Review of the NCD 46 (2025), 41–46

Dušan Tošić
Faculty of mathematics, University of Belgrade

CLASSIFICATION OF COMPUTER LANGUAGES USED FOR DIGITALIZATION

Abstract. The paper proposes a classification that includes all computer languages. The connection of each class with digitalization was considered. Thus, we can distinguish two large classes: Programming Languages and Domain Specific Language (DSLs). In the first class, two subclasses: low-level languages and high-level languages are further distinguished. Since high-level languages are the most numerous, we paid special attention and classify them into classes based on the dominant programming paradigm that they support. In the DSLs class (which contains a large number of diverse languages), we singled out markup and query languages because of their prevalence and importance for digitalization. The third subclass, within DSLs, consists of all other DSLs.

Keywords: classification, computer languages,digitalization.

1. Introduction

Modern computers are tightly coupled with a variety of languages ranging from machine language (built on a binary alphabet) to natural human languages. A large number of different computer languages have been created so far. So, it is useful to classify them somehow. There are various classifications of computer languages ([1], [2], [4], [5]). Thus, all computer languages can be divided into 2 classes: programming languages and domain-specific languages (Fig. 1). It is not easy to make a strict boundary between these classes. For example, the XSLT language is used to operate with XML data, i.e. it has the role of a programming language, but it is used within the framework of XML technology and therefore belongs to the class DSLs. Similarly, the UML language is primarily used for modeling, but can also take on the role of a programming language when used for a specific platform. Therefore, it is understandable that some authors classify these languages as programming moreover, we will make difference. Within programming languages, we will distinguish between: low-level languages and high-level languages. This division is accepted by almost all authors. Among the most used (most well-known) computer languages are high-level programming languages. As there are many of these languages [3], there is a need to classify them separately. They are often divided into procedural and declarative languages. However, this division is neither precise (most declarative languages contain procedural elements), nor comprehensive. A classification of high-level programming languages can be made based on the dominant programming paradigm that the language supports. However, some programming languages support two or more programming paradigms (for example, Python) so even this classification is not precise enough. Meanwhile, if the dominant paradigm is chosen, we can recognize 7 classes of high programming languages: procedural, concurrent, functional, logical, object-oriented, script and visual. (Figure 1) Some of these classes are important for digitalization, while some are less important. The role of each of these classes in the digitalization process is discussed further.

