AN APPROACH TO STUDY COMPARISON & ANALYSIS OF OBJECT ORIENTED PROGRAMMING LANGUAGES

Tejenderkaur H. Sandhu

Master of Engineering (M.E.) Scholar
Department of Computer Science & Engineering,
Sant Gadge Baba Amravati University, Amravati, India

Abstract : Object-oriented programming is one of the important programming structure of our times. From the time it was brought into existence by Simula, object-oriented programming has seen wide acceptance. Object-oriented programming languages (OOPLs) have certain special features such as classes, inheritance, information hiding (encapsulation), and dynamic binding (polymorphism). There is an enormous ways available to implement these concepts, and there is no general agreement presented on how these concepts must be interpreted.

Keywords **– Object Oriented Language, Criteria, OOP Concepts**

I. INTRODUCTION

Programming language is one of the most important areas for software developers. The three main types of them are functional, structural and object oriented. Studying all of them to develop applications is almost impossible but having a comparison and analysis of some of them can provide a means to develop the application. There is no fixed criterion on how these languages should be evaluated; one can obtain the criteria as per its need. There is not a completely perfect language, every one of them have their advantages and disadvantages and work very reliably or unreliably in certain areas and platforms. The choice of selection is completely depended on user's requirements of function, software and hardware requirements.

In the class-based object-oriented programming structure, "object" refers to an instance of a class where the object is a set of variables, functions, and data structures. A clear understanding of OOPs concepts can help in implying the ideas when designing an application that how we can design an application and what language would be best for it. Object Oriented programming is a programming style which is consists of concepts like class, object, Inheritance, Encapsulation, Abstraction, Polymorphism.

II. THE PURPOSE OF THIS STUDY

The purpose of this study is to achieve the fact that there are no fixed or defined set of procedures or evaluation criteria for comparing any programming language which each other. We can define our own criteria with verified results, calculations and observation for analyzing the difference between these languages.

III. ANALYSIS OF PROBLEM

Since there are hundreds of programming languages existing nowadays, we can compare and analyze the efficient one. We can classify the representative characteristics of languages and make a broader view on them according to some certain criteria. Thus our research problem is aiming to compare and contrast object oriented languages according to certain characteristics with the purpose of determining the suitability and applicability of the languages for each criterion, distinguish them their pros and cons, evaluate and explore the related features on those languages, illustrate the best language usage for evaluated characteristics and also get the details of resources required for a particular language. Our objectives of this study are:

1. To find the best language for a particular feature with respect to parameters.
2. To evaluate / find the best programming language in according to the need of development i.e. Desktop Application, Web Application, Application according to need.
3. To define broader view of Object Oriented Languages.

IV. PROPOSED STRATGEY

We will be using different object oriented programming languages such as C++, Java, C#, Python & Ruby for comparison and analysis of the efficient one. We are determining the efficiency of the language on the basis of various characteristics. These characteristics will give us comparison and analysis on the languages we are using. In order to better compare the overall features of the languages under study, different other criteria will also be considered to illustrate how some languages outperforms others in a given criterion and the reason behind that. For example one criteria can be comparing execution time, memory usage, object size, number of lines, reliability etc.

1) C++

C++ is a general purpose programming language. Designed by Bjarne Stroustrup in 1979. B. Stroustrup during his PhD thesis found that Simula has important features that will help in building large software. The motivation was to acquire distributed computing in the UNIX operating system in the AT&T Bell Labs. Therefore, the idea to advancing the existing C language with Simula-like features was considered. C was taken as it is a general-purpose language, fast, portable, and widely used. In 1983, it was named as C++ due to advancement in C.

