILDING, DOCTOR, and PATIENT

Association: Association correlates some related objects in form of data abstraction to form a superior group of objects. An example of association in GIS domain is neighborhood, which links parcel of land with its adjacent house lots. The detail of a member object is suppressed while the properties of the group object are emphasized. The NEIGHBORHOOD and the ROAD NETWORK are associated by the relationship inside [23].

Figure 6: Graphical representation for Neighborhood with Road Network, Adjacent House lot, and Parcel of Land.

Generalization: In Generalization, classes of an object are grouped with various attributes that are related with one another into a super-class. Super-classes and other subclasses are the concept and abstractions for a given project and do not portray dissimilar objects. For instance, BUILDING could be attributed with subclasses of CLASSROOM, LIBRARY, and LABORATORY.

Figure 7: Graphical representation for a generalization with multiple subclasses of a super-class

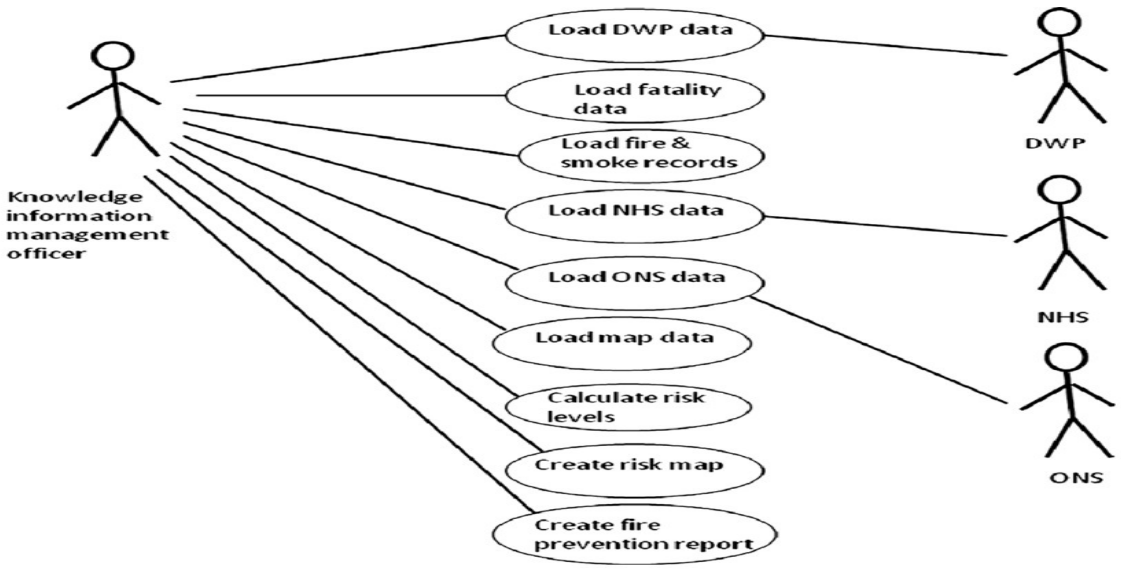

Figure 8: Use case diagram of Geographic Information Systems (GIS)

