Informatics in Education, 2003, Vol. 2, No. 1, 123–138

How Logo has Contributed to the Understanding of the Role of Informatics in Education and its Relation to the Learning Process

José Armando VALENTE
Department of Multimeios and Nied, Unicamp & Ced, PucSP
Nied–Prédio V da Reitoria, 13083-773 Campinas, SP Brazil
e-mail: jvalente@unicamp.br

Received: March 2003

Abstract. The use of informatics in education has provided many contributions to the understanding of teaching and learning processes. First, it made possible the distinction between instructionism, seen as transmission of information, and constructionism, as the process of knowledge construction that takes place when a learner produces a meaningful product through the use of computers.

Second, programming activity, especially with the Logo language, has helped to understand how knowledge is constructed in the learner-computer interaction. The article shows that this understanding has evolved over the course of the years. Initially the knowledge representation aspect was emphasized. Later the program development process began to be seen as a cycle of actions, *description-execution-reflection-debugging-description*. Finally, a spiral is shown to be the best model to represent the relation of these actions in the knowledge construction process.

The article explores the cycle and the spiral models to discuss the role of each of the actions and to explain how knowledge is constructed based upon several concepts used by Piaget and Papert, particularly reflection and debugging.

Key words: Logo language, informatics in education, learning process, construction of knowledge, learning spiral.

1. Introduction

The use of computers in education dates back to as early as the mid 1950's, when the first computers with programming and information storage capacity came on the market. In 1955 computers were used in graduate courses for problem solving, and in 1958 as teaching machines at the IBM Watson Research Center at the University of Illinois – Coordinated Science Laboratory (Ralston and Meek, 1976, p. 272).

However, the emphasis at that time was essentially on these machines' information storage and transmission capacities. Actually, it was an attempt to implement the teaching machine as conceived by Skinner. In the mid 1960s, the development of Logo constituted an alternative to the use of informatics in education. The possibility of programming the computer was seen as a way of allowing the user to pass information to the computers.

This opened up new avenues in the area of informatics in education, first, helping to establish a clear view between two educational approaches: the instructionism and the constructionism. Second, Logo programming has provided an important contribution to the understanding of the computer's role in the process of knowledge acquisition.

During these more than three decades much has transpired in the use of computers in education. Educational software has become much more sophisticated and the polarity between instructionism and constructionism is not as clear as it used to be. Also, our understanding of the computer's role in the construction of knowledge process has also changed. In the early days, Logo programming was seen as a way of representing thinking processes. Later it was clear that the computer could "execute" this thinking, offering results that could be used to improve this representation (the programming) and, consequently the knowledge used to produce this program. With this it was possible to see programming as a cycle of actions and to understand the contribution of each action in the process of knowledge construction. Today, all the facilities implemented in Logo and our better understanding about knowledge construction have shown that the idea of cycle captures a series of actions the student does although the knowledge construction may be taking place through an expanding spiral.

In this article I describe the instructionism and constructionism approaches as well as how Logo programming has contributed to the understanding of the process of knowledge acquisition.

2. Instructionism X Constructionism

Computer based activities can be designed to transmit information to the student and, therefore, reinforce the instruction process. This approach has been called instructionism (Papert, 1986). On the other hand, computer activities can be designed to create conditions for the student to build her/his own knowledge, allowing what Papert has called the constructionist approach (Papert, 1986).

When the computer transmits information to the student, the computer assumes the role of a teaching machine and the pedagogic approach is that of computer aided instruction. This approach is rooted in traditional teaching methods, where worksheets and textbooks are exchanged for computers. The software implementations of this pedagogic approach are the tutorials and drill-and-practice.

On the other hand, when the student is programming, the computer becomes a machine to be taught, offering situations for the student to describe the problem solution in terms of programming languages. The construction of knowledge arises through the student's search for new information within the subject matter or about the computer concepts, or for new strategies to solve the problem. All this knowledge is needed to improve her/his programming quality and, thus, her/his already existing level of knowledge.

