Object oriented programming: Concepts, limitations and application trends

Nehul Singh

Student, Department of Computer
Science and Engineering,
IIIT Sricity, Chittoor, INDIA
nehul.s20@iiits.in

Satyendra Singh Chouhan

Assistant Professor, Department of
Computer Science and Engineering,
MNIT Jaipur, INDIA
sschouhan.cse@mnit.ac.in

Karan Verma

Assistant Professor, Department of
Computer Science and Engineering,
NIT Delhi, INDIA
karanverma@nitdelhi.ac.in

Abstract— This article reviews and discusses emerging applications of object-oriented programming languages. A history in brief on development of object-oriented analysis and methods eventually leading for important soft programming techniques are discussed. Emphasis mainly is given to discuss and differentiate popular object-oriented programming languages such as; C++, Python and Java. The pros and cons of these programming languages are also highlighted in view of their futuristic developments and possible application areas.

Keywords— *Object oriented programming, Popular soft-languages, limitations, Current and future trends*

I. INTRODUCTION

Soft programming languages with qualities such as fast computation, easy-to-use, and easy-to-deploy have been attracting user and business communities to adopt them for various applications. Demand and the number of engineers equipped with soft skills are growing since more than 2-decades at a tremendous rate much faster than any other technical skill. The object-oriented analysis and design approaches bound practitioners to think for development of programming languages suitable for applications covering the objects and classes. This development era further led by coding practitioners to introduce various languages, all were classified within a single entity; object-oriented programming (OOP), suitable for building user interfaces— windows, buttons and menus [1]. The OOP language; C++ have been used to develop 2D and 3D interactive software environments for structural analysis by integrated computational approach into graphical user interfaces for Windows-98 and Windows-NT operating systems [2]. It also supports reusable interfaces, multithreading with better error-handling facilities, there by has been preferred to develop automation controllers, data processing, computational chemistry, physiology, biology, machine learning and statistical computing [3-5]. In the year of 1989, utilizing concepts of object oriented methods; Guido Van Rossum introduced an interpreted programming language; Python, which attracted developers to work upon since then. Perhaps, it has become one among other popular programming

languages such as classics Java, C, and C++. Its unique features have been used for applications like desktop-based GUI and Web based applications, games and with emerging areas such as natural language and big data processing [6-7]. James Gosling, 1995 (Sun Microsystems Inc.) invented a class based and concurrent computer programming description; termed as Java, which is an object-oriented and secured programming language. It is a widely used rich technology and has been preferred to develop soft platforms for mobile, enterprise, gaming, business and cross-platform computing applications [8-9]. This article reviews basic concepts and important features of OOPs. Later paragraphs are written to discuss popular OOPs such as C++, Python and Java with their unique features, application areas and limitations.

II. GENERAL CONCEPTS OF OBJECT-ORIENTED PROGRAMMING

Object-oriented programming has been preferred for modeling and to provide solutions for complex tasks with use of objects interactions. In OOPs, objects are created by interrelating variables and functions, whereby variables and functions define properties and behavior respectively for created objects. In OOPs coding, the data and respective functions are bind together in such a way that a specific part of the code can access bonded data with its respective function only. Popular OOPs are Objective C, C++, Perl, Python, JavaScript, Simula, PHP, Smalltalk, Emerald, Object Pascal, Common Lisp, etc. [10-11]. OOP codes are capable to implement real-world entities such as inheritance, polymorphism, data hiding, etc, that's why visualization is easier with OOPs. Some basic concepts mostly used with all types of OOPs are classes, objects, encapsulation, data hiding, inheritance, abstraction, polymorphism, interface and methods. A class is a user-defined data-type, which represent a blueprint for an object in objects-oriented programming language. It is the basic building block of various programming techniques such as C++, Python, Java, etc. A class can have multiple objects which have properties and behaviour that in common for all of them. An object has to be

created to access the members of a class and to allocate memory in the programming.