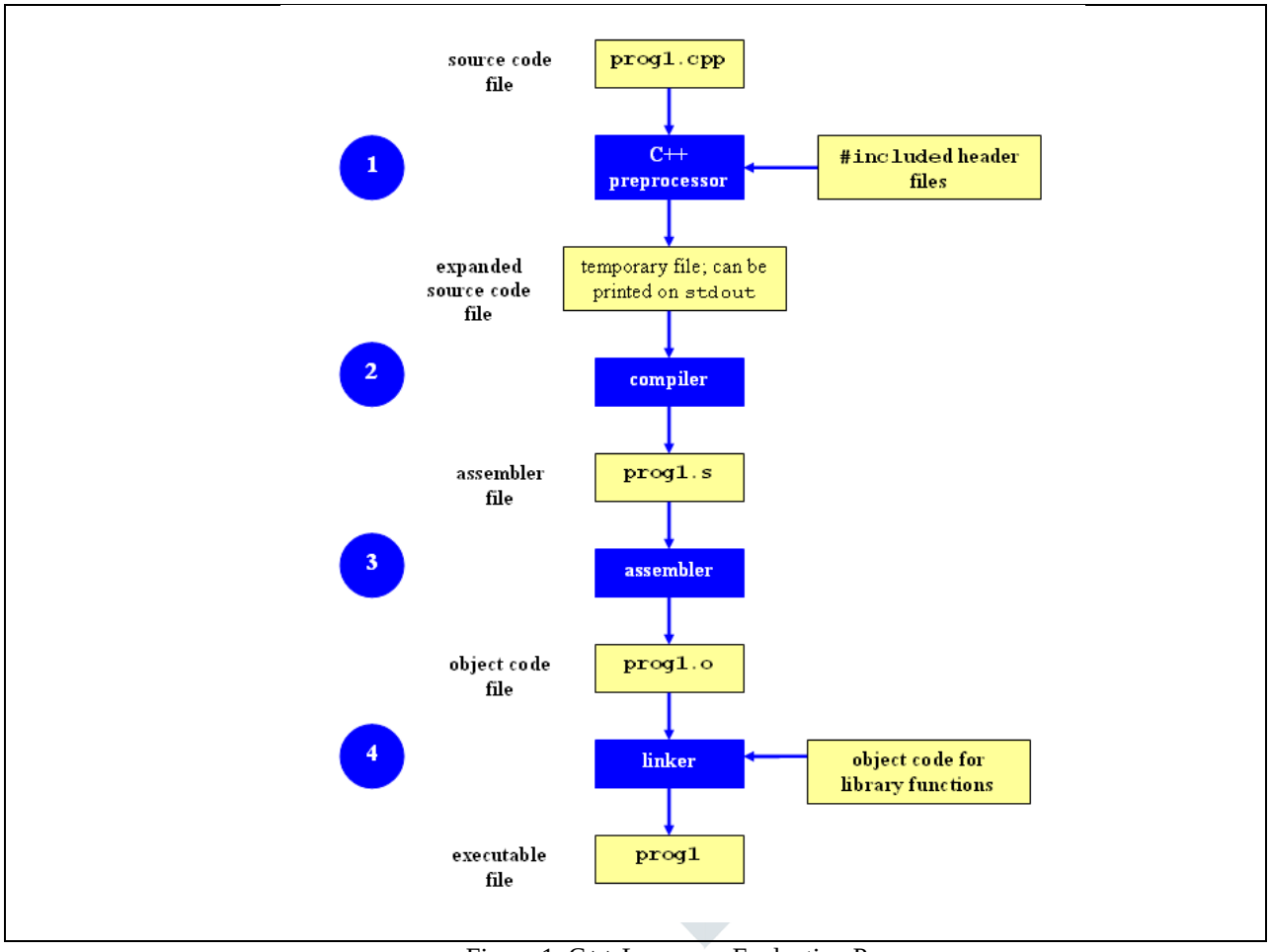

Figure 1: C++ Language Evaluation Process

2) Java

Java is a predominantly static typed programming language (though it also supports dynamic typing for some OOPs concepts like polymorphism) developed by James Gosling, Mike Sheridan, and Patrick Naughton in the year June 1991 at Sun Microsystems. Many of the paradigms were considered from C and C++ programming languages. Though it had been heavily influenced by C and C++, it was different in its own ways. It eliminated pointers as its designers thought that developers square measure mistreatment pointers the manner it had been not meant to be. Also, it born the headache of managing memory because it brought in automatic memory management with the introduction of trash collection. Another feature that created Java to face out of its peers was that it favoured WORA philosophy (i.e. Write Once Run Anywhere) which launched the Java Virtual Machine.

