EduSol
Vol. 21. Núm. 77
ISSN: 1729-8091
Publicada en línea: octubre de 2021 (http://edusol.cug.co.cu)

El pensamiento analítico - asociativo en la enseñanza y aprendizaje de la Programación. Analytical - associative thinking in teaching and learning programming

Juan Carlos Fonden Calzadilla
Facultad de Ciencias Médicas "Dr. Miguel Enríquez". La Habana. Cuba
ORCID: https://orcid.org/0000-0001-7478-8628
Correo(s) electrónico(s):
fonden1980@gmail.com

Recibido: 15/08/2020
Aceptado: 12/02/2021

Resumen: Recurriendo a los métodos de investigación análisis documental, modelación, enfoque de sistema y la observación, se realizó un estudio del proceso de enseñanza y aprendizaje de la Programación en las universidades José Eduardo Dos Santos" de Angola y la Universidad Tecnológica de la Habana "José Antonio Echeverría"; constatándose insuficiencias al relacionar clases, objetos y sus códigos asociados y como consecuencia bajos rendimientos académicos. Como resultado se avala la necesidad de elaborar recomendaciones metodológicas aplicables a ambos contextos educativos que contribuyan a la formación de un pensamiento analítico y asociativo que posibilite la adquisición de resultados docentes superiores en esta disciplina.
Palabras clave: Pensamiento analítico; Pensamiento asociativo; Programación; Recomendaciones metodológicas**.**

Abstract: Using research methods, documentary analysis, modeling, system approach and observation, a study of the teaching and learning process of programming was carried out at the José Eduardo Dos Santos "universities in Angola and the" José Antonio Echeverría "; finding insufficiencies when relating classes, objects and their associated codes and as a consequence low academic performance. As a result the need to develop methodological recommendations guarantees applicable to both educational contexts that contribute to the formation of an analytical and associative thinking that enables the acquisition of higher learning outcomes in this discipline.
Keywords: Analytical thought; Associative thought; Programming; Recommendations metodológicas.

Introduction

Learning programming, a fundamental element in the professional's computer training, is a continuous, complex and planned process. Its success is due to the planning, organization, execution and control of teaching task systems with an adequate theoretical-practical relationship, in which motivational, affective, cognitive and evaluative elements are integrated, in accordance with the demands and needs of the study plan and the objectives of the subject, and its study is considered a difficult activity due to the complexity involved in its development.

There are learners who fail to acquire the necessary skills for programming, even after completing an introductory course of algorithmization, "a study estimates that between

25 to 80 percent of students in the United States drop out of their first programming classes due to the difficulty they face in learning to program" (Insuasti, 2016, p.237) and teachers of this computer science discipline, perennially express the need to do something to counteract these low academic results, a matter in permanent debate.

Insuasti (2016) and other authors consider that the preliminary development of some thinking skills such as analysis, synthesis and abstraction would facilitate the learning of Programming.

On the other hand, the author has found, since the course 2009 - 2010, until today, that in Technological Universities and other educational institutes, in the Teaching and Learning Process (P.E.A) of Programming, low academic results persist and among its causes are observed the difficulty to analyze and identify accurately the links and associations between classes or entities of the real world and their associated codes; writing instructions directly into the computer without prior analysis and failures to establish relationships between components and subcomponents in the realization of a software.

The aforementioned problematic situations suggest the need for further research to obtain superior academic results in the PEA of Programming.

For the development of this article, the following tasks were planned and accomplished:

Systematization of the studies carried out on the P.O.O. PEA in the network of networks and other bibliographic sources; Diagnosis of the Teaching - Learning Process in the subjects Structured Programming, Object Oriented Programming (O.O.) and Data Structures in the subjects Data Structures in the subjects Structured Programming, Object Oriented Programming (O.O.) and Data Structures in the subjects Structured Programming, Object Oriented Programming (O.O.). O) and Data Structures in the Faculties of Informatics and Industrial Engineering at the University "José Eduardo Dos Santos" of the Republic of Angola and the Technological University of Havana "José Antonio Echeverría"; Identification of the particularities of the teaching of O.O.P. and interviews to programmers, professors and students; Analysis of the teaching results in the O.O.P. exams, methodological indications, books, booklets with solved exercises and guides for the students.

In view of the above, we will continue to explore whether the development of analytical and associative thinking contributes to solve the shortcomings detected, whether it enhances the solution of complex problems and the achievement of superior academic results. Next, different links among entities will be analyzed and methodological guidelines will be elaborated in order to apply them to the ADP of Programming.