Papert first used the term "constructionism" in a proposal to the National Science Foundation (Papert, 1986). He introduced this term to explain the difference between Piaget's construction of knowledge idea (constructivism) and the construction of knowledge

that can happen when the learner constructs a meaningful product such as a work of art, a research report or a computer program. In Papert's definition of constructionism there are two ideas that make this type of construction of knowledge different from Piaget's constructivism. One is that the learner is building something, that is, learning through hands on experience. Second, is the fact that the learner is building something that is meaningful to him. However, a significant difference that contributes to these two ways of constructing knowledge is the presence of the computer – the fact that the learner is building something by using the computer as a tool (Valente, 1994). As will be discussed later, the computer requires certain actions that are very effective in the process of constructing knowledge.

However, it is important to mention that promotion of learning, seen as memorization of information or knowledge building, is not a property of a particular software, but is related to the kind of interaction the student has with her/his computer activity and therefore how the computer is used. As Piaget has shown, the comprehension level of a particular concept is related to the level of interaction the learner has with the object and not only to the object in itself (Piaget, 1978). Some software present characteristics that favor comprehension, as in the case of programming; others, which lack certain characteristics, require a greater involvement of the teacher or the student, so as to create situations that complement the software usage in order to favor comprehension, as in the case of tutorials.

Thus, in order to understand the role of computers in education we should not concentrate too much on the software. Any attempt to analyze the software out of the context of its use can result in a simplistic view of the software and its potential. We should observe how the student is using the software to implement her/his activity and whether the teacher is interacting with the student challenging her/him in the knowledge construction process.

Another important observation to make is that our understanding of the role of computers in the knowledge construction process is changing as we understand more about learning and as new features are implemented in software designs. This was the case with Logo.

3. Logo Programming as a Way of Representing Knowledge

Computers have provided us with the possibility of penetrating into the user's mind in ways that we never had before. Artificial Intelligence and Cognitive Science have benefited from this enormously. And in the Logo community, it was common to hear that Logo provided us with a "window into the child's mind" (Weir, 1987). The argument was that a Logo program has imbedded in it the student's concepts, strategies and styles, which can only be the product of her/his mind. Looking at this program we could have ways of understanding how his mind worked in the process of developing this program, and the level of knowledge, strategies and style used.

It is clear that any intellectual activity can be seen as a window into the mind. However, there are important features in programming that allow us to penetrate into one's

mind that we did not have before. In the article “Is Programming Obsolete?” (Clements and Meredith, 1993) gave a summary of an American Educational Research Association meeting in which several speakers presented different views of programming. For example, programming is “the ’best ever’ representational support for cognitive activities” according to Andrea diSessa. Mitchel Resnick is cited as describing programming “as an expressive medium.”

The argument for programming as a representational tool has to be understood in a historical context, related to the kind of computers available to education. In the early 1980, most of the computers used in schools were Apples or MSX. They were very simple machines, reducing the usage of computers in education to two main alternatives: Logo programming or educational software such as tutorials, drill-and-practice, simulations or games. These educational software were seen as ready made programs developed to pass a particular content. Logo programming was the only alternative students had to create their own programs and therefore, a way students had to express their ideas to the computer using the Logo programming language.

The possibility of expressing ideas by means of a formal language (commands of the Logo language) was compared to other representational activities in other knowledge domains. For example, we can think musically and represent these ideas by means of a musical notation; or think about a physical phenomenon and represent it through an algebraic expression. However, what happens in these cases is that these notations are complex and their acquisition becomes pre-requisite to the capability to represent ideas in these domains. We can learn about how to solve a trigonometric equation and not about how to represent a phenomenon through this equation.

In Logo, the process of acquiring commands, and the representation of ideas using these commands, are taking place simultaneously. In order to learn about a command the student has to use it. This produces a result on the screen that leads the learner to wanting to solve a particular problem. This demands the coordination of this command with other ones that need to be acquired. Thus Logo programming does not happen first learning all the language commands and then using them to solve a problem.

Another advantage of Logo with respect to the process of knowledge representation is that a program can be used as a transition to understand abstract and complex concepts such as, for example, the concept of function in mathematics. The student can teach the Turtle to draw a square with variable sizes, using the **to** command and the notion of procedure with argument. In this case the program is named **square** with the argument **x**, as shown in Fig. 1.

For each value attributed to **x**, a square is drawn with the correspondent size. Thus, **square 10** draws a square of that has 10 Turtle steps for each side. The **square** program can be seen as a mathematical function that maps all integer numbers into squares of corresponding sizes. The concept of mathematical function can be represented in a very practical and concrete manner, making its comprehension easier. Thus the possibility of expressing ideas through Logo programming helps not only to understand about the problem being solved but about the concepts involved, and about how the problem can be represented. In fact, the program can be seen as a representation of the student’s thinking by means of a formal language.