Fig. 1: Basic concepts and popular applications of OOPs

The encapsulation is used bind the function with the data in a single unit. Specific use of encapsulating to hide the data originates other important concepts in OOPs; data abstraction and data hiding. Figure 1 summarizes important basic concepts and popular applications of OOPs.

Polymorphism in OOPs defines the efficiency to perform the task with use of multiple forms. For example; the operator and function overloading are methods by which polymorphism can be achieved like in C++. Operator overloading execute multiple behavior for different instances, while function overloading performs multiple operation for similar cases. The inheritance concept in OOPs is used for reuse of the codes. Inheritance makes a class (termed as child or sub-class) capable to derives characteristics of other class (base or parent class). Based on uses, inheritance can be classified as single, multiple, multi-level, hierarchical and hybrid. The terms; data abstraction and data hiding are important concepts of the object-oriented programming, which are used to hide the data and to allow the user to access relevant data only. This is accomplished by use of class or header file in C++ programming. Past three decades, many programming languages that use concepts of OOPs have been in use for various applications due to their unique features such as C++ for banking, cloud and distributed systems, Python for real-time and expert systems, artificial intelligence, client-server systems, etc. Figure 2 depicts important features of object-oriented programming techniques. Important advantages of OOP languages can be summarized as flexible and better-quality coding techniques, easy and effective troubleshooting, code reuse facilities and enhanced productivity [10-11]. Prior to popularity of OOPs uses, coding methods such as C, Pascal and BASIC were in practice for most applications, which are termed as the procedure-oriented programming (POP). With POP approach, the program is procedural in nature with functions to perform different tasks [12-13].

Fig. 2: Important features of OOPs

The procedural or structured codes work well for small tasks only, as it is hard to implement debugging, if the coding length is large and complex. Whereas code re-usability features of OOPs make it clearer and less complex coding technique. However there have been some limitations with OOP coding techniques such as complexity control [14]. Depending upon the applications, the OOP codes are complex in nature and posses increasing complexity trends as and when these codes are updates to cover future requirements. Controlling ever-increasing complexity issues may results into expensive procedure with increased probability of failure. Sometimes OOP code developers face issues due to crosscutting concern, code scattering and tangling for their optimum implementation [14]. Uses of OOP codes limited due to requisite vertiginous methods, large size and slow processing as compare to procedure-based programs. That's why; many problems can be solved by other programming techniques more conveniently than OOPs programs.

III. DISCUSSIONS ON C++, PYTHON AND JAVA

The C++ is a high-level and multi-paradigm programming language created by merging features of C and Simula 67 [15]. Over the years, this coding technique has been evolved to facilitate low-level memory manipulation in addition to the generic and functional features [5]. Being a fast and compiled programming language, it has gained popularity for its control feature to manage the memory spaces efficiently. However it doesn't support dynamic memory allocation, garbage pickup and threads built-in facilities. Due its high-level programming nature, C++ codes usually are complex and less flexible [16]. As compares to Java, the C++ is not a impressive programming language, however it is easier to manage a C++ code with less memory. Other important pros and cons of C++ programming language can be checked from figure 3. The Python is a high-level, interpreted programming language, suitable write clear and logical code for a wide range of projects.

Fig. 3: Important advantages and disadvantages of C++, Python and Java programming languages

In recent time, uses of Python have achieved a tremendous growth due to its availability as an open-source tool which promote easy and frequent readership. In recent times, its uses have been increased tremendously for data analytical problems, artificial and machine learning frameworks [17].

The dynamic nature of Python makes it a slow language due to higher execution time, hence not suitable for faster applications. Also, it is not suitable for memory optimized processes, due to requisite large memory spaces. It is also vulnerable for runtime errors and faces problems while accessing the huge data bases, thereby not suitable for client-side and big enterprises applications.

