Didactic Transposition in Extensa Sensu for Teaching Computer Programming Fundamentals

Jesus Insuasti *, Carlos Mario Zapata-Jaramillo , Alexander Barón

Posted Date: 21 June 2023

doi: 10.20944/preprints202306.1483.v1

Keywords: Didactic; Transposition; Programming; Fundamentals; Computational; Linguistics

Article

Didactic Transposition in Extensa Sensu for Teaching Computer Programming Fundamentals

Jesús Insuasti [1,*], Carlos Mario Zapata-Jaramillo [2] and Alexander Barón [3]

[1] University of Nariño; insuasti@udenar.edu.co
[2] Universidad Nacional de Colombia, sede Medellín; cmzapata@unal.edu.co
[3] University of Nariño; abaron_98@udenar.edu.co
* Correspondence: insuasti@udenar.edu.co;

Abstract: Today, computer programming is an essential pillar in the education of individuals. Even though technological evolution has eased the massive need for computer programming, a systematic literature review shows some problems remain with teaching computer programing. To address such issues, in this paper we propose an adapted theory, a method, and a model for teaching some knowledge objects associated with the concepts of variables, expressions, comparisons, loops, and functions in programming fundamentals in computer sciences. The main contribution of this study comprises the extension of the concept of didactic transposition to improve teaching in context; in this paper, the idea is called didactic transposition in extensa sensu. The proposal involves the development of a linguistic corpus based on *syllabi*, textbooks, reference manuals, and a survey applied to experts in teaching programming fundamentals based on seventy-eight universities worldwide. An adapted theory, a method, and a model for teaching programming fundamentals using computational linguistics are produced. In addition, a template is created for designing teaching practices in this regard. The proposal generates an important alternative for the development of teaching strategies where the conditions of the context are addressed as a priority; additionally, several aspects are considered for elaborating teaching strategies using computational linguistic techniques from written documents.

Keywords: didactic; transposition; programming; fundamentals; computational; linguistics

1. Introduction

Computer programming is the educational area where the capacity for abstraction is a critical success factor in computer science. This area includes a theoretical foundation, a series of programming languages and tools, and cognitive skills for solving problems from a systemic approach—computational thinking [3]. From this viewpoint, the preliminary step to mastering such an educational area in computer science is the course on programming fundamentals, commonly called Programming 101.

The traditional teaching of programming fundamentals includes the essential concepts of programming languages after teaching the theoretical foundation, and then strategies involving the entire programming process. Teaching the crucial concepts of programming fundamentals, such as variables, arithmetic-logical expressions, instructions, comparisons, loops, and functions, is a critical point in the education of students within the area of computer programming; these essential concepts are entirely abstract, and they are massively used worldwide regardless of the adopted programming language [20].

Success in computer programming comprises the solution of actual problems by using programming languages (code implementation) including these essential concepts [23]. For this reason, teaching programming fundamentals is a complex task, because students must develop computational thinking skills by using crucial concepts with a substantial level of abstraction [35]. This is a challenge for teaching such essential concepts. According to a systematic literature review, teaching programming fundamentals faces difficulties due to previous education, abstraction, complexity, and weaknesses [9].

According to some worldwide experiences of experts in teaching programming fundamentals, we established a linguistic corpus enriched with documents such as *syllabi*, textbooks, and some reference manuals of programming languages. We could conceptually identify some essential knowledge objects for teaching programming fundamentals using computational linguistics techniques; thus, some word clouds and ontologies could be represented in this regard. From this point, the transposition levels of the essential objects were defined, generating an adapted proposal based on the didactic transposition theory [7]. Considering this extended proposal, a method and a model were developed. The construction is called didactic transposition in extensa sensu (from Latin: "in an extended sense"), which involves an adapted theory, a method, and a model. The construction is based on the findings from a literature survey; the method and model produced in this study are relevant in the sense of providing an alternative method of teaching computer programming fundamentals by considering the needs of the context.

The remainder of this paper is organized as follows: In Section 2, we describe the methodological aspects and findings used to propose a solution. In Section 3, we state the didactic transposition in extensa sensu based on [21], which includes an adapted theory, a method, and a model for teaching programming fundamentals. Our conclusions and the relevance of the proposal are presented in Section 4. Finally, areas of future work are described in Section 5.

2. Context, Problem, and Theoretical Background

Programing skills are difficult to learn; therefore, teaching the programming fundamentals entails didactic challenges for developing abstraction and algorithmic thinking [22]. Here, creating and controlling environments and computational solutions using programming skills are difficult for an individual. For this reason, teaching programming fundamentals is considered a complex and challenging activity. Learning is also affected by the abstraction level of the knowledge objects. The high dropout rates in computer programming fundamentals reveals a serious problem in the teaching of such courses [27].

The teaching of initial programming courses has been the object of study, with a similar trend found in terms of the difficulty of teaching computer programming given the method of approaching the knowledge objects of abstract nature. So, teaching programming fundamentals is considered a challenging task because, for most students, this is associated with the interpretation of overly complex objects [11]. From the student perspective, one of the main reasons for dropping out from their educational process is the courses in question [21][34]. Some aspects have been considered in the nonprogression of students in computer science. According to the results of a specific survey:

"By far the most highly ranked reason for considering leaving was 'subject difficulty'. 74% of respondents indicated that they had encountered difficulties with particular modules in the first semester: Of these, 44% reported difficulties with an ICT module (e.g., computer organization, computer software, electrical engineering); 33% had difficulty with a maths module (e.g., computer maths, engineering maths)" [28].

The difficulty of teaching such courses has been the subject of debate for some professors of disciplines associated with computing. Additionally, some computer science academics have discussed the need to intervene in this problem; such is the case of the ACM and IEEE Computer Society manifestos regarding curriculum recommendations and didactic approaches to address this problem [1].

Regarding teaching, the didactic transposition theory becomes more relevant when dealing with abstraction issues. One of the best scenarios is teaching mathematics when interacting with abstract entities. Here, in mathematics teaching, the didactic transposition theory has been used in different contexts. Didactic transposition is how professional knowledge—scientific, academic, expert, or scholarly knowledge—is transformed into content to be taught [7]. The didactic transposition comprises the migration of knowledge in the reference community, called expert knowledge, toward the understanding alive in the classroom, called taught knowledge [35].

 doi:10.20944/preprints202306.1483.v1

3. Methodological Aspects and Findings

The paradigm of this study is qualitative with some quantitative complements. As such, we identified the nature of realities, their dynamic structure, and their behavior and manifestations. Qualitative research—the integrated whole based on qualities—is aligned with the quantitative one, which is only an aspect of measurement, because they complement each other, especially when we work with intersubjectivities [6].

In addition, this study involved a hermeneutical approach. According to [13], the hermeneutical exercise involves the integration of scientific progress and thought from a philosophical viewpoint, so the art of interpretation points toward an understanding of the characteristics of existence; traditionally, and given the specific and finite nature of the human being, experience is the claim of objective knowledge. The hermeneutical exercise has, as a principle, to leave open the dialogue oriented for holistically understanding. As such, this study focused on understanding the typical situations of teaching programming fundamentals within the classroom. Thus, [32] suggested, to understand a phenomenon in depth, "Understanding is hermeneutic: from now on, seeking meaning is no longer about spelling out awareness of meaning, but about deciphering their expressions."

Considering the nature of this study, we searched for the aspects related to experience in teaching programming fundamentals for creating a linguistic corpus starting from various sources, as depicted in Figure 1.

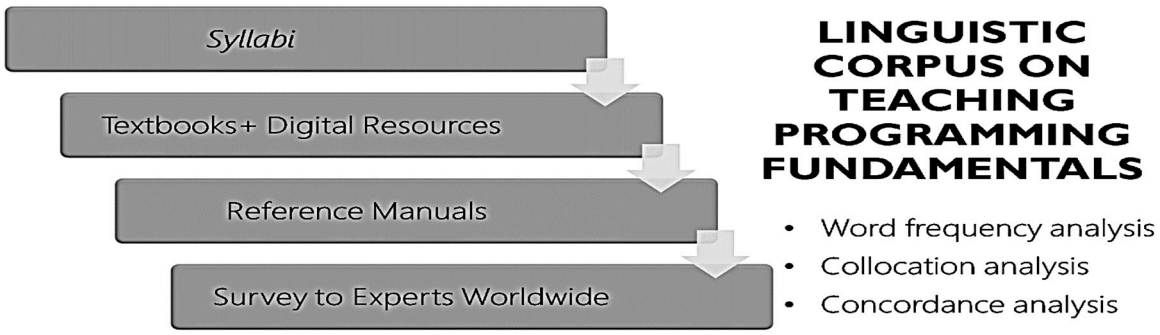


Figure 1. Sources of the linguistic corpus on teaching programming fundamentals. Source: The authors.