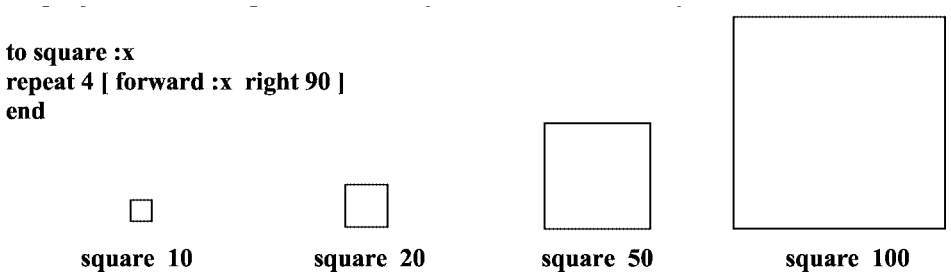

Fig. 1. **square** program using the variable **x** to change the size of the square.

However, it is important to understand that Logo programming allows much more than representing ideas. In fact, these "ideas" can be executed by the computer as the program is executed by the machine, producing a result, which can be confronted with the original idea. If the result obtained is not what was expected, the student has to revise her/his thinking, improving it and therefore, constructing new knowledge. With this was born the idea that Logo programming happens through a cycle of actions that helps in the process of knowledge construction.

4. Logo Programming as a Cycle of Actions

When the student is programming the computer s/he is engaged in a process of problem solving. By doing so s/he is using concepts and strategies that s/he already has or is searching for new information, processing it and transforming it into knowledge, which in a certain way are all made explicit in the program. The analysis of the programming activity using a language like Logo, more specifically the graphic part of Logo, allows the identification of different actions that happen in terms of the cycle of description-execution-reflection-debugging-description, which the student does and which are important in order to understand how the program is produced:

- *Description of problem solution in terms of the programming language*. This means using the entire knowledge structure (concepts involved in the problem, strategies to apply these concepts, concepts about the computer, the language, etc.) to represent and describe the steps in the solution of a problem in terms of the programming language;
- *Execution of this description by the computer*. The description of the way the problem is solved in terms of a programming language that can be executed by the computer. In the case of Logo graphics, the Turtle behaves according to each command, presenting on the screen the result in the form of a graphic. The student can look at the figure as it is being built on the screen, and at the final product, reflect on this information;
- *Reflection on what has been produced by the computer.* Reflection on results given by the computer can lead the learner to compare what was achieved with the original intended ideas and to take one of the following alternative actions: nothing,

since the problem is considered resolved; or debug[1] the description, since the result is different from what was intended;

- *Debugging the previous version, producing a new program version*. The learner can search for bugs in her/his program, getting more information so to modify the previously defined description. At this moment, the description-execution-reflection-debugging-description cycle is repeated.

From the point of view of the description-execution-reflection-debugging-description cycle, each version of the learner's program can be seen as an explicit expression of her/his reasoning, in terms of a precise and formal language. In this sense, the description here is equivalent to the knowledge representation mentioned before. However, the fact that the computer can execute the program offers new possibilities to the understanding of the student's knowledge, and of the bugs her/his program may have. By fixing these bugs a new version of the program is produced and the cycle is repeated.

The execution of the program can be interpreted as the execution of the learner's thinking as s/he solves the problem. We can see the result of it on the screen and this can facilitate understanding of what the student knows.

The learner can use the result presented to reflect on and to debug her/his ideas. Depending upon the kind of knowledge involved, the student can reach or not the problem solution. In some situations the student may not have the necessary knowledge to progress and this means breaking the cycle. In this case the teacher's intervention is necessary in order to help the student to keep the cycle going. The teacher can either challenge the student, asking questions, going over the problem solution or giving the information the student needs in order to continue working on the problem. When, what and how to intervene is a key element in helping the student to keep involved in her/his activities and requires the teacher to have some experience so as to be effective in this process (Valente, 1996). However, the greatest challenge is to help the student to maintain the cycle active.

Thus, the description-execution-reflection-debugging-description cycle does not happen just by simply putting the student in front of the computer. A professional – a learning agent – who knows how to be effective in the process of helping the student, must mediate the student-computer interaction. This professional, who could be the teacher, must understand the learner's ideas and how to intervene appropriately in the situation.

