Flexible Platform for Programming Languages: Effective and Efficient?

A programming language is a system of signs and communication from the user to the computer system. Similarly, a flexible programming language (or also known as "universal") is defined as a programming language that can cover all the quality requirements and is probably the highest level and latest generation of programming. It is adjustable and can be easily modified. Also, it is appropriate for all aspects, applications, and paradigms needed to attain.

As mentioned in the extended definition essay, it is absolutely possible to create a flexible programming language although there are three reasons why it has not been done. These three reasons are the effectivity, efficiency, and programmers' views. Firstly, the effectivity is showing how successful the programming language works. Secondly, even if the programming language is effective, it can be inefficient. Efficiency is depicting how good and satisfactory the quality the programming language performs. Lastly, programmers' views and perceptions regarding this issue is a minor reason. This is because there is no reassurance if each and every programmer would agree to bring a universal programming language to existence. Flexibility in programming means having the flexibility in modifying the source code for design and functions. It shows the
possibilities the programmer can have for revisions – the more he can edit depicts that it is more flexible that depends on his knowledge.

Up to present time, there has been no reported programming language wherein it possesses all aspects and requirements needed to cover and classify itself as a universal programming language. Opinionated by a programmer, there have been “most flexible programming languages” in the different paradigms they. She says that Smalltalk is the most flexible programming language for object-oriented models, Perl and Python for imperative, Scheme for functional, and Prolog for logical. Another programmer says that among the existing ones, Assembly is one of the most flexible but he also admires how Ruby is universal-like along with its modern qualities, clean syntax, real object system, and dynamism (“World’s Most Flexible Programming Language”, 2001).

An article written by John Pavlus on 2012 says that researches have showed that Perl is not the most outstanding programming language anymore – because of its complexity and difficulties in editing it source code. Novices were tested to try out two programming languages – Perl and Quorom (uses more characters yet acquires more simplicity and applies more comprehension to users). The result that was attained is that the novices got to work with programs more using the Quorom language. The author and his team asked an expert named Andreas Stefik how come this happened – it’s because programmers get to understand languages more that has more qualities and terms than those of simple ones because “there’s a potential danger of removing features that experts need,” says Stefik. Emphasizing on his statement, it is evident that a simpler and a flexible programming language will not assure efficiency in all qualities –
possible to perform in each of them but does not guarantee great quality in the outcome.

Newer programming languages could have been made better and more efficient and effective but then not every programmer’s opinion is asked about usability, design and flexibility because a committee or a group of programming experts create new languages that get to be offered once completed. Although a platform and interface flexible for every aspect has not been made, a new programming language for parallelism has been created – called Parasail (Parallel Specification and Implementation Language) that can map multicore, manycore, heterogeneous, or distributed architectures which makes it “flexible” for the parallelism feature.

Effectivity means being operative and successful at the same time. On the other hand, efficiency means having the orderly, productive, and organized system. These two characteristics are in question whether they can be acquired or not. It is apparent that programmers and users aim to work with successful and excellent programs and applications but the two attributes are in the way because there is absolutely no assurance if they would be efficient and effective in a great manner. An example of this issue would be the Swiss knife. It is universal and efficient for many purposes yet not so effective in the kitchen. Although, it is absolutely possible as well to create a universal/flexible platform for programming languages.