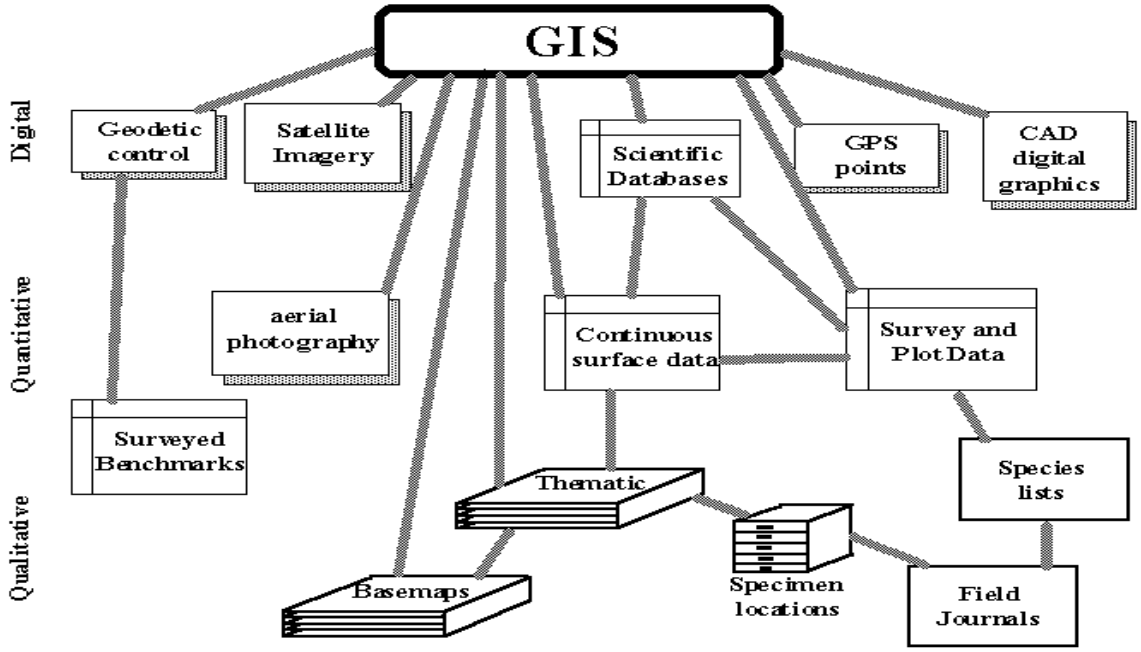

Figure 9: Activity diagram of Geographic Information Systems (GIS)

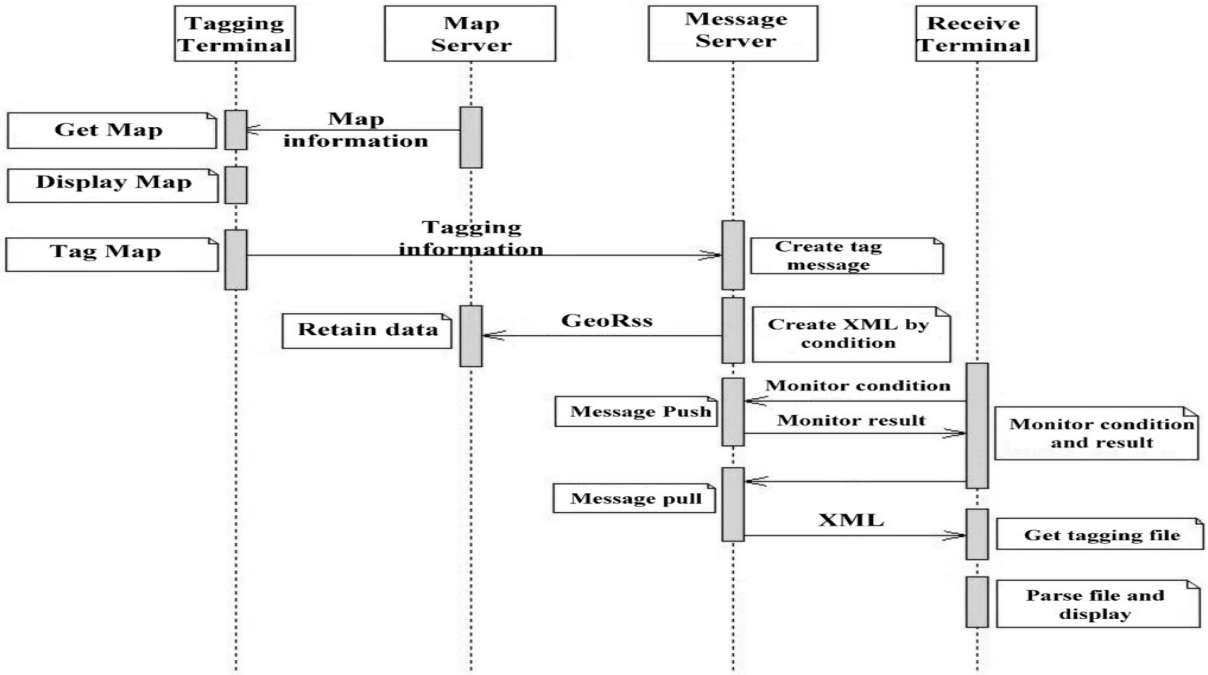

Figure 10: Sequence diagram of Geographic Information Systems (GIS)

8. Conclusion

The research work has drawn-out the essence of Object-Oriented Programming (OOP) in this modern programming milieu. OOP is properly suited as the most appropriate paradigm in the complexities of software development. OOP presents some ways of seeing and solving real-time problems and mustering solutions just like human beings. This work has shown also on how the concepts of OOP could be utilized in modeling of Geographic Information Systems (GIS) software and the use of OOP in the design of features of GIS software and its production.

References

[1] S. Barbey, M. Ammann, and A. Strohmeier. Open issues in testing Object Oriented software. In K. F. (Ed.), editor, ECSQ '94 (European Conference on Software Quality), pages 257–267, vdf Hochschulverlag AG an der ETH Z¨urich, Basel, Switzerland, October 1994.

[2] Simon Kendal. (2009). *Object Oriented Programming using Jav*a, www.bookboon.com, eBook Company, ISBN 978-87-7681-501-1, 14P.

[3] Boronczyk, Timothy (11 June 2009). "What's Wrong with OBJECT ORIENTED PROGRAMMING", zaemis.blogspot.com.