In addition, the learner is a social being and is part of a social and cultural environment built locally by colleagues, and globally by parents, friends or by the community where s/he lives. S/he can use all these social and cultural elements as sources of ideas and information or a place to find problems to be solved with the use of the computer. The interaction of the learner with the computer and the various elements that are present in the programming activity are shown in Fig. 2.

[1]Debug is a term introduced by Artificial Intelligence (Sussman, 1975), meaning the process of eliminating "bugs" that prevent computer programs from working. In this case, programmers do not ask whether the program is correct or wrong but if it has bugs and if is possible to fix it or "debug" it. Bug and debugging were adopted by Papert (1980) as a way of looking at intellectual activities as having bugs rather than errors for which to be punished.

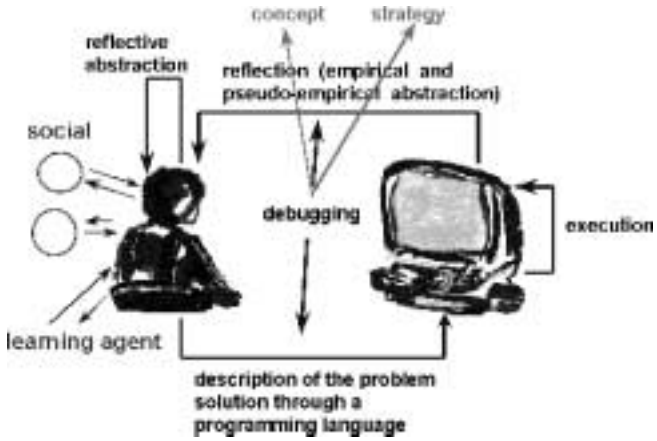

Fig. 2. Interaction learner-computer when developing programming activities.

Logo graphics presents characteristics that make the programming activity much easier, which does not occur in other Logo domains. In Logo graphics, the Turtle commands are relatively easy to incorporate into the student's existing knowledge (continuity principal). The description of the solution of spatial problems in terms of Logo graphics is not complicated, and the result of the computer execution is a picture, which facilitates the interpretation, reflection and debugging. But, programming in the list processing domain is much more difficult. First of all, the description of recursive processes is not an everyday activity. Second, the execution of recursive processes in list processing is opaque, making it difficult to follow what the computer is doing. In list processing, there is no entity like the Turtle, whose behavior directly corresponds to the commands and procedures being executed. Third, reflection is not assisted by the computer's actions. The absence of the Turtle and the kind of results that are obtained as a product of the list processing hinders the interpretation of what is happening in the procedures, and, consequently, the description of the problem solution.

Thus, it is not by chance that Logo graphics is the most well-known and widely used domain of Logo! On the other hand, this does not mean that list processing is impossible to penetrate. For example, using SlogoW debugger, developed by the NIED[2], it is possible to follow the processing of lists (inserted or removed elements, etc.), the modifications of the values of variables, the levels of the recursive calls, etc.

Even though the cycle makes it easier to understand the process through which a program is produced, this process can also happen when the learner uses other software such as a word processor or authoring systems. The difference between programming and these other uses is in how much is offered by these other software in terms of facilitating the description-execution-reflection-debugging-description cycle. The limitation is not in the possibility of representing knowledge but on the capacity of the computer to

[2]This software can be download from the, Nied's (Nucleus of Informatics Applied to Education) website: `http://nied.unicamp.br`

execute this representation. For example, in the case of word processing the interaction with the computer is given through the natural language and through the word processor commands for formatting the text (center the text, underline words, etc.). However, the word processor is only able to execute the text's format and some aspects of writing style. It still cannot execute the content and give feedback in terms of the meaning of what is written so the learner can compare it with her/his original idea. Since the computer can only present the results of the execution of formatting, the learner can only reflect in terms of her/his original ideas about the format, comparing them to the presented results.

As mentioned before, the emphasis of the analysis should not be only on the software. In any situation the teacher or learning agent can create situations for the student to be engaged in the cycle of actions. Thus, the question is how this cycle can help the student to construct new knowledge.