Development

Teaching programming in schools is not a new activity. Programming languages such as Fortran, Lisp, Cobol and Logo emerged at the end of 1955 and became powerful tools for young programmers who started this task. The PEA of Programming has, among its essential objectives, the training and development of skills, which makes it possible to solve problems in the school, professional or practical life (Cortés, 2017).

Likewise, the contribution provided by learning programming to the formation of logical and abstract thinking in students, who are trained in these tasks from an early age, is widely recognized (Cortés, 2017).

The (P.O.O), programming paradigm based on objects and their interactions, which simulates the way in which human beings live and interact with things and events in the real world, where actions are identified, modeled and programmed on classes related and/or associated with each other. It is a paradigm popularized since 1990 and introduced a new way of constituting the code of a computer program. Currently, there is a wide variety of programming languages that use the object-oriented paradigm.

Other reasons to encourage the study of programming at an early age are the development of systemic thinking, the identification of classes, attributes and variables in a problem, collaborative work, self-management of knowledge, learning from error, modeling of different strategies for problem solving, decision making based on logical reasoning and the development of computer projects.

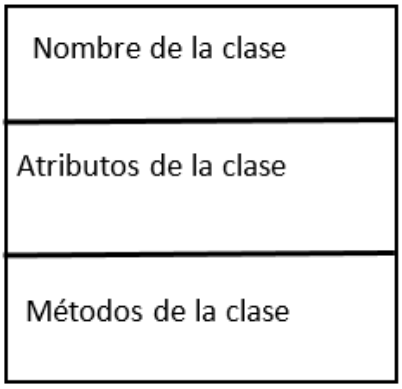
Due to the ideas expressed, nowadays, in countries such as France, Estonia, United Kingdom, Finland, Australia and Germany, the learning of programming is encouraged. For several years, several nations have begun to adopt programming as part of the common curriculum of the study plans, understanding that this discipline is necessary for the intellectual development of students.

In the present work, a programming paradigm constitutes a specific approach or ontology for designing solutions (Insuasti, 2016). Paradigms differ from each other in the concepts and the way of abstracting the elements involved in a problem, as well as in the steps that compose its solution. It is also established that the term relate is synonymous with associate, link, combine, connect, link and couple.

In addition, an entity is a recognized element, process or event, with identifiable information, examples: a person, company, chair, bank account, a planet, etc., of the real world, and as such possesses certain attributes that identify it and differentiate it from other entities (Fonden, 2019). Entities are the object of study in the Object Oriented Programming discipline.

A class is something similar to entities. A class is a model used to create new objects and does not exist in isolation. Every object or entity is an instance of a class. The possibility of creating hierarchies between classes is a particularity that differentiates P.O.O. from other programming paradigms, because it makes it possible to extend and reuse existing code (fig. 3).

A class is represented in the UML class diagram as a rectangle divided into three parts, (fig. 1):



Graph No. 1. Graphical representation of a class or entity. Own elaboration.

Analytical and associative thinking. Examples.

Analyze is a term constantly used in different contexts of life. Analysis is an intellectual activity that requires the decomposition of an object or process in all its constituent parts, thus, the analysis of the Teaching and Learning Process of a subject implies the

observation of the content, methods, forms of evaluation, organizational forms, teacher and students, among other elements.

Analysis is logically opposed to synthesis because it is an operation of thought that consists of the ordered composition of the different elements of a whole, an operation contrary to the analytical one, however, both complements each other.

Sometimes, in the processes of analysis, there are obstacles derived from mental models or predefined stereotypes that prevent reaching solutions, these models are far from the true analysis, for this reason, "the source of the problems in thinking is often in false assumptions. Since assumptions are usually unconscious, they often incorporate biases, stereotypes and arbitrary false beliefs" (Elder and Paul 2003, p.14).

At the same time, in teaching, as in nature, society and thought, knowledge, subjects, subjects, disciplines, sciences are linked and these links originate new subjects and sciences (Lazo, Valcárcel and González 2015; Pedroza, 2006). Thus, computer science is the automatic processing of information with the use of computers; biochemistry is an interdisciplinary science, which draws its subjects of interest from many disciplines and thus happens with other sciences such as political economy and telecommunications (Rosental and Ludin, 1984).

