2.1.3 Programming Paradigm

Return to DIDO CLI Background

Programming paradigms are a way to classify programming languages based on their features and characteristics. For example, is the language imperative or declarative in nature.

- *An **Imperative Language** uses a sequence of statements to determine how to reach a certain goal. These statements are said to change the state of the program as each one is executed in turn.* [1)]
- ***Declarative languages**, also called **Non-Procedural** or very high level, are programming languages in which (ideally) a program specifies what is to be done rather than how to do it. In such languages there is less difference between the specification of a program and its implementation than in the procedural languages described so far.*[2)]

An import aspect of any programming language is to determine the programming paradigm(s) features needed to properly and meet the requirements of the language. The DIDO-CLI is no exception. In order to discuss the programming paradigm for DIDO-CLI, a brief overview of the kinds of (i.e., classification) of some of the most common paradigms is presented.

Some paradigms are place a lot of emphasis on the implications for the execution model of the language, such as allowing side effects, or whether the sequence of operations is defined by the execution model. Other paradigms are concerned mainly with the way that code is organized, such as grouping a code into units along with the state that is modified by the code. Yet others are concerned mainly with the style of syntax and grammar.

There are many kinds of Programming Paradigms[3)]. Only those that are considered relevant to the DIDO-CLI are reviewed here.

- Procedural Programming
- Object Oriented Programming
- Functional Programming

The DIDO-CLI is expected to be a hybrid of these paradigms.

- ProcPro
- oopPro
- FuncPro
- HiBrPro

2.1.1.1 Hybrid of Functional and Procedural Languages

Return to Top

> *All programming paradigms have their benefits to both education and ability. Functional languages*

> *historically have been very notable in the world of scientific computing. Of course, taking a list of the most popular languages for scientific computing today,* **it would be obvious that they are all multi-paradigm**. *Object-oriented languages also have their fair share of great applications. Software development, game development, and graphics programming are all great examples of where object-oriented programming is a great approach to take.*
>
> *The biggest note one can take from all of this information is that the future of software and programming language is multi-paradigm. It is unlikely that anyone will be creating a purely functional or object-oriented programming language anytime soon. If you ask me, this isn't such a bad thing, as there are weaknesses and strengths to every programming approach that you take, and a lot of true optimization is performing tests to see which methodology is more efficient or better than the other overall. This also puts a bigger thumbtack into the idea that everyone should know multiple languages from multiple paradigms. With the paradigms merging using the power of generics, it is never known when one might run into a programming concept from an entirely different programming language!*[4)]

In keeping with Boundreau conclusion, *"it would be obvious that they are all multi-paradigm"*, the DIDO-CLI is best served using a hybrid programming paradigm approach. Where a Functional Paradigm makes sense, it should be used and followed. Afterall, two very successful CLIs, i.e., POSIX and Structured Query Language (SQL), have demonstrated quite a bit of success using a Functional approach. However, both of these languages also have embraced the Procedural Paradigm also:

- SQL has Stored Procedures and triggers
- POSIX has adopted the use of scripts

In addition, Oracle RDBMS, PostgreSQL and MySQL have included support of Object Oriented Programming[5)6)7)].

1)
user2102654, Stackoverflow, What is difference between functional and imperative programming languages?, 24 July 2013, Accessed 12 April 2021, https://stackoverflow.com/questions/17826380/what-is-difference-between-functional-and-imperative-programming-languages

2)
Encyclopedia Britannica, Accessed: 12 April 2021, https://www.britannica.com/technology/computer-programming-language/Visual-Basic#ref849838

3)
GeeksForGeeks, Difference between Procedural and Non-Procedural language, Accessed: 15 April 2021, https://www.geeksforgeeks.org/difference-between-procedural-and-non-procedural-language/

4)
Emmett Boudreau, What Is A Programming Paradigm?, 16 October 2020, TowardsDataScience, Accessed 15 April 2021, https://towardsdatascience.com/what-is-a-programming-paradigm-1259362673c2

5)
TutorialsPoint, PL/SQL - Object Oriented, Accessed: 15 April 2021, https://www.tutorialspoint.com/plsql/plsql_object_oriented.htm#:~:text=PL%2FSQL%20allows%20defining%20an,and%20methods%20for%20operating%20it.

6)
tapoueh.org, Object Relational Database Management System, 22 March 2018, Accessed: 15 April 2021,

https://tapoueh.org/blog/2018/03/object-relational-database-management-system/
7)

Nick Long, Does MYSQL Support Object Oriented Database?, StackOverflow, 21 August 2013, Accessed: 15 April 2021, https://stackoverflow.com/questions/7901907/does-mysql-support-object-oriented-database