5. The Cycle of Actions and the Knowledge Construction Process

The idea of cycle – a continuous movement toward new comprehension – is central in many theories that propose learning as knowledge construction, product of the interaction between the learner and her/his environment. This idea is present in practically all social interactionist theories formulated by authors such as Piaget (1976), Vygotsky (1978), Wallon (1989), D'Ambrosio (1986). Piaget was the researcher who studied in depth the process of knowledge construction that happens in the relation between the subject and other subjects and/or objects, explaining this construction through the cycle *assimilation-adaptation-accommodation*. For D'Ambrosio, intellectual evolution can be explained by the cycle *reality-reflection-action-reality,* emphasizing the dialectic relation developed in the interaction between the subject (individual) with the reality (social and cultural).

On the other hand, the cycle in computer activities presents characteristics that are important in learning, understood as construction of new knowledge. First, the action done with the machine is mediated by description of how the user intends to solve the problem. This goes beyond what we knew about the interaction between the learner and the objects, as described by Piaget (Piaget, 1976). Piaget's objects did not have the capacity to execute instructions. They could be manipulated and, based upon the actions made researchers could interpret and infer the learner's thinking process and knowledge s/he uses. On the other hand, to manipulate the computer the user has to give orders to it and they are still given by means of description, using some kind of software such as programming languages, spread sheets, word processing.

Second the computer can execute orders that are given to it, although – differently from humans beings – the responses produced by the computer are accurate, without any kind of animosity or affectivity that could possibly exist between the student and the computer. The computer does not add any new information to the learner's program[3]. Thus, if there is any bug in the program's behavior this can be attributed only to the

[3]Except in the case that this program uses Artificial Intelligence features allowing the development of programs that "learn" and thus, can present a "new" behavior that is unexpected.

person who has produced it. This accurate feedback is extremely important for the learner to become aware of what s/he knows and what kind of information is necessary to debug her/his ideas.

Third, the computer's feedback is immediate. After pressing the ENTER key, the learner receives the results that are built step-by-step by the computer, enabling the confrontation of her/his original ideas with the results obtained on the screen. This comparison is the first step in the reflective process and the grasp of consciousness about what should be debugged.

The reflective action can produce several levels of abstraction, which, according to Piaget (Piaget, 1977) will provoke alterations in the student's mental schemes. The simplest abstraction is empirical, which allows the learner to extract information from the object or the actions on the object, such as the color and shape. The pseudo-empirical abstraction allows the learner to deduce some knowledge from her/his action or from the object. For example, s/he may understand that the picture obtained is a square and not a rectangle because it has four equal sides.

Now, both the empirical as well as the pseudo-empirical abstractions may allow the learner to gather one or more properties from what is being observed and this information can lead her/him to debug her/his program. However, the learner may be too dependent on the empirical results and the debugging be only in terms of small adjustments, and not as major conceptual changes.

Conceptual changes and construction of new knowledge are fruits of the reflective abstraction. This type of abstraction, according to Piaget (1995), consists of two aspects that are inseparable: one defined as *reflectioning* that consists of projecting (as a spot light) to a higher level what has been extracted from a lower one; the other, which Piaget called *reflexion*, is a mental act to reconstruct or reorganize on the higher level what has been extracted and projected from the lower one. In this sense, the information originating from empirical and pseudo-empirical abstractions can be projected to superior levels of thinking and reorganized to produce new knowledge.

These different levels of abstractions can be illustrated in the case of a student who is interested in using the Logo commands to define a program to draw a square. Suppose s/he knows that this figure has four sides. This can lead her/him to define a program P1 that produces a figure with four sides, although formed by four lines of different lengths and with different angles between them. As soon as this program is executed the learner can conclude, by means of empirical and pseudo-empirical abstractions, that this is not a square since the figure is not "nice" and does not "close". S/he can measure the length of the lines and this information, together with the knowledge about the Logo commands that produced the lines, can lead the student to figure out that the length of the lines must be the same. These abstractions, including the reflective one, can lead the learner to formalize these ideas in terms of a piece of knowledge such as "a square has four sides and they must be equal". This new knowledge is used to debug program P1, producing P2, that draws a figure of four equal sides. However, as the angles are not the same, the figure obtained is not a square yet.

At this point the cycle of actions continues and, by different abstractions the learner can conclude that the angles must be the same. P2 is debugged, producing P3. This new

program draws a figure of four igual sides and with igual angles between them. The figure obtained is not a square because the learner does not know that the value of the angle must be 90 degrees.