These links are founded, philosophically, on a general law known as the Law of Universal Concatenation of all objects and phenomena and other laws of dialectics. "Connections among objects and phenomena can be direct and indirect, constant and temporary, essential and non-essential, casual and necessary, functional" (Rosental and Ludin 1984, p.76).

As follows, the author analyzes a set of possible links of the object or entity Student, in interaction with other entities:

1. A student is related to himself, he can be a group leader or subordinate. Both, the leader and the subordinate is a student. This type of relationship is called involutive classes in P.O.O. It is also known as a reflexive relationship. In mathematics, a reflexive relationship is one where each element relates to itself. The student relates to himself/herself at all times in life;

2. The student relates temporarily (transitorily) to a Teacher. 1:1 type of relationship. One-to-one, determined by a time. This may occur during a class, a consultation or an exam. It can also be a stable relationship (during a semester, a school year or more) with one or more teachers. Type 1 relationship: M. One-to-many. This relationship is well known and frequent, since the student also relates to many students and many other objects in life;

3. The student is a scholar (internal) or external. The relationship is inheritance relationship in Programming and constitutes a specialization - generalization. It is the mechanism by which a class inherits the characteristics and methods of another class;

4. A student, in the laboratory, at a given time is taking an exam. Here the entities Student, Laboratory and Hour intervene, which when related give rise to a new entity called Examination and is known as aggregation;

5. Many students are related to many teachers. Relationship of type M: M. From many to many, very used in relational databases;

6. Student relates to things or entities in the real world, among them, his computer, money, food, car and weather, relates to his thoughts and emotions. Infinite relationships and associations;

What universe of associations will result when simultaneously interacting the entities Teachers; Disciplines or subjects; Departments; Faculties; Classrooms; Laboratories; Exams; Parents and others?(fig. 2 and 3). Every time two classes unite to work together to achieve an end, an association is produced, which will give rise to others.

This article assumes associative thinking as the intellectual capacity of the human being to constitute and relate ideas, opinions, concepts, reasoning and representations in his mind, (Fernandez, 2016) characterized, in the teaching and learning process, by presenting the content intertwined (combined, associated, linked, connected), as a whole, in order to achieve the system of knowledge, skills and values to be fulfilled in the curriculum of the career.

Associative thinking allows the solution of complex problems and the design of new products and valued knowledge, because it conceives everything connected, which at the same time is the most difficult to accept, because superficially the world may seem disconnected, chaotic and unconnected.

Associations can be temporary or permanent, depending on the context, as happens in life. A person temporarily relates temporarily to the printer, the doctor or the lawyer, and permanently to himself, to the person who shares his room and to his pet.

Mental associations are more stable the more meaning they have for the subject. Thus, remembering something depends, to a great extent, on the interest one has in doing it and on the experiences acquired from other similar events.

Associative thinking analyzes events from the point of view of multiple interactions, internal or external, permanent or transitory.

Contrary to associative thinking is sequential thinking, in which a linear chaining of events and factors is perceived. Although a wide range of problems can be solved from the perspective of sequential thinking, others will require the establishment of multiple associations between objects, processes and phenomena for their understanding, both in everyday life and in learning programming.

In addition to everything expressed so far, an element to take into account when facing a problem is the significant knowledge of the identified entity, i.e., the previous experience acquired by the student about the process in which the entity is involved. If the entity is the Bank Account of a Client in a Bank, a previous knowledge or information search of the functioning of a Bank and its interaction with the entities, Bank Account and Client is needed.

Absolute ignorance of the process where the identified entities are located leads to errors when relating them. The subject under study must be meaningful to the learner, that is, potentially relatable to the cognitive structure of the learner, and thus can create mental associations.

Association between emotions and learning outcomes. Opinions.

Emotions influence learning, so it is advisable to promote enjoyment, self-esteem and avoid negative emotions such as fear. The lasting association detected between emotions and learning outcomes should be further studied and taken into account.

Inquiry instruments were applied to 236 students distributed in 8 groups at the University "José Eduardo dos Santos" of the province of Bie in Angola during the 2016, 2017 and 2018 school years and to 124 students distributed in 4 groups of second year of the

Industrial Engineering career at the Technological University of Havana, at different times.

The 124 Industrial Engineering students were asked the following questions at the conclusion of a mid-term exam:

1. Express your opinion about the P.O.O. subject.
2. Say whether you are pleased with your lecture and practical classes.
3. Do you consider that this course contributes to your professional training?