The world ranking of universities in computer science and engineering given by Shanghai Jiao Tong University—Academic Ranking of World Universities (ARWU)—was considered for establishing a baseline scenario [33]. We sent invitations to the experts (The inclusion criteria to select the experts were: (i) holding a position of full or emeritus professor; (ii) providing some evidence about currently teaching, or having taught, some courses related to programming fundamentals in the last five years.) in the top five hundred universities in the world to participate in this study; from such invitations, only seventy-eight professors agreed to be surveyed. In Figure 2, we depict the global distribution of the experts who participated in the baseline scenario for this study.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

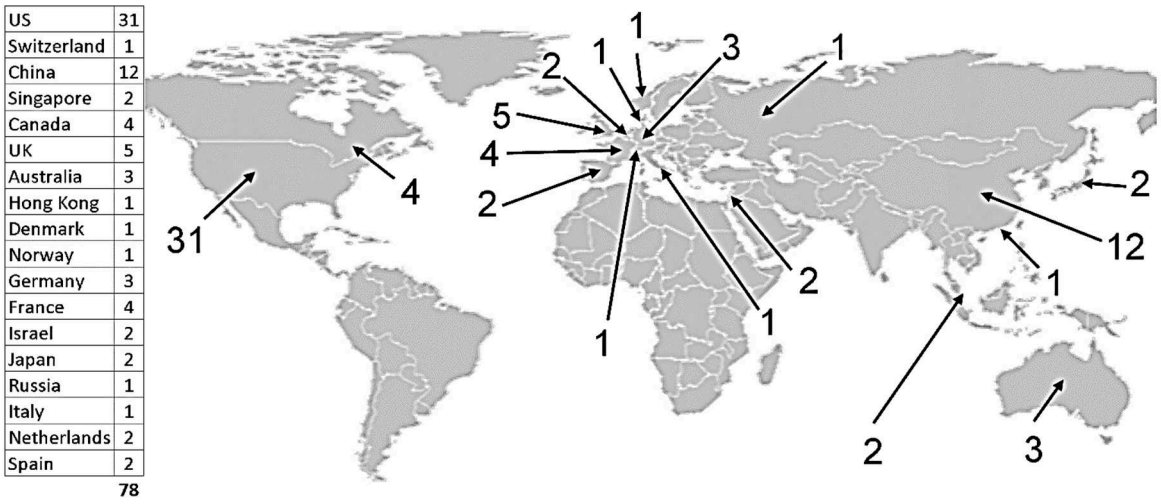
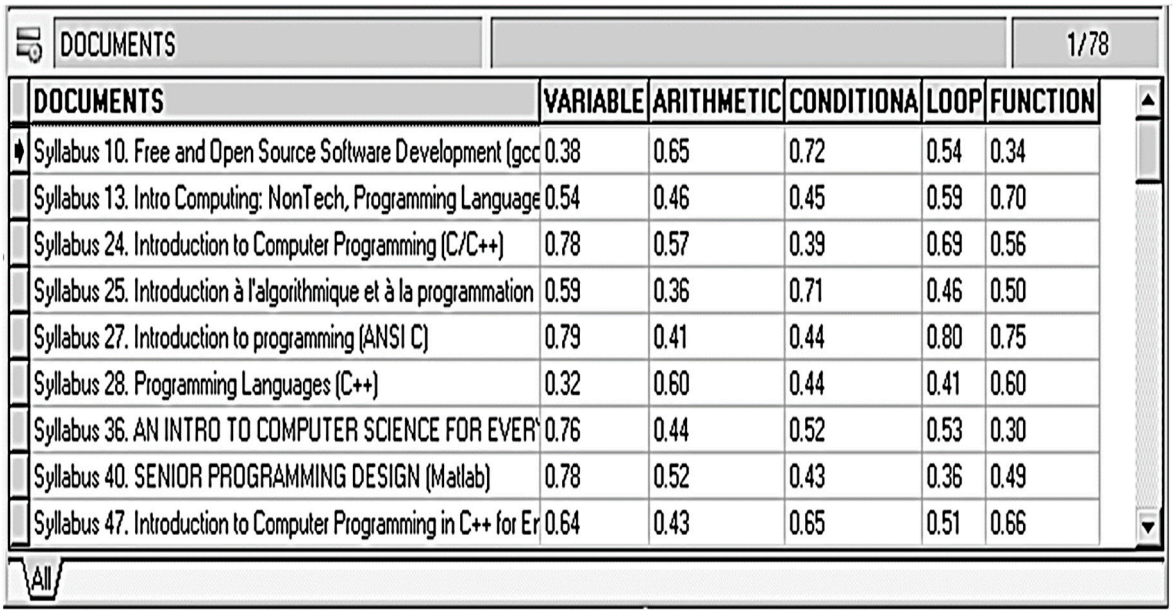


Figure 2. Global distribution of the experts involved in the baseline scenario in this study. Source: The authors.

The selected experts granted access to their syllabus, textbooks, digital resources, and reference manuals. In addition, the experts filled out a survey form; as such, we could address all the information using computational linguistics techniques. According to [18], "Corpus linguistics can be considered a sophisticated method of finding answers to the types of questions linguists have always asked". A large corpus can be a test bed for hypotheses and be used to add a quantitative dimension to many linguistic studies. Corpus linguistics has also led to a re-evaluation of language.

Some essential knowledge objects on computer programming fundamentals are considered common ground. For instance, "to ask students a question requiring a loop also necessarily involves asking about variables, expressions, and assignment, in addition to more complex structures such as selection" [40]. The characterization of five essential knowledge objects in this study is provided with this corpus. Such essential knowledge objects include variables, arithmetic-logical expressions, conditionals, loops, and functions. The QDA Miner® tool version 6 was used to analyze categories of each essential knowledge object. In Figure 3, we depict an excerpt of the frequency of use of each essential knowledge object by using the *syllabi*, textbooks, reference manuals, and responses from a survey of experts worldwide.



DOCUMENTS	VARIABLE	ARITHMETIC	CONDITIONA	LOOP	FUNCTION
Syllabus 10. Free and Open Source Software Development (gcc	0.38	0.65	0.72	0.54	0.34
Syllabus 13. Intro Computing: NonTech, Programming Language	0.54	0.46	0.45	0.59	0.70
Syllabus 24. Introduction to Computer Programming (C/C++)	0.78	0.57	0.39	0.69	0.56
Syllabus 25. Introduction à l'algorithmique et à la programmation	0.59	0.36	0.71	0.46	0.50
Syllabus 27. Introduction to programming (ANSI C)	0.79	0.41	0.44	0.80	0.75
Syllabus 28. Programming Languages (C++)	0.32	0.60	0.44	0.41	0.60
Syllabus 36. AN INTRO TO COMPUTER SCIENCE FOR EVER	0.76	0.44	0.52	0.53	0.30
Syllabus 40. SENIOR PROGRAMMING DESIGN (Matlab)	0.78	0.52	0.43	0.36	0.49
Syllabus 47. Introduction to Computer Programming in C++ for Er	0.64	0.43	0.65	0.51	0.66

Figure 3. Frequency of use of five essential knowledge objects. Source: The authors.

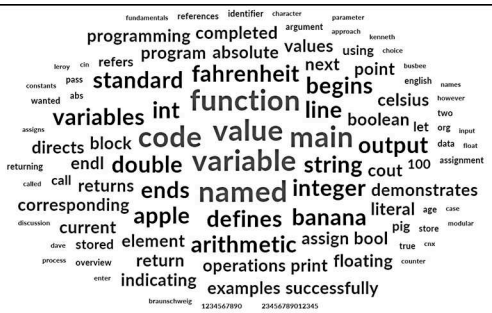
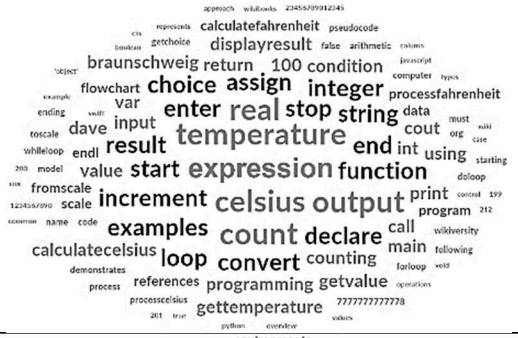
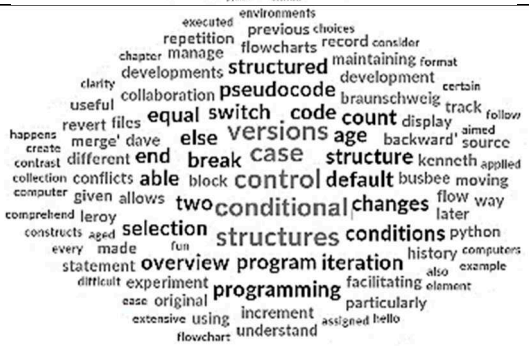
Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

The information in the form of text is complex to quantitatively visualize; for this reason, the standard word cloud was designed for entertainment, surprise, and self-expression; in recent times, word clouds have been optimally used for data analysis tasks [17].

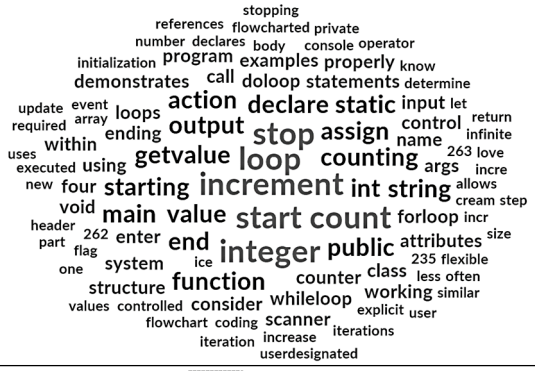
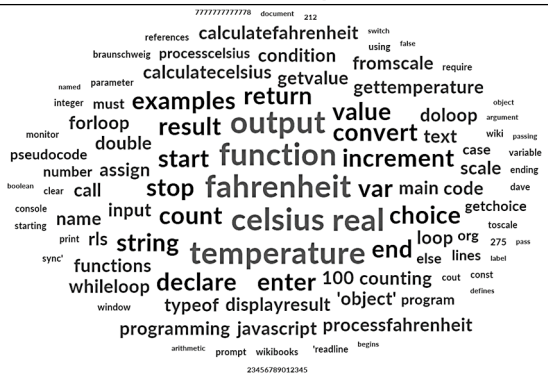
Then, we used the NVivo® 12 software to find the definitions of the essential knowledge objects. According to the findings, Table 1 lists the keywords and their corresponding weight in the semantic cloud. A word cloud is a visual representation of tags within large volumes of textual information, where the larger the tag, the more frequent the tag in the text. Tags in such clouds are keywords that are often grouped semantically [38].