The 90 degree angle can be found through various attempts, until the figure obtained closes nicely. At this point the student by herself/himself or with the help of the learning agent can work with the information the student has acquired throughout this experience, formalizing it in terms of knowledge about squares. As indicated by Montangero and Maurice–Naville:

> *"...the reflexion notably enriches the knowledge extracted. The result of a reflective abstraction is a new form of knowledge or instrument of thinking. This creative act can lead to two results according to Piaget: either it creates a new schema (instrument of knowledge) by differentiation, or it leads to "objectivation" of a process of activity coordination: what was an instrument of thinking becomes an object of thinking and expands the subject's conscious area. We can see, therefore, that the process constructs either forms of thinking or structures as notions (both not being very well differentiated in Piaget's theory probably because he insisted on the active nature of knowledge)"* (Montangero and Maurice–Naville, 1998, p. 93).

The cycle of actions that is present in the process of programming the computer allows the student to solve a problem, reaching a result and, in this activity, constructing new knowledge. The explanation for how this happens was based upon Piaget's theory. From the theoretical point of view it is possible to use this theory to explain how any individual, being a child or a graduate student, can construct knowledge. What was described in the process of producing a program to draw a square may be typical of a five-year-old child. Keeping the same proportions, this same experience can happen with an adult trying to use the computer to simulate a phenomenon that s/he still does not understand. That is, the process of knowledge construction, at any level, goes through similar steps as was described in the case of the development of a program to draw a square.

Papert, emphasizes the importance of creating learning environments that are rich and impregnated with activities, concepts and ideas that individuals will be able to work with and acquire (Papert, 1980). Also, in these environments learners can be helped by more experienced people. From the practical point of view and, more precisely, from the educational point of view, it is not feasible to think that all that a person must know is constructed by her/him working alone, without any support. First, it is too costly to create environments that cover all the concepts about every domain available. Second, as an educational solution this is not efficient since the time that it would take for a person to re-invent all knowledge already constructed would be enormous. In this sense, the idea of construction, as Piaget has proposed, could be improved if prepared teachers could help students (Piaget, 1998). The teacher's role is fundamental in helping to formalize concepts that are historically agreed upon. Without the presence of the teacher it would be necessary for the student to re-create these conventions. Certainly, the teacher can play this role.

The teacher or learning agent intervention is facilitated by the existence of the computer program. The program represents the learner's idea and there is a direct correspon-

dence between each command and the way the machine behaves. These characteristics available in the programming process facilitate the analysis of the program so that the learner can find her/his mistakes (bugs) and the teacher can understand what the learner is doing and thinking. In this way, the process of finding and correcting the mistake is a unique opportunity for the learner to learn about a specific concept involved in the solution of the problem or about problem solving strategies. The learner can also relate her/his program to her/his own thinking at a metacognitive level, and the program can be used by the teacher to discuss ideas about learning to learn, once the learner, in the process of seeking new information, is exercising her/his ability to learn. The learner can raise questions about thinking-about-thinking, once s/he is able to analyze the program in terms of the effectiveness of the ideas, strategies and style of the problem solving. In this case the learner begins to think about her/his mechanism for reasoning and learning. Furthermore, the different versions of the programs produced can show the development of the learner's ideas. If we save all the different versions of the program, we can follow the process through which the learner has built the concepts and strategies involved in the program.

The cycle of actions that help to understand how a program is produced and how knowledge can be constructed in this activity, can be applied to understand how knowledge can be constructed through the use of other educational software, such as word processing, spread sheets, Internet (Valente, 1999). Also, as was discussed, the cycle has been helpful to understand the role of each action the student develops and its contribution to the process of problem solving, of knowledge construction, of learning how to learn, and of thinking. However, as a mechanism to explain what happens to the learner's mind as s/he interacts with the computer, the cycle idea is too limited. The actions can be cyclical and repetitive but each time a cycle is performed the constructions expand. Even when the learner makes a mistake and does not successfully reach the result, the learner is gathering information that can be useful to the knowledge construction process. In fact, as a cycle finishes, the learner's thinking is not exactly identical to what s/he had when it began. Thus a more adequate idea to explain the mental process of this learning is a spiral

6. The Learning Spiral in the Computer Interaction

As was mentioned before, Piaget also utilized the idea of cycle, as *assimilation-adaptation-accommodation*, to explain the process of knowledge construction. However, to describe the expanding and temporary characteristics of the equilibration that are present in the cycle, Piaget emphasized the "heightening equilibration" or "*équilibration majorante*". He mentioned that the sources of progress in learning are the disequilibration, incoherence and conflicts that arise as a learner develops an activity. Once the mental equilibrium is disturbed its tendency is to re-equilibrate, although at a higher level, with improvements.