More than 60% of the students answered positively to the three questions and here is one of the opinions expressed by them:

I like the subject, I understand the lectures well, although I enjoy the practical classes more because the professor is very patient, well treated and tells us how to apply the knowledge.

On the other hand, an Angolan student referring to his computer science professor wrote in a social network again I never tire of recognizing and thanking him for the great person he is, always ready to teach and learn detailed, prudent, great companion and friend. I will always be grateful.

Another teacher who worked with the aforementioned students in Angola was congratulated by his students with phrases like "the teacher helped us to think and we will always remember him because we have learned more than we thought" and others said "we can now create our software company".

In the teaching of science, information on the role of emotions in learning is limited. In this regard, the author recommends reflecting on what Einstein said in The New York Time newspaper on October 5, 1952:

> It is not enough to teach a man a specialty. Although this may turn him into a kind of useful machine, he will not have a harmoniously developed personality. It is essential that the student acquire an understanding of the results and a deep affinity for them. He must acquire a vigorous feeling for the beautiful and the morally good. Otherwise, with the specialization of his knowledge he will look more like a well-

> trained dog than a harmoniously developed person. (Proceedings of the International Convention on Public Health, 2012, p.1)

Other research carried out

The author, as a participant in and observer of the Program's PEA, elaborated an observation guide to verify the implementation of actions aimed at the formation of analytical and associative thinking in teaching. Information was collected on the following questions:

1. Is it a work style to exemplify the content of the P.O.O. from phenomena addressed in other subjects or disciplines?

2. Is the content of the programming linked to the students' daily life experiences?

3. Does it provide a type of learning that leads the student to make conjectures, assumptions, questions or hypotheses about the subject matter received in Programming, linking it with knowledge from other disciplines and from life?

4. Does it establish relationships between the aspects studied in a class with others known to them in previous grades?

5. Do you take into account the relationship that the facts and phenomena you teach have with laws or theories that, due to their level of generalization, require associations between subjects or disciplines?

In the 30 visits to classes made to teachers in the subject P.O.O. and other subjects during the school year 2018 - 2019, in the faculty of Industrial Engineering, 90% partially or totally failed to comply with the controlled questions. Coincidentally, the evaluation results of the students of a partial examination, who were being taught by the controlled teachers in the school year 2018 - 2019, yielded the following results:

(a) More than 90% of the 130 evaluated identify all the classes involved in each subject;

b) Between 40% and 50% of those evaluated failed to accurately identify the associations between the classes and the programming codes that emanate from them (Fig. 3).

Some research results suggest that the association of ideas and/or events is acquired at early ages and is exercised throughout life, in this sense, the analysis of the data of this research shows that students over 40 years of age, from the courses for workers, were

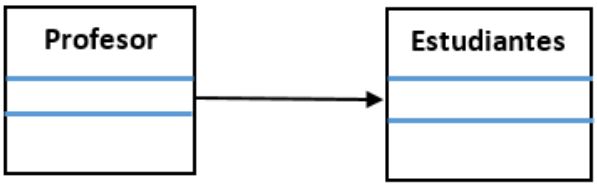
more likely to face problems in the P.O.O., especially when relating classes, objects and associated programming codes.

At the "José Eduardo Dos Santos" University, successive comparisons were made between students of the regular day course and the course for workers, in the Computer Science specialty, when identifying and associating classes to build the programming code that solves the problem. In both courses, low academic results were obtained, although very low in the students of the course for workers, less than 30%. In the case of adult students and workers over 45 years of age, their academic performance was less than 20%.

Relationships between classes. Graphical modeling of association and inheritance.

In P.O.O., two objects or entities can interact with each other, relate to each other in different ways, among them: The association relationship, a basic type of relationship between two classes, when one object accesses the attributes and methods of another; inheritance or generalization relationships and dependency relationships that indicate the relationship existing between two classes, where one depends on another to perform its work, in a possible temporal relationship. There is also a type of strong relationship in which one class contains another and the contained one is meaningless without the containing class, this type of relationship is called nested classes. The following are the graphical representations of the association and inheritance relationships (fig. 2, 3).

There is a strong type of relationship called composition, where the lifetime of an object determines the lifetime of other content. Example: A book and its pages. If the book is destroyed, its leaves cease to exist.



Graph No. 2. Relationship of association between the classes Teacher and Students. Authors' own elaboration.