Java is a dynamic programming language and has been in use to develop large desktop applications due its platform independency. Java has been preferred as a high-performance coding for distributed internet environment and high-performance computing, if used with Just-in-Time compilers [18]. The multithreading feature enables Java programming to perform many tasks simultaneously. It is also suitable to avoid error-prone situations with use of time error and runtime checking compilation process. Java developers use public-key encryption for authentication purposes and use its secure features to define virus-free, tamper-free systems. Java programs are capable to handle vast amount of run-time information requisite for smooth running of interactive applications. Interpretation of the code at the machine level makes the Java programming slow as compared to native languages such as C or C++, due to time needed for compilation, abstraction and to interpret at the machine levels. The Java code also consumes more memory space and leads to complex structuring. Lack of back-up facilities and requisite long and complex sentences are other limitations for Java

coding techniques. The figure 3 summarizes important pros and cons of C++, Python and Java.

IV. CHALLENGES AND FUTURISTIC APPROACHES

For futuristic applications, developers working with C++ have to chalk out plans to build appropriate libraries to provide flexible interfaces using generic programming skills. The use of C++ for distributed computational methods and at the level of individual components is expected in near future. New codes with the concept of OOPs have to be developed, which will be platform-independent for their best use and to cover a wider range of applications. Variation in coding style has to be adopted by the developers to generate component-oriented software with support of investment, better design criterion and disciplined programming steps. There are many challenges, which needed to address to utilize object-oriented programming for real-world applications [19-23].

- Relational database: For most of enterprises, data is stored in a relational database pattern, which is suitable for procedural but not with the object-oriented programs. However, OOPs developer have been using functions such as; <insert>, <search>, <modify> and <delete> to accommodate relational databases.
- Encapsulation of procedures: It is expected, that a procedure should be encapsulate properly, while defining the object. However, in a situation, whereby a procedure may appear with two or more objects; the developers should use procedure encapsulation in a way that all such objects can inherit it from each other. Although this will lead to a complex code,

eventually will pose difficulties for the application programmer to manage the objects. Therefore, proper planning and thought process would be needed from a developer to have either a stand-alone procedure-based object or to work simultaneously with objects having a common encapsulated procedure.

- Object identification: With ever-increasing demands to write object-oriented codes on real-world things, identification and creation of requisite objects have also been a challenging task. Until necessary, keeping the object undivided would indeed help to maintain and to upgrade the code for its future use. Placing attributes and member procedures with objects will be a better choice to distinct them with each other. Restricting the numbers of objects in a code is recommended, as it is easier to maintain code which uses few objects.
- Use of multilevel inheritance: In recent time, multilevel inheritance has been created for optimum use of OOP codes, however on the cost of complex hierarchy, which is more difficult to handle. Perhaps regular updates in the code will be more useful and easier to implement rather than creating unnecessary inheritance levels. Depending upon the requirements, developer may choose to keep the inheritance levels up to three or less until it is very important to have few more.

V. CONCLUSION

In this article, important features and applications areas of various OOP languages have been discussed in brief with limitations in present scenario. Theories covered in the article also anticipated that OOPs existence and popularity will be continued due to their flexible, efficient and non-proprietary nature. The object-oriented programming shall remain an important coding style for decades to come and will be able to meet futuristic challenges. Acceptance of OOP codes will remain high and the demand will drive by internet networks, World Wide Web and telecom interfaces in near future as well, however variation in OOP coding has to maintain as per the vendor requirements and to compete with new soft languages. Possible competition with other functional languages may compel OOP developers to introduce functional features in their present coding style. Unique and valuable aspects of OOPs keep it omnipresent even when it has its drawbacks and therefore will be use for various applications in future as well.

ACKNOWLEDGMENT

We pay our gratitude to all contributors, whose works have enlightened us with knowledge to prepare this article. We also acknowledge the Pandemic Covid-19 Era, subsequent lockdown and virtual study mode to motivate us for this study.