As a result, all a system needs to have is a JVM to run a java code.

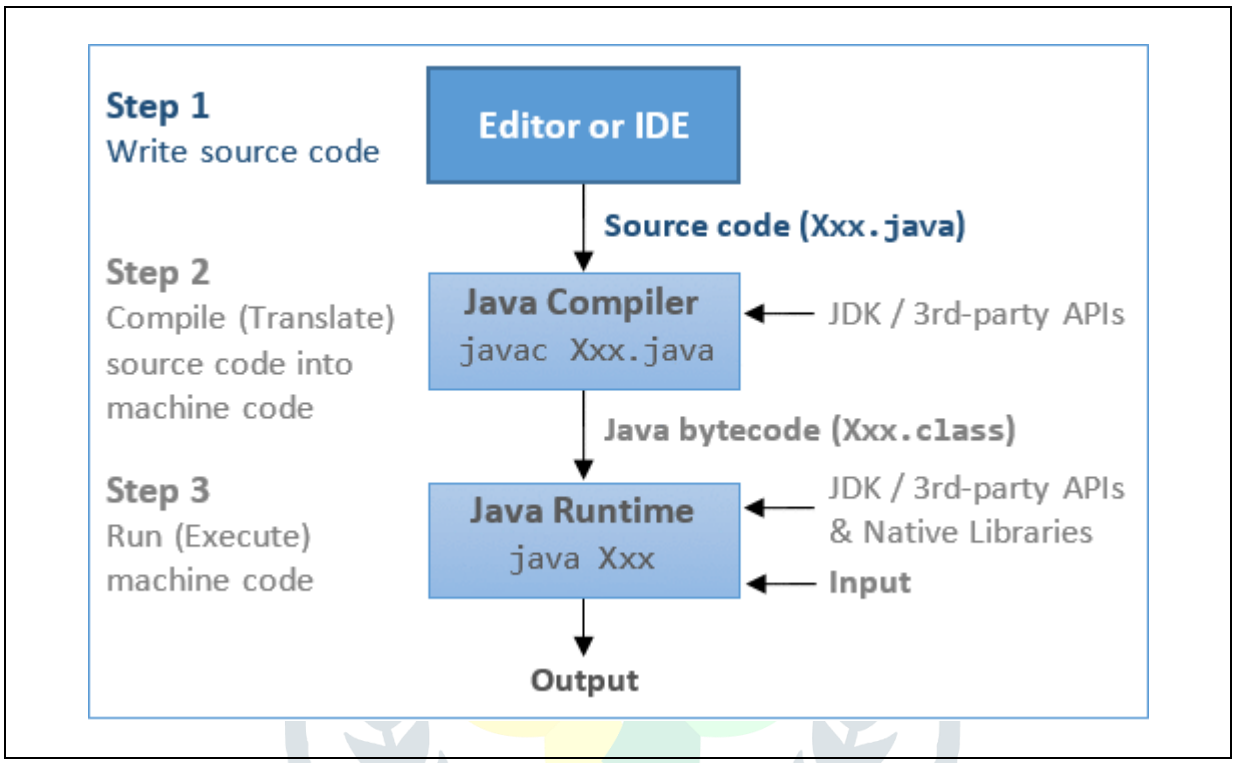

Figure 2: Java Language Evaluation Process

3) C#

C# programming language was introduced to the world by Microsoft at the same time as the .Net platform, and the different versions of C# were also introduced in parallel with the Microsoft .Net new versions. C# is a modern type-safe multi-paradigm programming language that became very popular and widely spread in the development field because of its simplicity, flexibility and productivity. The fact that C# benefits from several key features and powerful ideas found in different programming languages with different programming paradigms result in a programming language that can be used to develop applications with the clean syntax of Java, the simplicity of Visual Basic and the power and expressiveness of C and C++ programming languages.