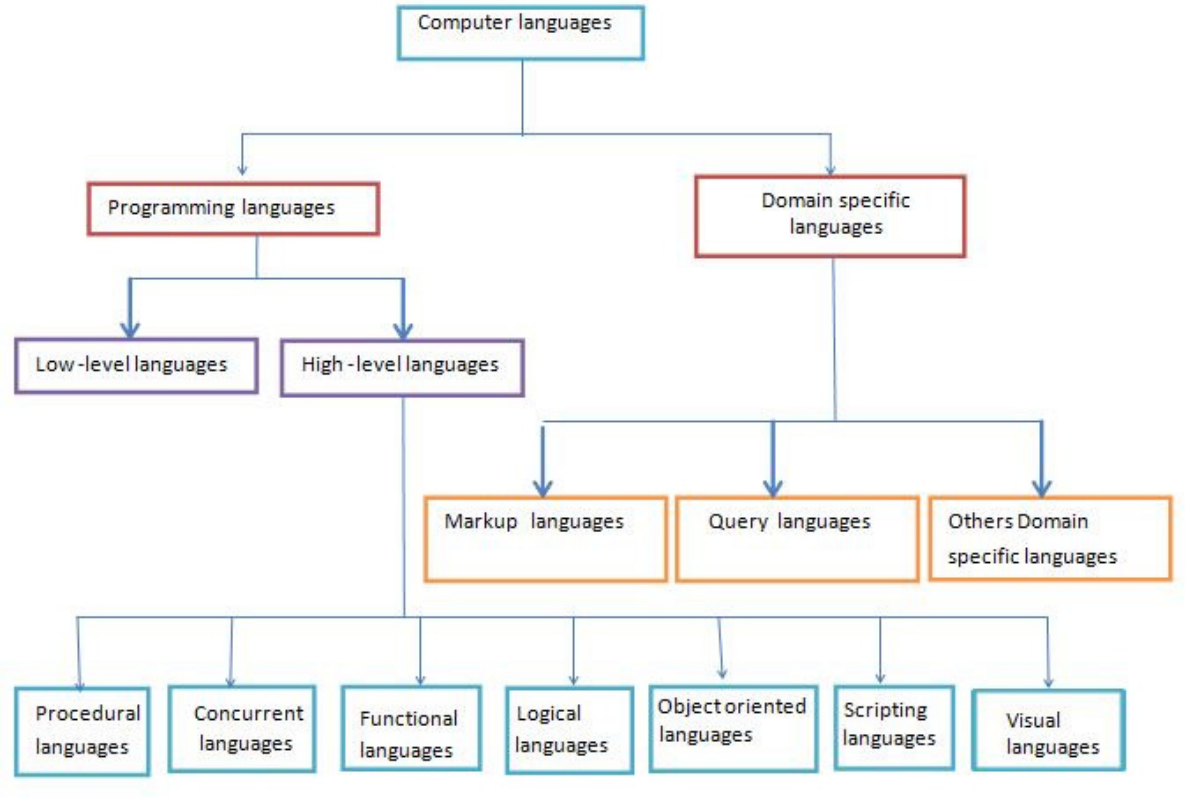

Figure 1. Classification Computer languages

2. Related work

Various classifications of computer languages can be found on the Internet and in computer books, depending on the classification criteria ([5], [15]). As high-level programming languages are the most important, their classifications are most often encountered [7]. The role of high-level programming languages in the digitalization process was discussed in the paper [9] by selecting the most popular programming languages of certain periods and analyzing their role in digitalization. In the paper [8] the role of XML technologies in the digitalization process is described. Papers [10] and [11] show in more detail the capabilities of XML technologies in digitalization.

3. Programming languages and digitalization

At the end of the Second World War, electronic digital computers supplanted analog computers and it could be said that the first steps towards digitization were made then. At the beginning of the development of digital computers, low-level programming languages were used, i.e. machine, assembly and macro languages. In this period, all processing on computers was realized with the help of these languages, and this is their contribution to digitalization. However, today they are completely supplanted by high-level programming languages. So, we will not consider their role in digitalization. Low-level languages are unsuitable for humans, so scientists tried to create a language that would be comfortable for humans, i.e. a language close to human spoken language and independent of the specific type of computer. The development tree of high programming languages is presented in the paper [9]. The characteristics of the most prominent high programming languages, created so far, are described, as well as the impact of each of them on the digitalization process. A large

number of advanced programming languages were created by forcing one or more programming paradigms [6], [7]. It is, therefore, convenient to classify these languages based on the dominant paradigm that they support. Here we will list the most famous languages of the existing programming paradigms, as well as, their impact on the digitalization process.

3.1. Procedural language. The first programming paradigm is procedural. Some authors recognize the imperative and structural sub-paradigm within the procedural paradigm. The basic concept of both sub-paradigms is procedure (hence the name procedural), and the main difference appears in the use of the GOTO statement. With the structural subparadigm, the use of the GOTO statement is limited. Prominent programming languages of the imperative sub-paradigm are: FORTRAN, COBOL, BASIC, ... The main languages of the structural sub-paradigm are: Pascal and C. The procedural paradigm is essential for digitalization. Some of actual software packages that support digitalization process are written using FORTRAN and C-language.

3.2. Concurrent languages. Concurrent paradigm (parallel) is close to procedural. Programming languages that support parallel processing are: PL/I, ALGOL 68, Ada, Concurrent Pascal, ...In these languages, the emphasis is on handling parallel processes and faster processing of large amounts of data. They are not necessary for digitalization, but they contribute to faster processing and efficient use of computer resources. It should be noted that the first languages of this paradigm (mentioned previously) are less used - they have been pushed out by modern languages such as Java and C# which also support parallel processes.

3.3. Functional languages. The functional paradigm emerged quickly after the procedural one. The first language of this paradigm was LISP, in which operations are performed by processing lists. In addition to LISP (which probably has the most descendants such as: NIL, MACLISP, Common LISP ...) the following languages are also based on this paradigm: FP, ML, Scheme, Iswim, Miranda, Haskell ... The basic concept of the functional paradigm is a function. Soon after its creation, LISP was recognized as an artificial intelligence language and many promoters of LISP assured users that the main problems of artificial intelligence would be solved by LISP. That did not happen, and interest in LISP and functional languages is rapidly declining. The functional paradigm becomes relevant again by development of purely functional languages such as: Miranda, Haskell,... LISP was used to develop various artificial intelligence applications, but also to create software packages: MATHEMATICA, AutoCad, etc. Bearing in mind these facts, we can conclude that functional languages are still successfully used and have played a significant role in the digitalization process.