> "*... disequilibrium represents a role of discharge, as its fecundity is measured by the possibility to overcome it – that is, to get out of it. The real source of*

> *progress must be looked at in the re-equilibration, naturally, not in the sense of returning to the initial form of equilibration, whose insufficiency is responsible for the conflict to which this temporary equilibration has reached, but to a improvement of this preceding form. However, without the disequilibration there would be no "heightening re-equilibration" (indicating thus the re-equilibration with obtained improvements)*" (Piaget, 1976, p. 19).

This constant improvement in thinking and in heightening equilibration is better represented as a spiral rather than a cycle. As mentioned by Morin, "*The spiral circuit of a whirlpool is, in fact, a circuit that closes, opening itself and thus, forming and reforming itself.*" (Morin, 1997, p. 197). The cycle suggests the idea of repetition, of periodicity, of a certain order, of closed ness, with initial and final points that coincide. In this sense, knowledge could not grow and would be repeated, in a circular movement.

Thus, the idea of spiral to explain the process of knowledge construction that grows continually is more adequate as a model of what is happening in the learner-computer interaction.

For example, the expanding spiral is present in the development of different programs P1, P2, P3. . . to produce the square, described above. Program P1 is produced based upon the learner's level of knowledge about the problem and about the computer technical resources – knowledge involved in the problem, concepts about computers and commands of the programming language, strategies to apply these concepts, etc. This knowledge must be coordinated in order for the learner to be able to propose an initial solution to the problem, in terms of a program P1.

It is important to notice that for the development of P1 it is not necessary for the learner to know everything about the problem or everything about computers. P1 is defined in terms of an initial comprehension of these concepts, made explicit in the program by means of the programming language commands. Therefore, the development of P1 means the ***description*** 1 of the learner's knowledge about the problem's solution in terms of commands that can be executed by the computer so the problem can be solved.

The ***execution*** 1 of P1 gives a result R1, obtained immediately and produced according to what the machine was ordered to do. This result R1 is used as the object of ***reflexion*** 1, leading to a debugging of P1. The ***debugging*** 1 of P1 means the production of program P2, that is ***description*** 2. This version of P2 incorporates more sophisticated levels of knowledge, fruit of the reflection done by the learner or new concepts or strategies the learner has assimilated by looking at books or talking to specialists, colleagues, etc. When program P2 is executed it produces results R2 that becomes object of reflection and so forth. However, in each action of the cycle there is an increment of knowledge. Each of these actions *description* 1, *execution* 1, *reflection* 1, *debugging* 1, *description* 2. . ., contributes to the constitution of an expanding spiral of knowledge that is constructed as the learner interacts with the computer, as shown in Fig. 3.

Although these actions are presented sequentially and as independent of each other, in the real process of programming they may happen simultaneously. This separation is done so as to comprehend the role of each one of them in the process of knowledge construction. For example, during the execution, as the results are shown, the learner can

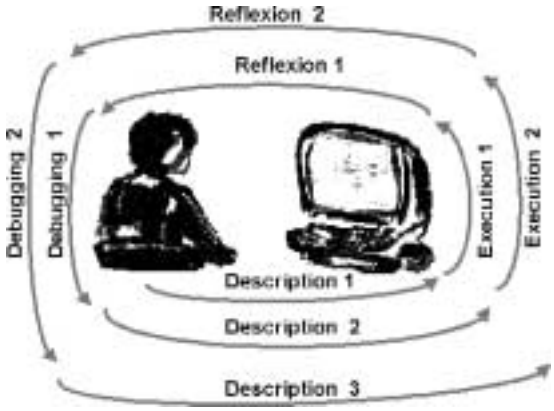

Fig. 3. Expanding learning spiral that happens in the interaction learner-computer.

be reflecting. Therefore, the best representation for this spiral could be a whirlpool, where the actions are taking place simultaneously.