Despite word clouds being mainly used on the web, they are gaining importance in the educational context as they are useful in visualizing keywords for representing topics related to teaching/learning concepts. Work with word clouds can stimulate linguistic and visual intelligence and the development of the ability to synthesize complex concepts [30].

Table 1. Word clouds based on essential knowledge objects.

Essential Knowledge Object	Word Cloud
Concept: variable Keywords: space, represents, represented, storage, abstraction, programming, type, assigned, language, memory, identifier, short, data, supported, corresponds, element, reserved, value, named, points.	
Concept: expression Keywords: type, arithmetic, logical, Boolean, values, built, data, symbols, calculation, calls, elements, returned, function, language, combination expressions, syntax, operations, commonly, numerical, depends, variables, returns, result, assign, increment.	
Concept: conditional Keywords: accurate, false, instruction, returned, Boolean, commonly, type, programming, conditions, case, equal, two, control, choices, conditions, flow.	

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

Essential Knowledge Object	Word Cloud
Concept: loop Keywords: series, program, structures, words, algorithmic, instructions, reserved, language, interpreted, repetition, computer, execution, controlled, increment, counting, initialization, counter, scanner.	
Concept: function Keywords: part, output, input, parameters, program, called, facility, tasks, value, handle, exported, instances, perform, return, computer, specific, platform, declare, call.	

Source: The authors.

The philosophical term "ontology" was first adapted to computing scenarios by [15] as an "explicit specification of a conceptualization" for the artificial intelligence community. Scholars in computer science have used ontologies for representing knowledge in several ways; so, an ontology comprises complex concepts within a knowledge domain [4]. We can use an ontology for specifying how concepts are related to each other with logical axioms expressed in a formal language, for example, OWL by W3C Description Logical Language for Semantic Web.

Computational ontologies are frameworks for understanding each concept with solid linguistic relationships. The conceptualization of knowledge objects includes the nature of such objects for establishing how such objects should be taught. For each knowledge object—variable, expression, conditional, loop, and function—an ontological definition is proposed as a conceptual basis for determining each object level of didactic transposition. Similarly, the teaching strategies presented based on didactic transposition theory include such ontological definitions [25].

Next, each knowledge object was expressed in the form of computational ontology using preconceptual schemas. Preconceptual schemas allow for the management of a controlled language within a specific knowledge domain, in this case, computer programming fundamentals. The aim of the use of preconceptual schemas was to graphically represent the theoretical constructs of focus in the form of computational ontologies. A pre-conceptual schema includes knowledge by using controlled language, regardless of the context in which they are used [39]. For specific contexts such as software engineering, "preconceptual schemas are computing models used on Requirements Engineering for representing a domain, which is near to stakeholders" [29]. Furthermore, preconceptual schemas use a simple notation, they are easy to understand, and they are adaptable to any knowledge domain [39]. Additionally, as preconceptual schemas easily represent ontological relationships, they were used in the conceptual definitions of the study. The simple notation of preconceptual schemas is summarized in Figure 4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

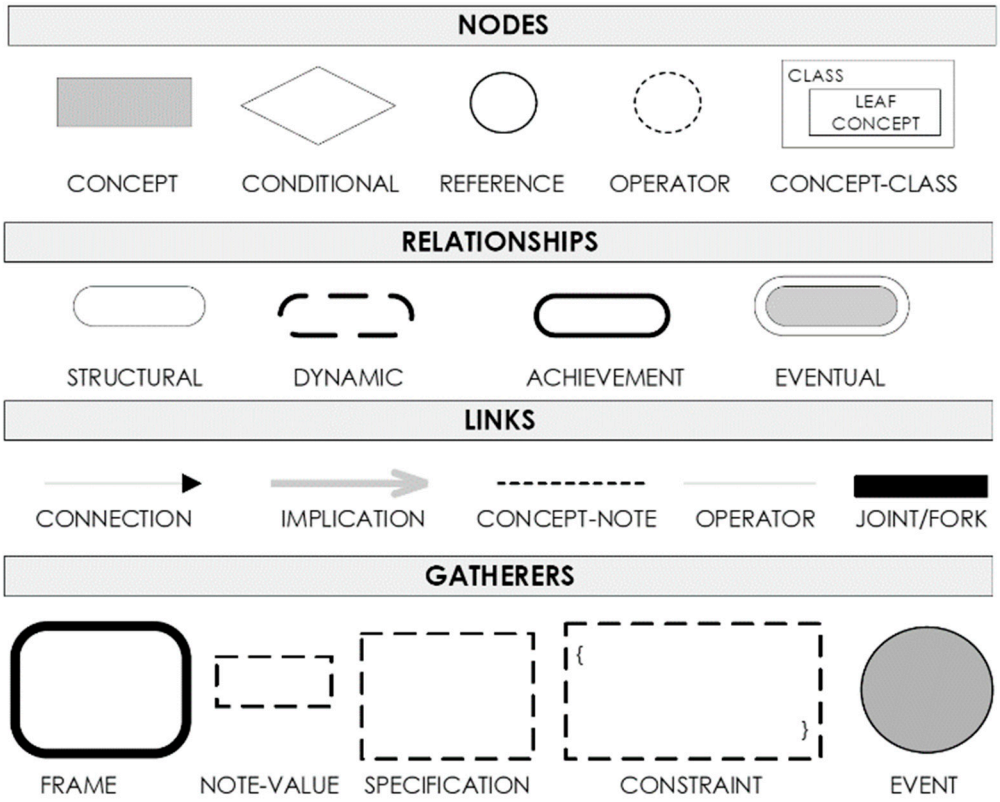


Figure 4. Preconceptual schema notation. Source: The authors based on [39].

Consequently, the theoretical constructs were formulated as computational ontologies supported by hermeneutical processes, which were based on the teaching experiences described by the experts. These theoretical constructs arose from linguistic analysis; they contemplate the different opinions from experts around each concept. Finally, the definitions that appeared according to the corresponding bibliography in the syllabus were also considered. As an example, Figure 5 depicts the ontology based on a preconceptual schema about the essential knowledge object called expression. We considered four more ontologies, one for each essential knowledge object; as such, we created an ontology about the essential objects called variable, conditional, loop, and function.

According to Figure 5, the knowledge object called expression involves a set of values, variables, constants, and operators for performing an arithmetic-logical calculation; operators have a hierarchy within an expression. In addition, how an expression is written depends on the syntax of a specific computer programming language. People in charge of teaching this knowledge object should generate a motivational environment for their students, and they should assume their role as professionals in matters of computer program coding.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

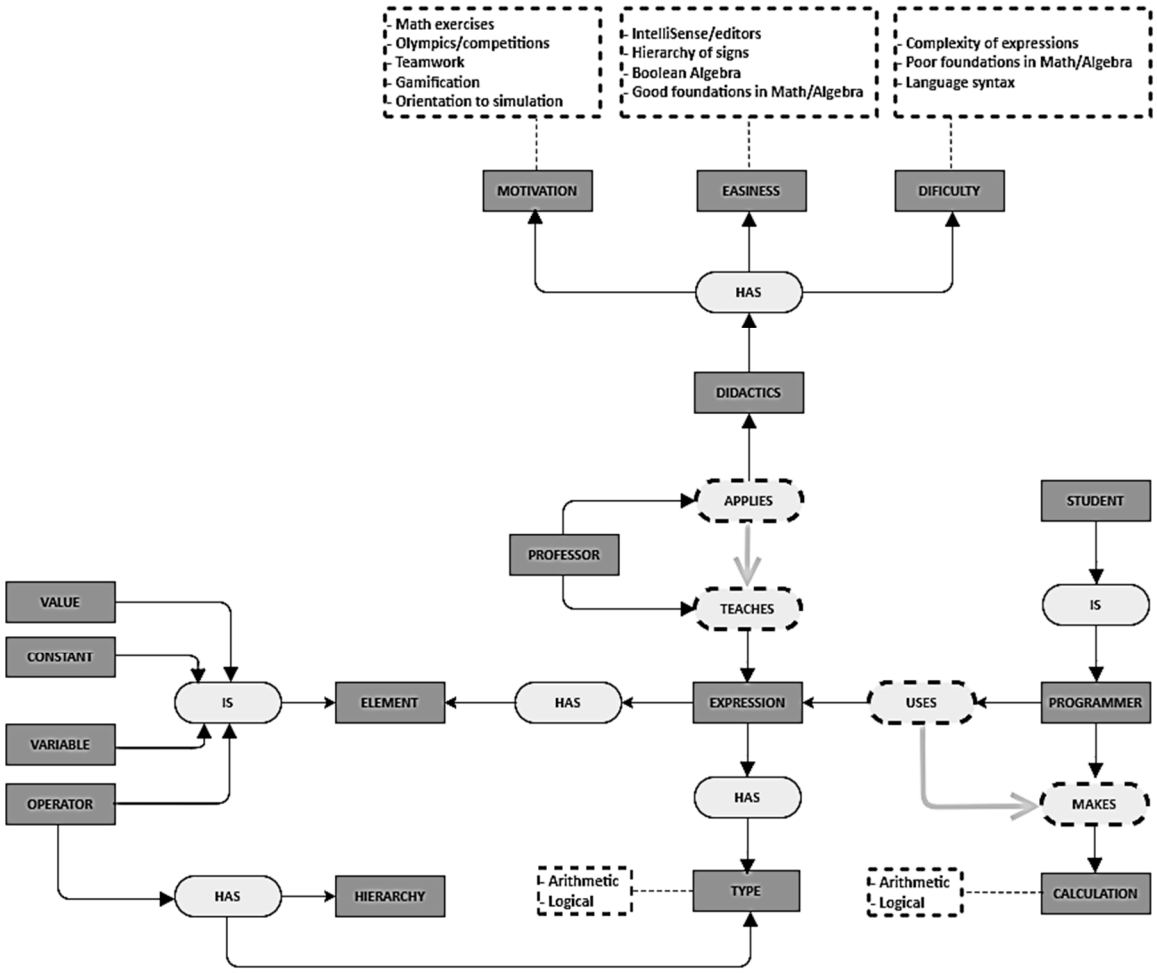


Figure 5. An ontology about the expression knowledge object. Source: The authors.