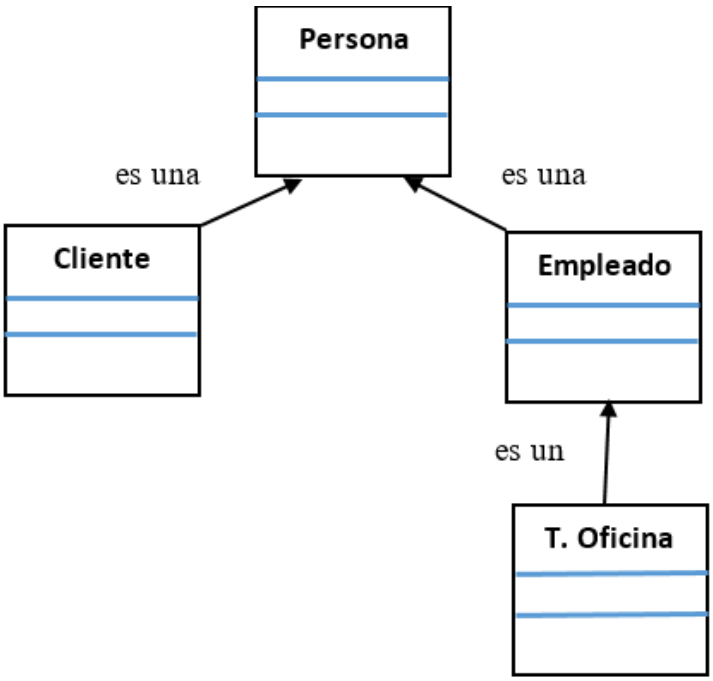


Figure 3. Inheritance relationship between the classes Person, Client and Employee.

Own elaboration.

Methodological recommendations.

The systematization made by the author to the studies of (Rosental and Ludin, 1984; Elder and Paul, 2003; Fernández, (2016), Izaguirre, (2014), Lazo, Valcárcel and González, (2015) and Acevedo, (2004)), the results obtained from the classroom observation guide, the surveys conducted and the midterm exams, together with his professional experience as a researcher and teacher of Programming allowed him to conclude that analytical and associative thinking is necessary to obtain superior academic results in the PEA of Programming, although it does not arise from spontaneity.

Therefore, whether you have a textual problem or a problem expressed in a diagram and you have to identify parts of the whole, relationships between parts, internal or external. Thus, if you have to relate entities to programming code fragments or to the parts of a software. The following methodological recommendations, aimed at teachers and students, will help you to achieve this:

- Propitiate, above all, an atmosphere of collaboration, respect, joy and satisfaction when approaching each Programming content in the classroom or study room. Avoid unnecessary criticism and encourage self-confidence;

- Read, examine and interpret the teaching problem or process to be coded, identify its main and secondary parts, and their relationships. Collect relevant information and discover its properties. Analyze the problem from different points of view and link seemingly unrelated elements. Think without restrictions;

- Analyze and reflect comprehensively to harmoniously interpret the teaching problem. Identify the elements composed by the whole; examine their properties and interrelationships;

- Synthesize, after joining the integral parts of the whole, expose the essential with clarity and precision. The synthesis is part of the analysis.

- Verify the existence of errors or inconsistencies in each part identified in the problem or process. Make logical deductions from the data;

- Identify permanent and temporary relationships between parts, i.e., between classes, attributes, operations or parts of a program and the common and different between components of one or more software;

- Develop concept maps to relate concepts and ideas;

- Model theoretically or graphically the identified elements and the conformation of the whole, to structure and promote the student's learning process. This idea is supported by Arias (2016), among many authors. Modeling is key to introduce new concepts and develop them (fig 3);

- Model theoretically or graphically each solution to be programmed, through algorithms and/or diagrams. A good algorithm is as necessary and essential as a good programming code (Manovich, 2017). Establish the progressive transition from algorithm to programming code;

- Relate with professors and students from other universities, with different points of view than the one received in classes and dialogue with them about your project;

- Enjoy working with others, in teams, in which there is diversity of criteria and consciously organized collaboration between people, technologies and disciplines that cooperate to achieve a common goal. Apply teaching and learning methods used by teachers of other subjects or disciplines, which is called cooperation from the method;

- Avoid, in working with others, erroneous inferences "whereby one concludes that something is true on the basis of what something else is true, or appears to be true" (Elder and Paul 2003, p. 47). Inferences emerge from assumptions, and if the assumptions are incorrect, so will be the inferences. Verify and question assumptions. The fact is the reality and the assumption is what is believed to be the fact, without gathering all the necessary evidence. Analyze and verify beforehand;

- Frequently examines the sequence and coherence between contents, i.e., inter-subject and inter-subject relationships;

- The process of analysis and synthesis, together with the identification of inter-subject, inter-subject and interdisciplinary relationships, promotes analytical-associative thinking.