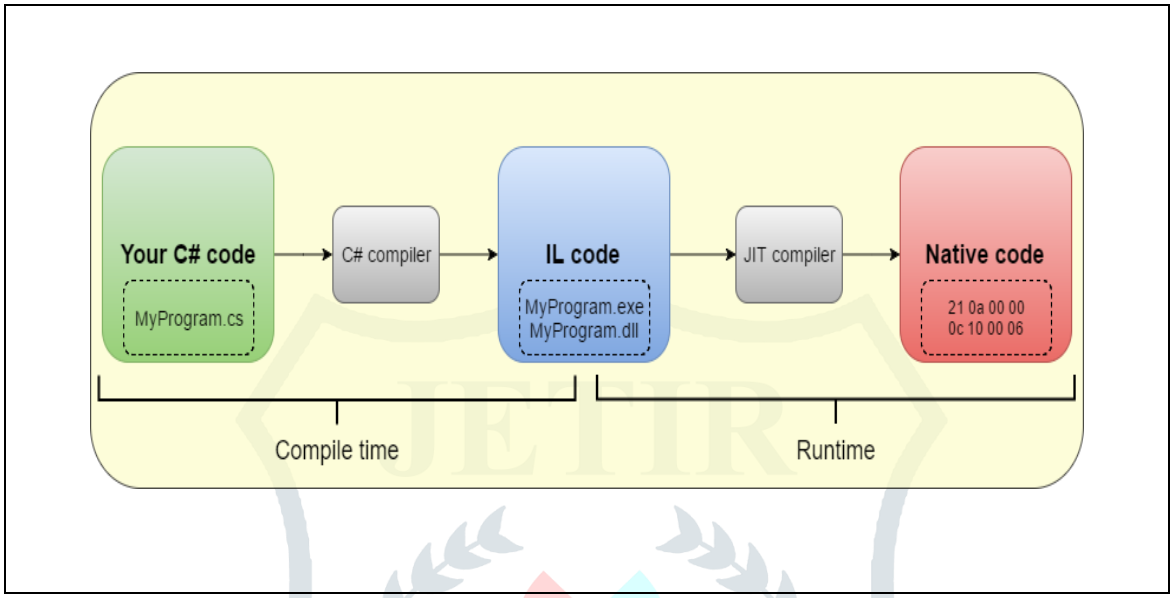

Figure 3: C# Language Evaluation Process

4) Python

Python is preponderantly a dynamic typewritten programing language that was initiated by Guido van Rossum within the year 1989. The major style philosophy that was given a lot of importance was the readability of the code and expressing an inspiration in fewer lines of code instead of the wordy method of expressing things as in C++ and Java. The other style philosophy that was price mentioning was that, there ought to be continually one method and one obvious thanks to categorical a given task that is contradictory to other languages such as C++, Perl etc. Python compiles to an intermediary code and this in turn is interpreted by the Python Runtime Environment to the Native Machine Code.