4. Didactic Transposition in *Extensa Sensu*

In this study, the Latin locution "in extensa sensu" was used to create an extended view of the phenomenon at a specific stage in computer science education. A comprehensive statement of the phenomenon of didactic transposition in computing education setting is depicted in Figure 6. In general, didactic transposition in extensa sensu comprises the phenomenon about the transposition from the knowledge objects (i.e., expert/scholar knowledge (toward teaching and assessment objects.

According to Figure 6, everything is part of the world of life. This term was intentionally used in this approach due to the strength of its meaning. [19] coined the German word "*Lebenswelt*" from a phenomenological perspective to indicate the complex relationships of subjectivities in the world aligned with society, culture, and the individual. In this challenging context called the world of life, we have a first cloud of knowledge; however, in this case, we speak of a cloud of expert knowledge in computer science.

According to this approach, expert knowledge is part of the body of knowledge of computer science. The context has a series of needs we should satisfy; additionally, the educational scenario, which is immersed in the world of life, involves the processes of didactic transposition to convert expert knowledge into knowledge to be taught. Finally, a new object was developed in such context called the assessment object for providing feedback to the object to be taught.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

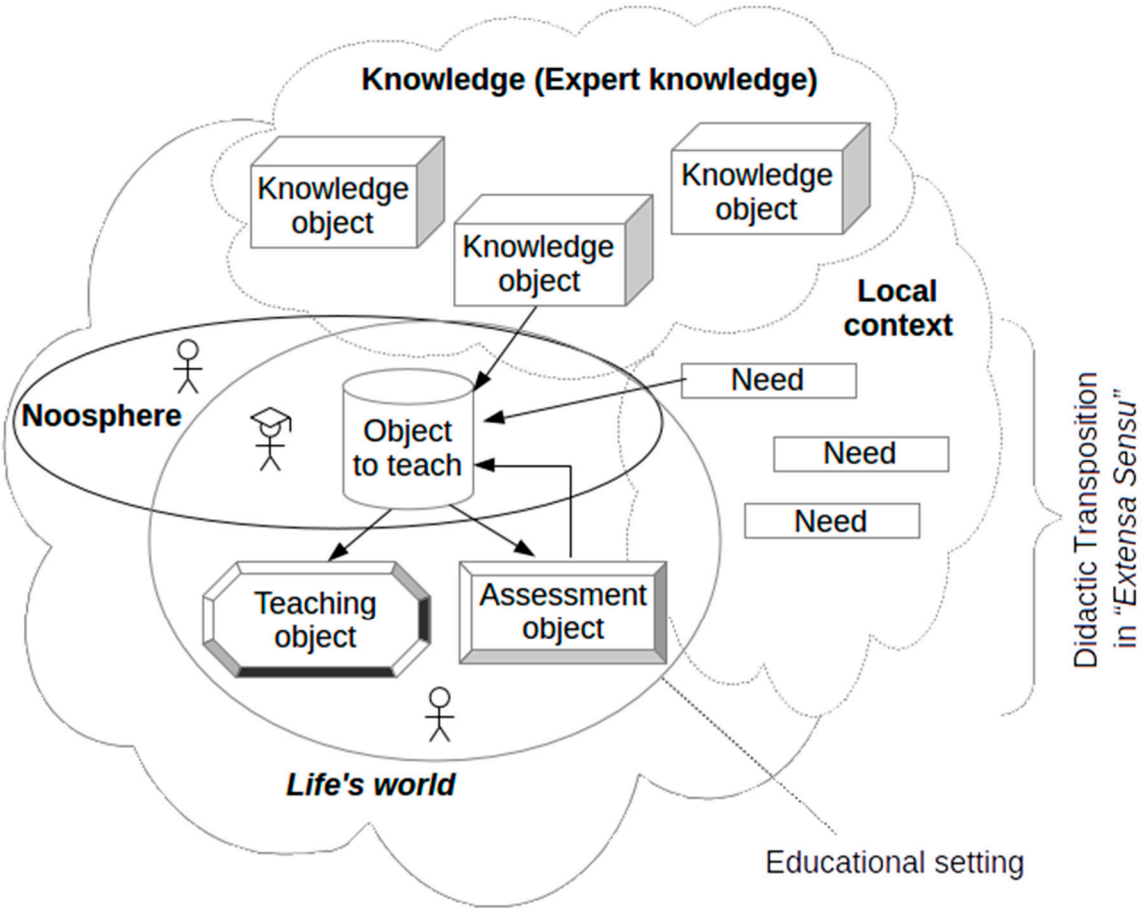


Figure 6. Didactic transposition *in* extensa sensu. Source: [21].

Considering the bodies of knowledge on computer science, we have some knowledge objects related to specific topics; we selected a set of knowledge objects is related to the programming fundamentals for this study. As a result, the first scenario of the didactic transposition phenomenon due to the guidelines recommended by ACM was applied; this scenario is depicted in Figure 7. The first transposition identified in this study is called "Transposition due to ACM curricula recommendations". In this part, the phenomenon of didactic transposition in extensa sensu begins with identifying the basic concepts and general terms belonging to specific topics in computer science. In this sense, a group of experts in the field especially from SIGCSE, established the body of knowledge for such a domain.

The participation of the industry in the construction of such bodies of knowledge is well known with regard to knowledge in computer science. In the document of curricular recommendations for computational sciences, the work team composed of the ACM and IEEE Computer Society says:

"Industry Insights: Unsurprisingly, industry feedback was quite different, but invariably, it was provided willingly—even enthusiastically—and with a deep conviction. It tended to confirm the importance of recruiting high-quality students who had to be in tune with a clever work ethic and receive a solid education in the fundamentals of the subject" [2].

What is recommended by the ACM is closely related to the intention of answering the questions: What to teach and what to teach for? However, some aspects are missed in this intent. For example, the answers to the questions as to how and why the original author/creator did it might seem less relevant than the ACM intention. According to the hegemonic economic model, the questions regarding what and why to teach are "more useful" nowadays. We depict the first scenario of the didactic transposition by the ACM curricula recommendations with the previous considerations in Figure 7.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

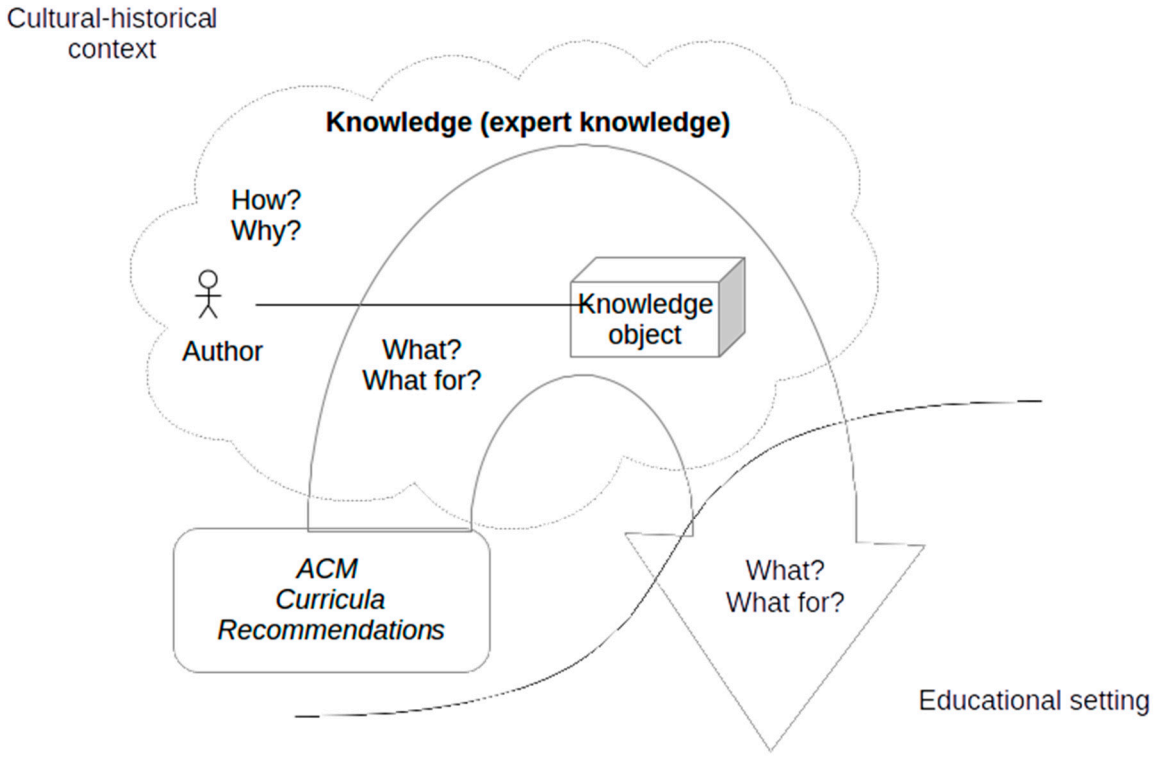


Figure 7. First transposition by ACM curricula recommendations. Source: The authors based on [20].