REFERENCES

- [1] Zheng, Li Sun "Object-oriented method for graphical user interface design in a distributed system environment". Graduate Student Theses, Dissertations, & Professional Papers. 5108, 1993.
- [2] Marco L. Bittencourt, Claudio A. C. Silva, Raul A. Feijoo, "Object-oriented Programming Applied to the Development of Structural Analysis and Optimization Software". J. Braz. Soc. Mech. Sci. 23, 2001.
- [3] Russell T. Berman "Using C++ to Write Automation Controller Software. JALA" Journal of the Association for Laboratory Automation, 12(1):12-16, 2007.
- [4] Mirams GR, Arthurs CJ, Bernabeu MO, Bordas R, Cooper J, Corrias A, et al. "Chaste: An Open Source C++ Library for Computational Physiology and Biology" PLoS Comput Biol 9(3): e1002970, 2013
- [5] Rassokhin, D. "The C++ programming language in cheminformatics and computational chemistry". J Cheminform 12, 10, 2020.
- [6] Charles Dierbach "Python as a first programming language" Journal of Computing Sciences in Colleges, 29(3): 73, 2014.
- [7] Charles Severance "Guido van Rossum: The Early Years of Python", Computer, 48(2): 7-9, 2015.
- [8] O'Regan G. "James Gosling. In: Giants of Computing" Springer, London, 2013.
- [9] Shubham Singh, Hansraj Yadav, Amarjeet Singh "Overview of Java Programming" International Journal of Progressive Research in Science and Engineering, 1(2):75-77 2020.
- [10] Andrew P. Black "Object-oriented programming: Some history, and challenges for the next fifty years" Information and Computation, 231:3-20, 2013.
- [11] Stroustrup B. "What is "Object-Oriented Programming"? In: Bézivin J., Hullot JM., Cointe P., Lieberman H. (eds) ECOOP' 87 European Conference on Object-Oriented Programming. ECOOP 1987. Lecture Notes in Computer Science, vol 276. Springer, Berlin, Heidelberg, 1987.
- [12] White, G., Sivitanides, M. "Cognitive Differences between Procedural Programming and Object Oriented Programming" Inf Technol Manage 6,333-350, 2005.
- [13] Farooq MS, Khan SA, Ahmad F, Islam S, Abid A "An Evaluation Framework and Comparative Analysis of the Widely Used First Programming Languages" PLoS ONE 9(2): e88941, 2014.
- [14] Massimiliano Dessi, "Spring 2.5 Aspect Oriented Programming", Packt Publication, pages 332, ISBN: 9781847194022, 2009.
- [15] Jiang Guo "C and C++, Editor(s): Hossein Bidgoli, Encyclopedia of Information Systems", Elsevier, Pages 99-111, ISBN 9780122272400, 2003.
- [16] Ian Joyner "C++?? A Critique of C++ and Programming and Language Trends of the 1990s" 3rd Edition, pages 60, 1996.
- [17] Nguyen, G., Dlugolinsky, S., Bobák, M. et al "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: asurvey" Artif Intell Rev 52, 77-124, 2019.
- [18] Aidan Fries "The use of Java in large scientific applications in HPC environments" PhD thesis, submitted to the University of Barcelona, pages 223, 2012.
- [19] Sikha Bagui "Achievements and Weaknesses of Object-Oriented Databases", in Journal of Object Technology, vol. 2, no. 4, July-August 2003, pp.29-41.
- [20] Lim Tong Ming, Lee Sai Peck, "Extended Object Languages for the Extolware Persistence Framework" Issues in Informing Science and Information Technology, pages: 407-416, 2004.
- [21] Johanna Wenny Rahayu, David Taniar, Eric Pardede, "Object Oriented Oracle" IRM Press, pages: 326, ISBN 1-59140-608-0, 2006.
- [22] Frank Stajano "A Gentle Introduction to Relational and Object Oriented Databases. ORL Technical Report TR-98-2, pages: 37, 1998.
- [23] M. Aniche, J. Yoder and F. Kon, "Current Challenges in Practical Object-Oriented Software Design," 2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER), 2019, pp. 113-116.