[4] Kindler, E.; Krivy, I. (2011). Object-Oriented Simulation of systems with sophisticated control**.** *International Journal of General Systems*. pp. 313–343.

[5] Simon Kendal. (2009). *Object Oriented Programming using Jav*a, www.bookboon.com, eBook Company, ISBN 978-87-7681-501-1, 21P.

[6] S. Deshpande, Collaboration of Object Oriented Programming and Software Development International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056 Volume: 03 Issue: 09 | Sep-2016 www.irjet.net p-ISSN: 2395-0072 © 2016, IRJET | Impact Factor value: 4.45 | ISO 9001:2008 Certified Journal | Page 524

[7] Simon Kendal. (2009). *Object Oriented Programming using Jav*a, www.bookboon.com, eBook Company, ISBN 978-87-7681-501-1, 16P.

[8] K. Gujar, S. Mishra, P. Kawadkar, "Analysis of Function generation on the basis of Object Oriented paradigm", International Journal of Advanced Computer Research, Vol. 1, No. 1, pp. 91-95, September 2011.

[9] Simon Kendal. (2009). *Object Oriented Programming using Jav*a, www.bookboon.com, eBook Company, ISBN 978-87-7681-501-1, 16P.

[10] Vedpal, N. Chauhan, and H. Kumar. A hierarchical test case prioritization technique for object oriented software. In Contemporary Computing and Informatics (IC3I), 2014 International Conference on. 2014.

[11] Milojkovic, N., et al. Polymorphism in the Spotlight: Studying Its Prevalence in Java and Smalltalk. in Program Comprehension (ICPC), 2015 IEEE 23rd International Conference on. 2015.

[12] Boehm B, "A Spiral Model of Software Development and Enhancement", IEEE Computer, IEEE, 21(5):61-72, May 1988

[13] Meyer, Bertrand (1988). Object-Oriented Software Construction. Cambridge: Prentise Hall International Series in Computer Science. p. 23. ISBN 0-13-629049-3.

[14] Jacobsen, Ivar; Magnus Christerson; Patrik Jonsson; Gunnar Overgaard (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press. pp. 15,199. ISBN 0-201-54435-0.

[15] *Jacobsen, Ivar; Magnus Christerson; Patrik Jonsson; Gunnar Overgaard (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press. pp. 77–79. ISBN 0-201-54435-0.*

[16] *Conallen, Jim (2000). Building Web Applications with UML. Addison Wesley. p. 147. ISBN 0201615770.*

[17] Jacobsen, Ivar; Magnus Christerson; Patrik Jonsson; Gunnar Overgaard (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press. pp. 15,199. ISBN 0-201-54435-0.

[18] *Jacobsen, Ivar; Magnus Christerson; Patrik Jonsson; Gunnar Overgaard (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press. pp. 77–79. ISBN 0-201-54435-0.*

[19] Jacobsen, Ivar; Magnus Christerson; Patrik Jonsson; Gunnar Overgaard (1992). Object Oriented Software Engineering. Addison-Wesley ACM Press. pp. 15,199. ISBN 0-201-54435-0.

[20] Paul A. Longley, Michael F. Goodchild, David J. Maguire, David W. Rhind (2005). Geographic Information Systems and Science.2nd Edition. John Wiley & Sons.

[21] Onu F.U, Uche-Nwachi E, Chigbundu K. E, The Role of Object-Oriented Programming (OOP) in Modeling of Geographic Information Systems (GIS Journal of Mobile Computing & Application (IOSR-JMCA) e-ISSN: 2394-0050, P-ISSN: 2394-0042.Volume 3, Issue 4 (Jul. - Aug. 2016), PP 40-46 www.iosrjournals.org DOI: 10.9790/0050-03044046 www.iosrjournals.org .

[22] Onu F.U, Uche-Nwachi E, Chigbundu K. E, The Role of Object-Oriented Programming (OOP) in Modeling of Geographic Information Systems (GIS Journal of Mobile Computing & Application (IOSR-

JMCA) e-ISSN: 2394-0050, P-ISSN: 2394-0042.Volume 3, Issue 4 (Jul. - Aug. 2016), PP 40-46 www.iosrjournals.org DOI: 10.9790/0050-03044046 www.iosrjournals.org

[23] Onu F.U, Uche-Nwachi E, Chigbundu K. E, The Role of Object-Oriented Programming (OOP) in Modeling of Geographic Information Systems (GIS Journal of Mobile Computing & Application (IOSR-JMCA) e-ISSN: 2394-0050, P-ISSN: 2394-0042.Volume 3, Issue 4 (Jul. - Aug. 2016), PP 40-46 www.iosrjournals.org DOI: 10.9790/0050-03044046 www.iosrjournals.org