Considering the question regarding form whom knowledge is developed from a practical viewpoint, perhaps more questions can be raised in this regard: Is such knowledge for the interests of hegemonic societies supporting big tech companies? Is such knowledge for the common good? Knowledge has an altruistic motivation for humanity. However, a practical-merchant view is linked to the educational scenario [31]. In this scenario, the "most useful" content is implanted according to experts; however, some aspects about the historical-cultural context are missed where the author/creator of the knowledge object was immersed.

We addressed five knowledge objects outlined by scholar expertise in this study. We had special interest in analyzing their nature, origins, and why the authors created such knowledge objects. This information is essential to consider, far beyond its instrumental utility for providing additional value to what students learn from such knowledge objects.

Knowledge about how and why the author created a knowledge object beyond its instrumental utility was relevant in this study. Such knowledge could promote new learning in students because we can explore alternative methods of "reconstructing" the knowledge objects.

When a professor selects a knowledge object for teaching purposes, the second transposition occurs regarding educational settings. This second step is the transposition due to the noosphere; we depict this second step in Figure 8.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

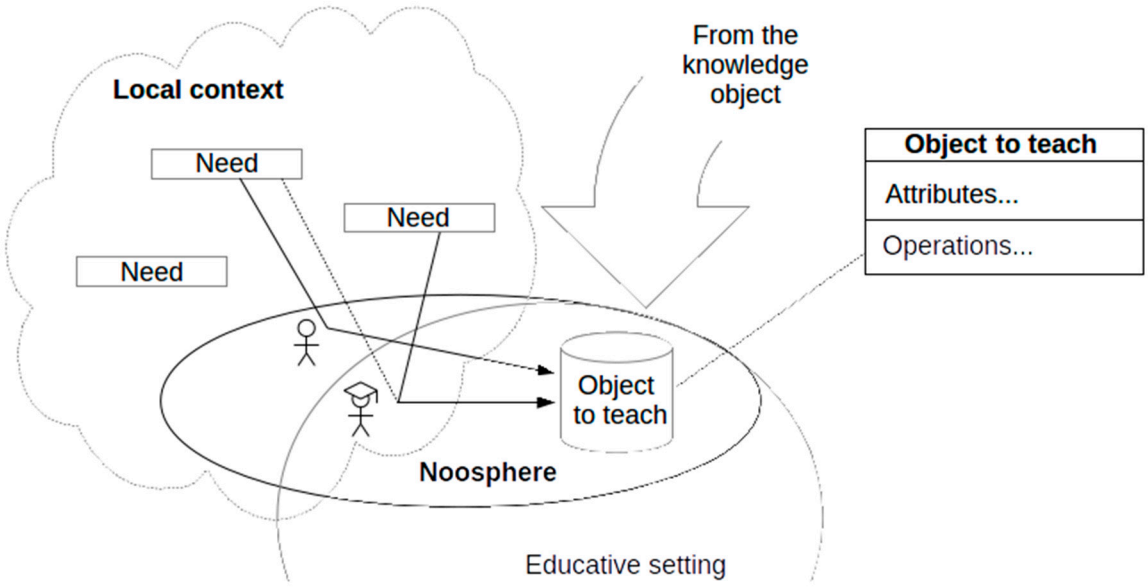


Figure 8. Second transposition due to the noosphere. Source: The authors based on [20].

The noosphere concept is crucial, as shown in Figure 8. According to [7], the noosphere is the sphere in which educational practices are conceived and designed. This sphere, or scenario, comprises people in charge of designing and implementing the curriculum for specific contexts such as administrative personnel, department heads, professors, experts in pedagogy, and industry consultants, among others. From the noosphere, some members make decisions by selecting knowledge objects to transform them into objects to be taught; simultaneously, they capture the needs of the context.

"The noosphere is made up simultaneously of representatives of the educational system and representatives of society: members of the teachers' association, professors, parents of students, specialists in the discipline who militate around their teaching, representatives of political organisms. The noosphere is then 'the sphere of people who think' to return to the expression of the author [Chevallard]" [14].

At this point, a new object is created and derived from the knowledge object; however, it is a different one, called the object to teach. This object is different from its predecessor. An object to teach has attributes and operations for facilitating the teaching process. In this sense, an object to teach includes unique features according to the learning model, such as active learning, problem-based learning, competency-based model, etc.; it is explicitly designed for addressing the needs of the context.

However, needs may vary according to the individual perspective; for example, in speaking about subjectivities, the personal interests of those who teach could be different from those who work in industry and those the productive sector. This new object has clearly defined educational goals due to the orientations of the noosphere. Thus, the essence of the second transposition is based on people's subjectivities in the noosphere: such subjectivity is already present when designing objects to teach.

After an object to teach is designed, a new object appears: the teaching object. We depict the third transposition due to the practice of teaching in Figure 9. At this point, some differences in what is designed and what is practiced arise. Consequently, "modern educational research is abundant with co-existing contradictory 'theoretical' frameworks; with the models and pedagogical approaches that can neither be empirically tested nor refuted" [26].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

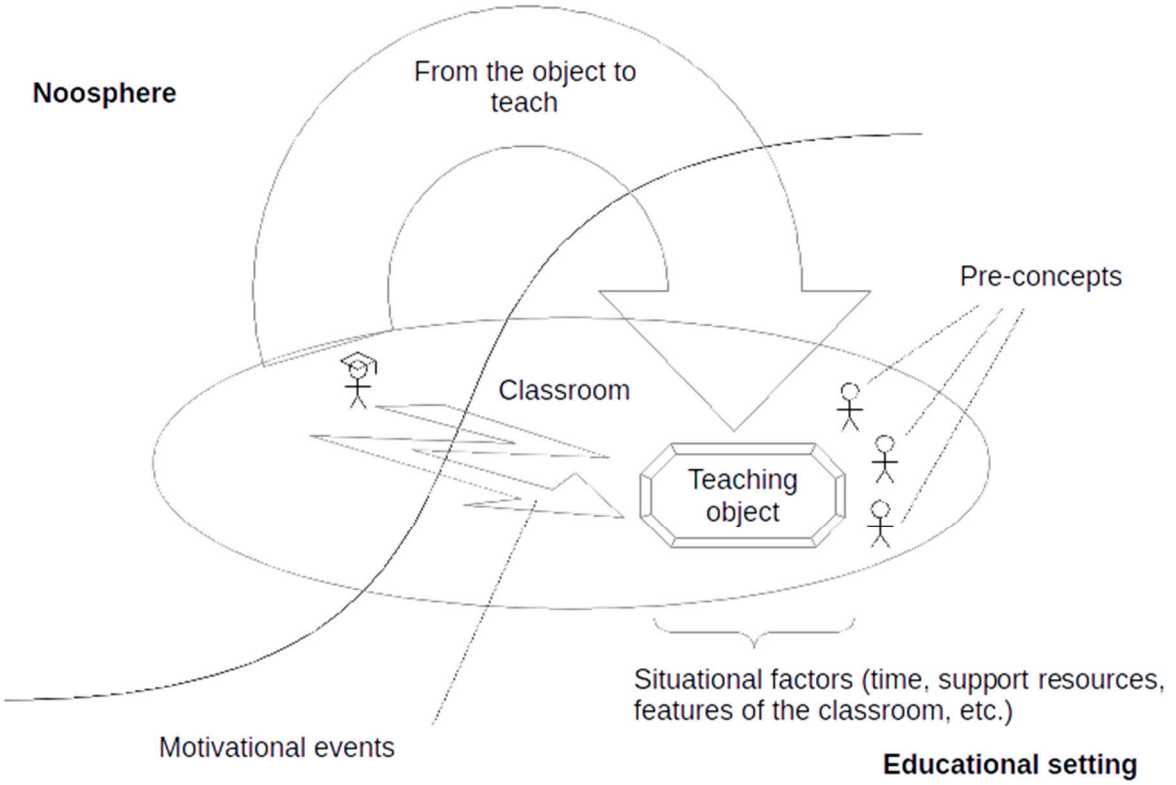


Figure 9. Third transposition due to the practice of teaching. Source: The authors based on [20].

As in the previous scenarios, the teaching object is different from its preceding object. Teaching objects are related to the objects to teach and vice versa, but they are different objects. This difference is the essence of the didactic transposition. The third transposition involves the object to be taught within a practical setting: the classroom. Teaching practices are transformed into real situations based on previous theoretical designs in classrooms. However, in reality, this encounter entails circumstantial factors such as time management, available resources, and features of the physical space (lighting, acoustics, ventilation, hygiene, climate conditions, etc.); consequently, the lack/deficiency of such factors can affect student learning.

In some cases, students have prior knowledge before attending a programming fundamentals course. Such prior knowledge is associated with previous training in specific programming languages. So, [12] argued the difficulty of teaching programming fundamentals to students with previous experience with some programming languages given their characteristics.