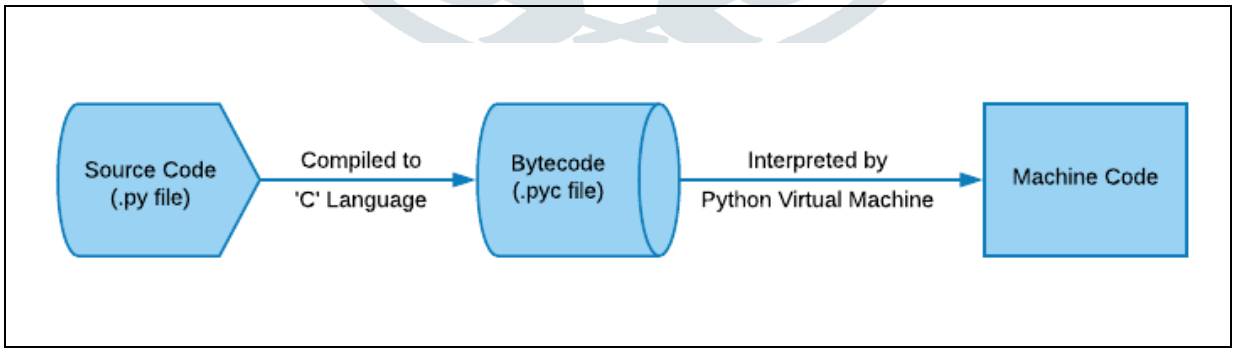

Figure 4: Python Language Evaluation Process

5) Ruby

Ruby was created by Yukihiro Matsumoto, or "Matz", in Japan within the middle 1990's. It was designed for engineer productivity with the concept that programming ought to be fun for programmers. It emphasizes the need for package to be understood by humans initial and computers second. Ruby continues to realize quality for its use in internet application development. The Ruby on Rails framework, engineered with the Ruby language by David Heinemeier Hansson, introduced many folks to the fun of programming in Ruby. Ruby encompasses a vivacious community that's validating for beginners and smitten by manufacturing high-quality code. Ruby has features that are similar to those of Smalltalk, Perl, and Python. Perl, Python, and Smalltalk are scripting languages. Smalltalk is a true object-oriented language. Ruby, like Smalltalk, is a perfect object-oriented language. Using Ruby syntax is far easier than Smalltalk syntax.

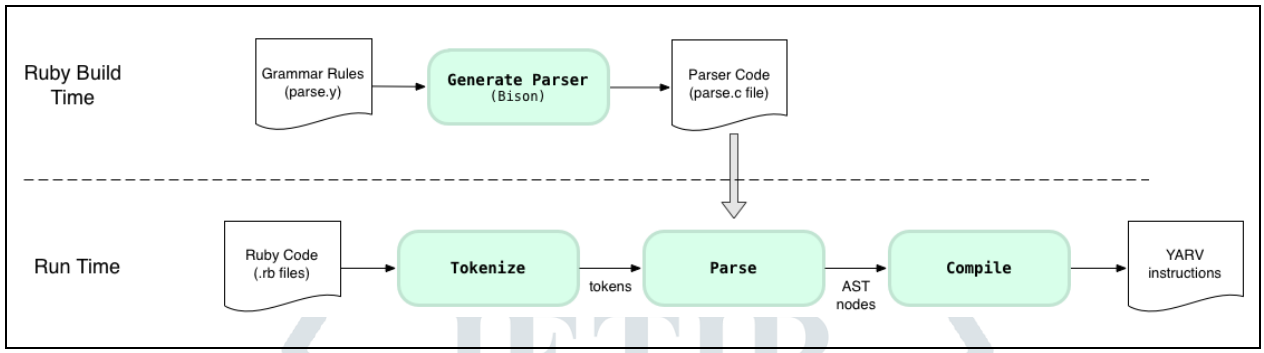

Figure 5: Ruby Language Evaluation Process

V. CONCLUSION

Object-oriented programming languages are used worldwide on many alternative projects and applications. Mastery of the object-oriented paradigm has become an essential part of any programmer's careers. The key features of the object-oriented paradigm (abstraction, encapsulation, inheritance, and polymorphism) have different flavors in the various OOPLs available to the users. There is still lot of work to be done not only to reach a common representation for these crucial features of OOPLs, but also to find appropriate ways to implement features like inheritance and polymorphism to avoid misuse.

Studying all of them to develop applications is almost impossible but having a comparison and analysis of some of them can provide a means to develop the application. There is no fixed criterion on how these languages should be evaluated; one can obtain the criteria as per its need. There is not a completely perfect language, every one of them have their advantages and disadvantages and work very reliably or unreliably in certain areas and platforms. The choice of selection is completely depended on user's requirements of function, software and hardware requirements.

VI. REFERENCES

[1] Maya Hristakeva, RadhaKrishna Vuppala "A Survey of Object Oriented Programming Languages", pp 1-23, Univ. of California, Santa Cruz.
[2] O. Nierstrasz "A Survey of Object-Oriented Concepts", pp 3–22, 1989.
[3] D.J. Armstrong "The quarks of object-oriented development" Communications of the ACM, 49(2):123–128, 2006.
[4] Sleiman Rabah, Jiang Li, Mingzhi Liu, Yuanwei Lai "A Comparative studies of programming languages", pp 1-139, 2010.
[5] Robert Henderson, Benjamin Zorn "A Comparison of Object-oriented Programming in Four Modern Languages", pp 1077-1095, November 1994.