Application of the methodological recommendations

A partial introduction of the methodological recommendations identified above in the educational practice has been carried out in the course of Industrial Engineering, during the course 2018 - 2019, in the subject Design of Computer Solutions of the new E plan. In this sense, the teaching analysis of the results showed a tendency to recovery, both quantitatively and qualitatively, due to the superior quality of the computer projects presented and their defense in court. It was possible to avoid, to a great extent, those students from other specialties or careers would do the programming codes and the design of the database in the projects.

The quality of the digital reports was also improved, as well as the quantitative results of the discipline, in terms of final grades and reduction of the number of failures.

In methodological meetings held in the course 2018 - 2019 with teachers of the subject, the essential elements expressed in the methodological recommendations were discussed, showing a high level of approval by teachers, which is reflected in the preparation of classes and the analysis and design of computer projects.

Conclusions

The partial introduction of the methodological recommendations in the educational practice in the Industrial Engineering course, during the 2018 - 2019 academic year,

showed a tendency to recovery, in terms of the acquisition of analytical and associative thinking in the PEA of Programming, both quantitatively and qualitatively.

The Object Oriented Programming (O.O.P.) highlights an accumulation of links and associations between entities, which requires the formation of an analytical-associative thinking, and the organized collaboration between people, technologies and disciplines that cooperate to obtain a common goal.

For a good development of the PEA of Programming, it is necessary to have acquired a certain level of analysis and reciprocally this is reached and developed by going through this process.

The analysis of the possible links between one, two or more entities and the methodological recommendations expressed, enhance the development of an analytical, associative and analytical-associative thinking and as a final result, superior academic results in the PEA of Programming.

P.O.O. is a discipline that prepares students to work in teams, face challenges, solve complex problems, manage conflicts, elaborate projects, develop perseverance and associate ideas, processes and life phenomena.

Bibliographic References

Acevedo, J. A. (2004). El papel de las analogías en la creatividad de los científicos: la teoría del campo electromagnético de maxwell como caso paradigmático de la historia de las ciencias. *Eureka sobre Enseñanza y Divulgación de las Ciencias*, *1*(3). https://revistas.uca.es/index.php/eureka/article/view/3947.

Arias, L. A. (2016). Lenguaje de modelamiento unificado (UML) para modelamiento de embotelladora. *Scientia et Technica*, *21*(1). https://www.redalyc.org/pdf/849/84950584006.pdf.

Cortés, J. (2017). Educación. La importancia de que los niños aprendan a programar. *El país*. 6 de Abril de 2017. https://retina.elpais.com.

Elder, L. y Paul, R. (2003). Pensamiento analítico *The Foundation for Critical Thinking*. https://www.criticalthinking.org.

Fernández, C. (2016). Pensamiento relacional en primaria: el papel del maestro. *Uno. Revista didáctica de las matemáticas*, (73).

https://rua.ua.es/dspace/bitstream/10045/65269/1/2016_Fernandez_Ivars_UNO.pdf.

Fonden, J.C. (2019). Importancia del pensamiento abstracto. Su formación en el aprendizaje de la Programación. *EduSo*l, 20(72). https://edusol.cug.co.cu/index.php/EduSol/article/view/1195/2499.

Insuasti, J. (2016). Problemas de enseñanza y aprendizaje de los fundamentos de programación. *Educación y desarrollo social*, 10(2). https://revistas.unimilitar.edu.co/index.php/reds/article/view/1966. DOI: org/10/18359/reds.1701.

Lazo, M.A., Valcárcel, N. y González, T.R. (2015). Modelo de Superación con enfoque interdisciplinario en tecnologías de la Salud. *Revista cubana de tecnología y salud, 6*(4). https://www.medigraphic.com

Manovich, L. (2017). Los algoritmos de nuestras vidas. CIC. *Cuadernos de Información y comunicación*, 22. https://www.redalyc.org/.

Rosental, M. & Ludin, P. (1973). *Diccionario filosófico. Representación.* Editorial Política.