Such prior knowledge is associated with information on educational material related to computer programming. In some cases, a group of students might have prior knowledge about how to create computer programs; this is the case of those who receive some type of training related to the fundamentals of computer science in elementary and high school. Here, personal skills are strongly related to student learning; a theoretical foundation for the development of student skills was proposed by [37] during reflections on the zone of proximal development. [37] stated the difference between learning performance due to the presence/absence of external support. Therefore, personal competencies are critical factors in determining independent problem-solving ability. As such, the previous notion about the knowledge and skills of the students becomes an input for the development of teaching practices.

The previous knowledge and skills of the students should be integrated into the instance of a teaching object. At this point, the actual work in the classroom comprises the third level of the didactic transposition phenomenon because the integration of new elements produces changes in the initial object—the object to teach—that was selected to be taught. Because the teaching object differs from the object to teach, some elements are missed in this transposition. The mere interaction between the

 doi:10.20944/preprints202306.1483.v1

one who teaches and those who learn implies something can be lost due to the principle of the imperfection of the communicative action, which consciously or unconsciously manifests itself [16].

Effective communication is a critical success factor in teaching and learning processes at any level. The cognitive abilities, preknowledge, self-concepts, and interests of the students should be considered to achieve effective communication. This constitutes a challenge for those who teach [10].

Some theories regarding the fault of the communicative action can be used for explaining why a message from the one who can be misunderstood by those who receive such a message. For example, [23] indicated some "flaws" in the language according to empirical evidence; some of them include lexical ambiguity, syntactic ambiguity, morphological irregularity, extra-grammatical errors, morphological redundancy, movement, and locality conditions, among others. Student learning is affected by this dynamic behavior of the teaching object due to the transposition phenomenon in the real world.

Finally, the last part of this extended view of the didactic transposition is related to assessment. Assessment is recognized as a challenging activity in education. Assessment of learning is conceived as one of the fundamental processes in education intended to evaluate the work of its participants due to the collection and analysis of information for making decisions related to continuous improvement. The assessment of learning is a stage of joint work. The person who teaches is in charge of directing how such learning is assessed; moreover, positive experiences are reported when involving student self-assessment and peer-assessment in the assessment processes [8].

The assessment of learning represents a complex approach to the diagnostic functions of the previous knowledge of students. The assessment process fulfills a function of orientation and guidance for determining the state of student learning. Additionally, the assessment process fulfills a stimulus function for promoting skills in students.

The assessment object is the last transposed object in this approach. This object is based on a set of instruments allowing the assessment of student learning beyond qualification. Due to this nature of the object, the essence of the transposition at this point is reflected in the construction of assessment resources for qualifying student achievements in terms of the expected learning. Subjective elements should be carefully addressed.

The levels of responsibility students must face in the self-assessment tasks are also important. This implies actions promoting a culture of self-assessment and continuous improvement should be applied in the classroom. Students must have in-depth knowledge of their abilities in the classroom, which can be articulated in their life projects. The previous object's motivational events must be strongly coupled with teaching practices. So, we present the last transposition in Figure 10, which is called "Transposition due to evaluation and feedback". It was defined to create spaces for monitoring student learning. In such a situation, a new object is created, which depends on the transposed image of the teaching object: the assessment object.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

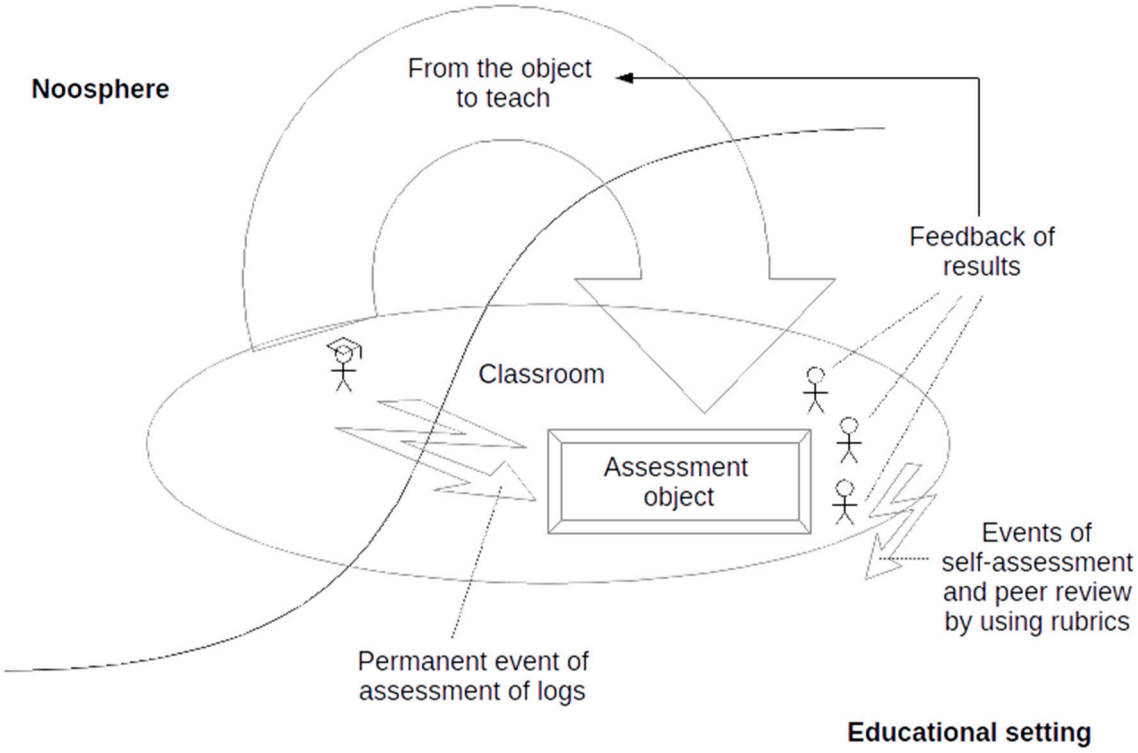


Figure 10. Fourth transposition due to evaluation and feedback. Source: The authors based on [20].

As with the teaching object, the assessment object has its events; in this case, self-assessment and peer review events are linked to rubrics, as some empirical studies in the field of teaching programming fundamentals have recommended both individual work and working in pairs inside and outside the classroom: tasks, project development, readings analysis, consultation work, etc. A study on the benefits of pair programming for achieving individual programming skills revealed students feel more confident in their careers after using such learning techniques [5].

The teacher has the responsibility to execute a permanent action on the assessment process by changing a point of view. Learning assessment includes a collection of isolated evaluative events: a quiz, an exam, a questionnaire, an assignment, etc. Additionally, assessment implies a holistic, integral, and continuous process including some aspects of the formation of the human being in relation to specific disciplinary learning.

Human factors are always relevant in the assessment of student attitudes. The evaluation process involves a high degree of subjectivity. Feedback based on student achievement is a new input for strengthening and improving the design of the objects to be taught, and this is part of the dynamic educational world. In the end, didactic transposition in extensa sensu comprises a theory, a method, and a model. The latter model is depicted in Figure 11.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