3.4. Logical languages. Logical paradigm was created at the beginning of the seventies of the twentieth century. The main language of this paradigm was and remains PROLOG. Apart from PROLOG, the following languages are also based on the logical paradigm: Absys, Datalog, ASP, CLP, ILOG, Solver, ParLog, LIFE,… .PROLOG was created on the predicate calculus of the first order, and the other languages of this paradigm are also based on mathematical logic. In these languages, the main activity is the description of relations between objects. When PROLOG was created, it was bombastically announced as "the machine language of 5th generation computers". However, it did not fulfill that role. This is also the reason why interest in the logical paradigm has weakened. Nevertheless, the role of PROLOG is significant because it made possible to easily solve the problems of artificial intelligence. Also PROLOG stimulated interest in functional languages when it was realized that the solutions in PROLOG can also be implemented in functional languages. Otherwise, logical languages support digitalization, but their influence here is not of great importance.

3.5. Object-oriented languages. The object-oriented paradigm with the scripting paradigm is dominant today. This paradigm was created in the seventies of the 20th century, but unlike the logical one, it was slow to make its way. With the appearance of the language SmallTalk-80, it became known to the general public. The basic concept of this paradigm is an object. Objects are dynamic and interact with each other by exchanging messages. The programming languages that support this paradigm are: Simula 67, SmallTalk, C++, Eiffel, Java, C#,... The original language of this paradigm was Simula 67, but its main promoter was the SmallTalk language. With the advent of C++ and then Java, this paradigm became dominant. By using the language of this paradigm, digitalization is well supported and accelerated. Many scripting languages (such as Python, JavaScript, Ruby, PHP, ...) support this paradigm too.

3.6. Scripting languages. The scripting paradigm is probably the most contested. A large number of authors recognize scripting languages, but not the scripting paradigm as a separate programming paradigm. Unlike other paradigms, there is no essential concept that characterizes all its languages. Scripting languages are expanding - most of them are closely connected to the Internet and will be more and more popular in the future. A script is a list of commands that can be executed without user intervention, that is, without interaction with the user. In other words, a script is a program written in a scripting language. Script-languages often possess the features of programming languages of other paradigms, such as: object oriented, procedural, functional,...So this is also the reason why the scripting-paradigm is not recognized as a separate paradigm. Some of the script languages are: JavaScript, PHP, Perl, Python, AutoLISP, VBScript,…These languages are of great importance for digitalization and are intensively used in the digitalization process.

3.7. Visual languages. The visual paradigm is based on the use of diagrams instead of text, so it is also known as the diagrammatic, i.e. graphic programming paradigm. The basic building elements of the program are blocks and arrows. Blocks are a kind of objects, and arrows are used to connect blocks. Visual languages are used for educational purposes, for programming video games, for operating with multimedia, etc. Some of visual languages are: Alice, Flowcode, StartLogo, Dynamo, Nodal, Analytica, etc. We also include UML in Visual languages, although it is primarily a modeling language. When combined with Model Driven Architecture, UML becomes a visual language and can be said that it supports the visual paradigm. The Visual Basic, Visual C#, Visual J# etc. languages of the Microsoft Visual Studio and are not visual programming languages even though they have the attribute visual in the name. It cannot be said that visual languages have great importance for digitalization.

4. Domain-specific languages (DSLs)

While most programming languages aim to be general-purpose, DSLs are applied to solve problems within a specific domain ([12], [16]). DSLs are more often used within a program package (for example Matlab, Matcad, AutoCad, …) which also determines the application domain. The domain can be very narrow. For example, in finance, the following DSLs are used: AxLang, CPL, DAML, ... which are useful only to some people from financial circles. In terms of their characteristics, these languages are close to script languages. DSLs, in contrast to programming languages, aim to be: less comprehensive, much more expressive in their domain and convenient to eliminate redundancy. There are many areas of application of computer languages and we will not list them all (such a classification loses its meaning), but we will single out two classes of languages whose domains are very widespread. We are talking about Markup and Query languages whose domains are the Internet and databases.

Some forms of programming appear in all these languages, so some authors classify them as programming languages [15]. However, their essential characteristic is that they originated and are tied to one area. Therefore, we classified them in the DSL class. Excluding Markup and Query languages, we classified all other DSLs into one class. Some of these languages can be very specific and can only be called from certain software, such as the MASH language. A lot of them are more widely distributed such as LaTex. DSLs can be combined with each other and even with programming languages.