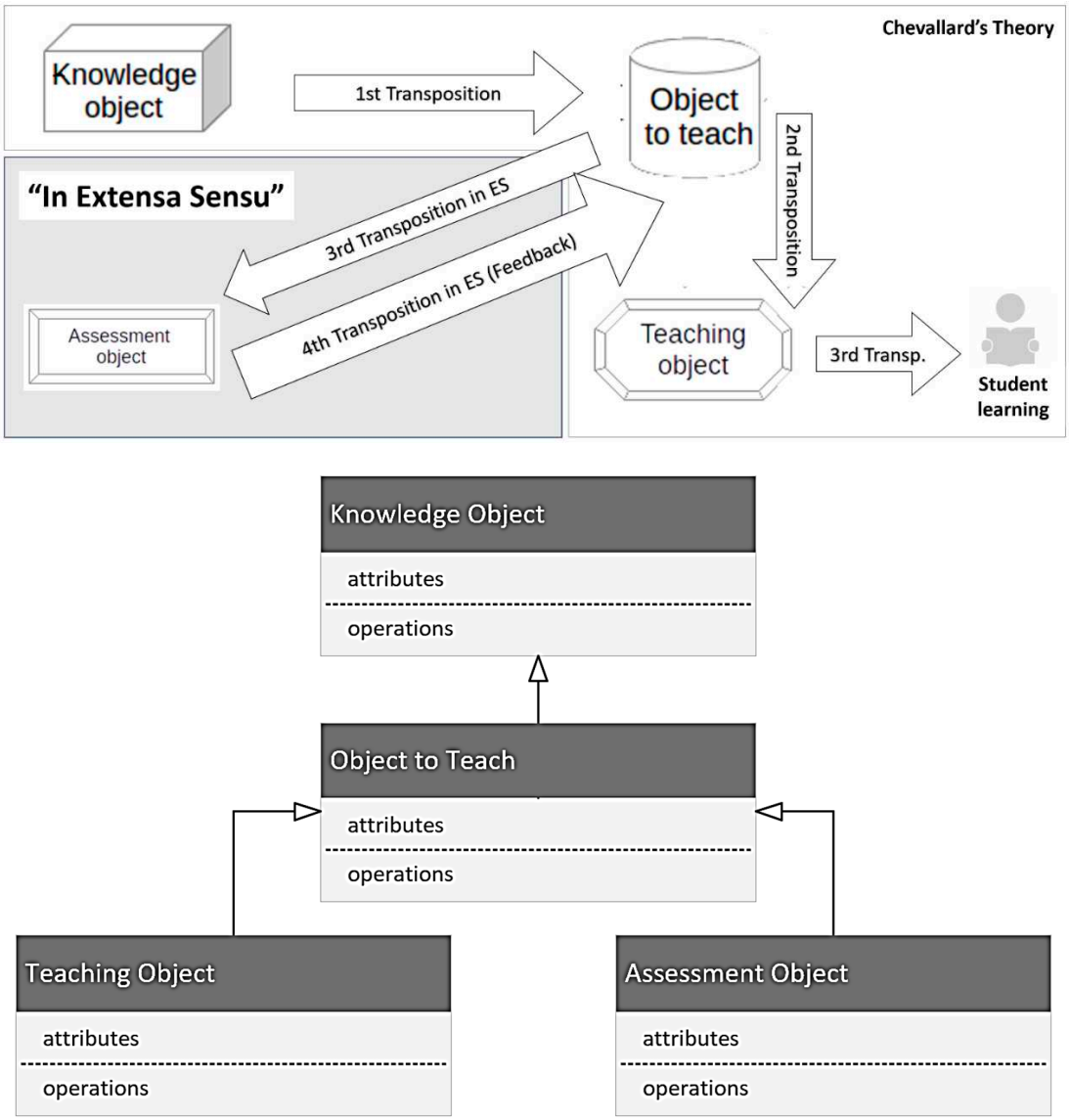


Figure 11. A model based on didactic transposition in extensa sensu. Source: The authors based on [20].

We created a method consistent with the model based on didactic transposition in extensa sensu. This method is based on the steps performed in this study: a sequence of activities with high incidence in computational linguistic techniques to achieve success. The main point involves using corpus linguistics techniques to analyze large amounts of textual information. In this regard, some qualitative data analysis tools are highly recommended. To represent such a method, Figure 12 depicts an activity diagram for a general view of the method based on didactic transposition in extensa sensu.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

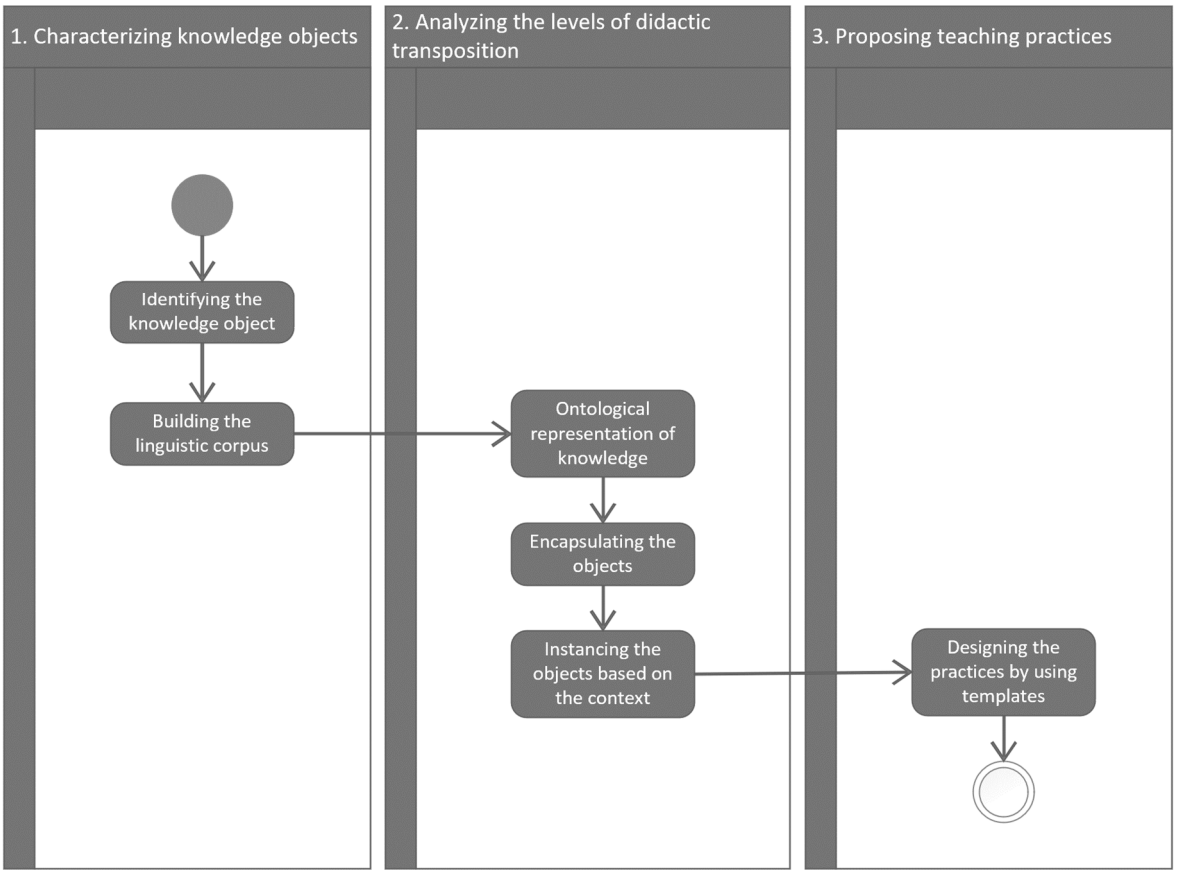


Figure 12. A method based on didactic transposition in extensa sensu. Source: The authors based on [20].

This method, based on didactic transposition in extensa sensu, allows the creation of a scenario for planning how to teach computer programming fundamentals. This method is composed of three main stages: characterizing knowledge objects, analyzing the levels of didactic transposition, and proposing teaching practices.

In the first stage, based on characterizing the knowledge objects, the main aim is identifying the knowledge objects to teach in a course. In this stage, the creation of a linguistic corpus is highly recommended. Software for creating and analyzing the linguistic corpus is useful in this regard.

The second stage of the method is based on analyzing the levels of didactic transposition. In this stage, the knowledge representation of the objects is relevant for establishing the pathway to teach those objects.

Finally, to propose teaching strategies within a stage of computer science education, the use of a design template for these strategies is proposed considering the relevant theory and concepts. We depict the template for designing teaching practices based on didactic transposition in extensa sensu in Figure 13.

A teaching practice according to the developed theory must formulate four objects to be transposed: the knowledge object, the object to teach, the teaching object, and the assessment object. As such, the four transpositions can be achieved through the application of the proposed theory.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 June 2023 doi:10.20944/preprints202306.1483.v1

Figure 13. A template for the design of teaching practices based on didactic transposition *in extensa sensu*. Source: The authors based on [20].

5. Conclusions

Once the study was complete, the literature review and analysis of the collected information led to a series of activities for creating teaching practices in computer programming fundamentals. These practices are based on an extended vision of didactic transposition, the seminal theory proposed by [7].

Defining the group of people linked to the noosphere is essential. Professors and program directors (heads or their equivalent) must be present; additionally, some professionals from the industry sector can be present as much as possible, together with people with experience in education and pedagogy. Here, having knowledge of the discipline is not enough for proposing the objects to be taught, which will later become teaching objects. In addition to disciplinary knowledge, some aspects should be considered for articulating the objects to be taught within an educational setting located within a specific context and using appropriate teaching strategies.

Based on a systematic literature review, active learning and problem-based learning models include activities within the framework of this proposed method; however, the integration of the

 doi:10.20944/preprints202306.1483.v1

activities of such models is just a part of the method. The actions of any other learning models can be considered in this method.

Additionally, the learning scenarios where students can interact within the classroom are challenging according to our framework due to the complexity of defining actions for maintaining student interest when they are active within classroom dynamics. The challenge is the creativity and imagination of the teaching staff in motivating the students the programming fundamentals courses. In this framework, this type of challenge is addressed by the concept of a motivational event.

Educational resources play an important role in supporting teaching strategies, according to our study; however, motivation promotes student interest in learning. Motivated learners are individuals with a strong interest in moving forward with their learning process. As such, the motivational events intend to create a warm and positive atmosphere that is conducive to focusing efforts on learning.

Engaging experiences can be added to this framework by incorporating new skills into motivational events by critically analyzing source code written by peers in a competitive collaboration environment. Such situations have produced reliable results in terms of motivation for learning according to the experiences described by their authors.

Didactic transposition reduces the complexity of abstraction of the objects of knowledge. The didactic transposition in extensa sensu contributes to the planning of strategies of activities typical of teaching in computer science education scenarios; in the case in this study, this is computer programming fundamentals.

People involved in the noosphere help to figuratively build bridges connecting scholar knowledge with students seeking to learn by reducing the abstraction complexity. Thus, didactic transposition applies for such purposes.

The didactic transposition in extensa sensu comprises the main contribution of this study: a theoretical construct of how to transpose objects of expert knowledge in the field of computer science within a noosphere of context and involving a new object to starting from the object to teach the assessment object.

The assessment object involves an agent of the permanent learning evaluation, which was incorporated as a fundamental principle for critical review. The major goal is student learning; as such, the assessment object has a high responsibility for achieving the goal. According to this framework, the assessment object arises from the last transposition of the teaching object. It represents a more substantial challenge for those participating in its design and deployment. Thus, the assessment object closes the transposition circle, always allowing space for improvements.

The proposed template for the design of teaching practices is an instrument that can facilitate the understanding of the didactic transposition processes of the objects involved. It is a form that synthesizes all the proposed theory about didactic transposition in extensa sensu.

In addition, we suggest the use of computational linguistics techniques for analyzing large amounts of textual information. A linguistic corpus is crucial in applying such techniques. Additionally, preconceptual schemas are appropriate for representing computational ontologies to help organize knowledge. In this study, preconceptual schemas and computational linguistics techniques enabled the completion of our specific objectives.

6. Future Work

Because the didactic transposition in extensa sensu is a theory, a method, and a model, some applications can be developed in the educational setting. The most important aspects can be achieved in designing didactics for teaching programming fundamentals for several scenarios including elementary school, where computational thinking is essential.

The teaching of computational thinking before starting higher education can be explored. The didactic transposition in extensa sensu is a valuable element in this regard. In addition, the construction of software tools for supporting the processes of teaching computer programming fundamentals under an approach based on didactic transposition in extensa sensu is a promising field that could be exploited in academic and research settings.

References

1. ACM & IEEE Computer Society. Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science. 2013. Available online: http://www.acm.org/education/CS2013-final-report.pdf
2. ACM & IEEE Computer Society. Computing Curricula 2020. 2021. Available online: https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2020.pdf (accessed on).
3. Aki, O.; Güllü, A.; Kaplanoğlu, E. An application for fundamental computer programming learning. *Procedia-Soc. Behav. Sci.* **2015**, *176*, 291–298. https://doi.org/10.1016/j.sbspro.2015.01.474.
4. Barbagallo, A.; Formica, A. ELSE: An ontology-based system integrating semantic search and e-learning technologies. *Interact. Learn. Environ.* **2016**, *25*, 650–666. https://doi.org/10.1080/10494820.2016.1172240.
5. Braught, G.; Wahls, T.; Eby, L.M. The case for pair programming in the computer science classroom. *ACM Trans. Comput. Educ.* **2011**, *11*, 1–21. https://doi.org/10.1145/1921607.1921609.
6. Campos, A. Métodos mixtos de investigación. In *Integración de la Investigación Cuantitativa y la Investigación Cualitativa*; Cooperativa Editorial Magisterio: Bogotá, Colombia, 2009.
7. Chevallard, Y. La transposition didactique. In *Du Savoir Savant au Savoir Enseigné*; La Pensée Sauvage: Grenoble, France, 1985.
8. Costelloe, L.; Egan, A. "Because, as a teacher, giving feedback and assessment is actually really difficult": Using self-and peer-assessment to develop Higher Education teachers' skills in assessment and feedback. In Proceedings of the 6th International Conference on Higher Education Advances (HEAd'20), València, Spain, 2–5 June 2020.
9. de Deus, W.; Barbosa, E. An Exploratory Study on the Availability of Open Educational Resources to Support the Teaching and Learning of Programming. In Proceedings of the 2020 IEEE Frontiers in Education Conference (FIE), Uppsala, Sweden, 21–24 October 2020; pp. 1–9.
10. Duta, N.; Panisora, G.; Panisora, I. The Effective Communication in Teaching. Diagnostic study regarding the academic learning motivation to students. *Procedia-Soc. Behav. Sci.* **2015**, *186*, 1007–1012.
11. Enström, E. On Difficult Topics in Theoretical Computer Science Education. Ph.D. Thesis, KTH School of Computer Science and Communication, Stockholm, Sweden, 2014.
12. Friedman, F.; Koffman, E. *Problem Solving, Abstraction, and Design Using C++*, 6th ed.; Pearson: Philadelphia, PA, USA, 2011.
13. Gadamer, H. *Philosophical Hermeneutics*, 30th Anniversary ed.; University of California Press: Berkeley, CA, USA, 2008.
14. Gómez, M. La transposición didáctica—Historia de un concepto. *Latinoam. Estud. Educ.* **2005**, *1*, 83–115.
15. Gruber, T.R. A translation approach to portable ontology specifications. *Knowl. Acquis.* **1993**, *5*, 199–220. https://doi.org/10.1006/knac.1993.1008.
16. Habermas, J. *The Theory of Communicative Action*; Beacon Press: Boston, MA, USA, 1984.
17. Hearst, M.A.; Pedersen, E.; Patil, L.; Lee, E.; Laskowski, P.; Franconeri, S. An Evaluation of Semantically Grouped Word Cloud Designs. *IEEE Trans. Vis. Comput. Graph.* **2019**, *26*, 2748–2761. https://doi.org/10.1109/tvcg.2019.2904683.
18. Hunston, S. Corpus Linguistics. In *Corpus Approaches to Idiom*; Elsevier Ltd.: Birmingham, UK, 2006; pp. 234–248.
19. Husserl, E. *La Crisis de las Ciencias Europeas y la Fenomenología Trascendental*; Una introducción a la filosofía fenomenológica, Editorial Crítica: Barcelona, Spain, 1991.
20. Insuasti, J. Enseñanza de los fundamentos de programación de computadoras y trasposición didáctica. **2021**, Editorial Universidad de Nariño: Pasto, Colombia.
21. Insuasti, J; Dodero, J. About Didactic Transposition: Teaching programming fundamentals at different levels of the school system. **2015** International Conference on Learning and Teaching in Computing and Engineering, Taipei, Taiwan, 2015, pp. 91-94, doi: 10.1109/LaTiCE.2015.53.
22. Jancec, L.; Vujicic, L. Project Algorithmic Thinking Skills through Play-Based Learning for Future's Code Literates. In Proceedings of the 2021 44th International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 27 September–1 October 2021; pp. 641–644.
23. Kinsella, A.; Marcus, G. Evolution, Perfection, and Theories of Language. *Biolinguistics* **2009**, *3*, 186–212.
24. Krpan, D.; Mladenović, S.; Rosić, M. Undergraduate Programming Courses, Students' Perception and Success. *Procedia—Soc. Behav. Sci.* **2015**, *174*, 3868–3872.
25. Mercier, C.; Roux, L.; Romero, M.; Alexandre, F.; Viéville, T. Formalizing Problem Solving in Computational Thinking: An Ontology approach. In Proceedings of the 2021 IEEE International Conference on Development and Learning (ICDL), Beijing, China, 23–26 August 2021; pp. 1–8.
26. Milner-Bolotin, M. Evidence-Based Research in STEM Teacher Education: From Theory to Practice. *Front. Educ.* **2018**, *3*, 92. https://doi.org/10.3389/feduc.2018.00092.
27. Mutero, T.; Mawere, G.; Kwenda, C. An Investigation on the Challenges Faced in Teaching and Learning Computers in Higher Education. *Int. J. Eng. Res. Technol.* **2018**, *7*, 467–470.

28. National Forum for the Enhancement of Teaching and Learning in Higher Education. Student Non-Completion on ICT Programmes. Briefing Paper 1. 2015. Available online: https://hub.teachingandlearning.ie/wp-content/uploads/2021/06/NF-2015-Student-Non-Completion-on-ICT-Programmes.pdf (accessed on).
29. Noreña, P.; Zapata, C. Simulating Events in Requirements Engineering by Using Pre-conceptual-Schema-based Components from Scientific Software Domain Representation. *Adv. Syst. Sci. Appl.* **2022**, *21*, 1–15.
30. Padmanandam, K.; Bheri, S.; Vegesna, L.; Sruthi, K. A Speech Recognized Dynamic Word Cloud Visualization for Text Summarization. In Proceedings of the 2021 6th International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 20–22 January 2021; pp. 609–613.
31. Ranson, S. Public Education for the Common Good. *Forum* **2019**, *61*, 157–164. https://doi.org/10.15730/forum.2019.61.2.157.
32. Ricœur, P. *Hermeneutics: Writings and Lectures*; Polity Press: Cambridge, UK, 2013.
33. ShanghaiRanking. Global Ranking of Academic Subjects—Computer Science and Engineering. 2020. Available online: https://www.shanghairanking.com/rankings/gras/2020/RS0210 (accessed on).
34. Takacs, R.; Horvath, Z. Dropping-out prevention of computer science students: Developing studying, thinking, and soft skills among students using training programs. In Proceedings of the 11th International Technology, Education and Development Conference, Valencia, Spain, 6–8 March 2017; pp. 7094–7099. https://doi.org/10.21125/inted.2017.1644.
35. Tiberghien, A.; Vince, J.; Gaidioz, P. Design-based Research: Case of a teaching sequence on mechanics. *Int. J. Sci. Educ.* **2009**, *31*, 2275–2314. https://doi.org/10.1080/09500690902874894.
36. Tutillo-Arcentales, I.; Rebollar-Morote, A. Teaching model of problem-solving Programming Fundamentals. *Maest. Soc.* **2016**, 160–171.
37. Vygotsky, L. Zone of Proximal Development. In *Mind on Society: The Development of High Psychological Processes*; Harvard University Press: Cambridge, MA, USA, 1987.
38. Xie, Y.; Lin, S.-Y. Using word clouds to support students' knowledge integration from online inquiry: An investigation of the process and outcome. *Interact. Learn. Environ.* **2018**, *27*, 478–496. https://doi.org/10.1080/10494820.2018.1484774.
39. Zapata, C., Arango, F. & Gelbukh, A. (2006). "Pre-conceptual Schema: a UML Isomorphism for Automatically Obtaining UML Conceptual Schemas," Lecture Notes in Computer Science (Artificial Intelligence Bioinformatics), vol. 4293, no. 65, pp. 27–37.
40. Zingaro, D. Examining interest and grades in computer science 1: A study of pedagogy and achievement goals. *Trans. Comput. Educ.* **2015**, *15*, 1–18.