4.1. Markup languages. With the advent of the Internet, digitalization gains even more importance. An important role for the expansion of digitization is played by the www (World Wide Web), whose popularity is due to the markup language HTML. However, HTML did not make it possible to use all the possibilities of the Internet. Therefore, the W3C group (http:/www.w3c.org) was created, which had the task of enabling the maximum use of the Internet while forming standards for presenting data on the internet. XML (eXtensible Markup Language) is markup language, which is a meta-language for creating other languages. XML enables digitization, and the languages created from it are suitable for digitalization. Nowadays, it is difficult to imagine digitization without XML. XML has become the development of a new technology that is intensively used on the internet. More about digitization and XML technologies can be found in the paper [8], and we will not consider more this technology and XML here.

4.2. Query languages. Query languages are related to databases, so they are also known as database query language (DQL). A database is an organized collection of structured information, or data. The database is managed by a database management system (DBMS). The DBMS enables the data to be easily accessed, managed, modified, updated, controlled, and organized. Query language is used to operate data (by means of DBMS), i.e. to query, manipulate, and define data. The most famous query languages are: SQL (Structured Query Language), NoSQL (Not Only SQL), GraphQL, SPARQL, XQuery, Cypher, DMX (Data Mining Extensions), MDX (Multidimensional Expressions), etc. The digitization of databases began in the 60s of the 20th century, and intensive digitalization began with the appearance of query languages. Starting in the 80s of the last century, the most important query language was SQL, but in recent years it has been supplanted by NoSQL.

5. Conclusion

A large number of different languages is used on modern computers, so there is a need for their classification. We have classified them all into two classes: Programming and Domain Specific Languages. Programming languages can be further divided into low and high. Low programming languages are rarely used and we are not considered them. Contrary, high languages are intensively used and we proposed a classification based on the dominant programming paradigm they support. When it comes to DSLs ([13], [14]), there are a large number of domains in which they are used (word processing, finance, mathematics, creating computer games, ...) and it does not make sense to list them all. We have singled out only 2 domains: internet and databases, that related to Markup and Query languages. These languages are of great importance for both: digitization and digitalization.

References

[1] Alan Mycroft: Concepts in Programming Language, Computer Labarator, University of Cambridge, 2015 , http://www.cl.cam.ac.uk/teaching/1415/ConceptsPL/
[2] Jiyan Epözdemir: Programming Language Categories, 2024, https://medium.com/@jepozdemir/programming-language-categories-6b786d70e8f7.
[3] 10 Best Programming Languages to Learn in 2025. https://www.hostinger.com/tutorials/best-programming-languages-to-learn
[4] Introduction of Programming Paradigms, https://www.geeksforgeeks.org/introduction-of-programming-paradigms/, 2024
[5] R. Sebesta: Concept of programming languages, Addison Wesley, (7. ed.), 2006.
[6] Golding, S. A . Smolka and P. Wegner, Interactive Computation: The New Paradigm, Springer Verlag, 2006.
[7] S. Barry Cooper, Benedikt Löwe, and Andrea Sorbi: New Computational Paradigms: Changing Conceptions of What is Computable, Springer, 2007
[8] Tošić D. "XML-tehnologije i digitalizacija", Pregled nacionalnog centra za digitalizaciju, 3(2003), 1–12.
[9] Tošić D.: ROLE OF PROGRAMMING LANGUAGES IN DIGITALIZATION, Review of the NCD 44 (2024), 28–37.
[10] Tošić D., Filipović V. i Kratica J. : Using SVG – XML for representation of historical graphical data, Review of the National Center for Digitilization , 9 (2006), 39–45
[11] Tošić D., Filipović V, Tuba M. and Kratica J.: Potential Role of SMIL in Digitalization of National Heritage, Review of the National Center for Digitalization, 10 (2007), 33–39.
[12] Deursen, Arie & Klint, Paul & Visser, Joost, Domain-Specific Languages, ACM SIGPLAN Notices, 35(2000), 26–36. 10.1145/352029.352035.
[13] Igor R. Alves: Making your life easier with domain-specific languages (DSLs), 2023, https://medium.com/wearewaes/making-your-life-easier-with-domain-specific-languages-dsl-1838d351d35
[14] Federico Tomassetti: The complete guide to (external) Domain Specific Languages, 2024, https://tomassetti.me/domain-specific-languages/
[15] Peslak A.: Computer programming languages in 2020: what we use, who uses them, and how do they impact job satisfaction, Issues in Information Systems, 21:2(2020), 259–269
[16] Domain-specific language, https://en.wikipedia.org/wiki/Domain-specific_languag

dtosic@matf.bg.ac.rs