The exercise of rethinking the actions that take place in the computer-learner interaction in terms of a spiral has allowed us to understand much more about them, helping to identify other important functions. For example, the fact that a particular computer program always produces the same results (characteristic literally cyclic) functions as an anchor for the learning process that is based upon conflicts. Without this anchor the learner would be confused in terms of evaluating whether her/his mental confusions or uncertainties originated from misunderstandings or a game the computer is playing with her/him by changing the program automatically. If the computer does not behave this way, the learner can be sure that the confusion is only due to some deficiency in her/his knowledge.

Another aspect to be observed is that the literature that mentions the spiral idea, such as Morin (1997), Maturana and Varela (1997; 1995) discusses also the emotional and affective side of learning. The knowledge representation and the cycle ideas emphasize the concepts and strategies the learner is using in the problem or project. This is the rational side, the cognitive aspect of the problem solution. However, the problem or project being solved also presents aesthetic aspects that cannot be ignored. They are represented in terms of commands that can be analyzed in the same way as it is done with the cognitive aspect. The aesthetics constitute the emotional side, the affective aspect that has normally been neglected in school activities. As the possibilities of combining texts, pictures, video, and animation are becoming easier and simpler to explore and manipulate in modern software, it is possible to understand how people express these sentiments through these software. To represent and to make explicit this aesthetic knowledge constitutes the first step in the process of comprehending the emotional side, what in Education has been overlooked by the cognitive and rational aspects.

Nevertheless the theories of complexity presented by Morin (1997) and of autopoiese proposed by Maturana and Varela (1995; 1997) allow the understanding that the human mind is not only based upon cognitive structures of knowledge. As mentioned by Moraes,

"*cognition – the process of knowing – is broader than the concept of thinking, of reasoning, and of evaluation because it involves the perception, the emotions and the action, everything that constitutes the process of life*" (Moraes, 2002, p. 4). The works developed through the use of computers have served to explain these actions in multiple dimensions and comprehend better how the process of thinking and learning is taking place.

7. Conclusion

One of the research agendas in the Informatics and Education area has been the understanding of how learning is taking place in the computer-learner interaction. Initially the computer was useful as a means to represent knowledge. Next, it emphasized the capacity this machine had to execute these representations, giving rise to the idea of the cycle of actions the learner uses to produce the program and the role of each action in the process of knowledge construction. This was very useful to understand the importance of each ingredient that had to be present in the creation of computer based learning environments. However, as it was possible to comprehend more about learning it was evident that the idea of cycle was limited. The cycle conveys the notion of continuous movement, one of the important aspects of learning, but does not allow the notion of expanding circularity. The model that best captures this aspect is of a spiral.

This article discussed the different functions the computer plays as a resource to help the knowledge construction process, especially as a means to represent knowledge. The cycle of actions that takes place in the computer-learner interaction was used to show the spiral characteristic of learning, a more adequate mechanism to understand the process of knowledge construction that happens when a person is using informatics to solve a problem or develop a project.

The idea of spiral is more adequate for understanding how the computer can help learning. It also allows the dissection of certain actions in order to understand them in a much broader sense. For example, up to this moment only the cognitive aspect had been emphasized in learning. The spiral means also openings to new aspects such as the aesthetic and the emotional ones that are becoming as important as the cognitive one. In fact, the information and communication technologies are creating circumstances for people to express themselves as a whole, and not only the cognitive but the affective and social as well. The resources to explore the aesthetic aspects and the possibilities of forming networks of people interacting via Internet have facilitated the exploration of these other human being dimensions, forcing us to continuously rethink our role as learner, the role of technologies in this process and our conceptions about learning, especially when done with the help of the computer. This is the movemnt of the spiral in action!

References

D'Ambrosio, U. (1986). *Da Realidade à Ação: Reflexões sobre Educação e Matemática* (English translation of the title, From Reality to Action: Reflections on Education and Mathematic). São Paulo, Summus.

Clements, D.H., and J.S. Meredith (1993). Is Programming Obsolete? *Logo Exchange*, **1** (12).
Maturana, H., and F. Varela (1995). *A Àrvore do Conhecimento* (English translation of the title, The Knowledge Tree). Campinas, SP: Editora Psy,.
Maturana, H., and F. Varela (1997). *De Máquinas a Seres Vivos*. Artes Médicas (English translation of the title, From Machines to Living Beings), Porto Alegre.
Moraes, M.C. (2002). Aprendizagem e Vida (English translation of the title, Learning and Life). Non published article.
Morin, E. (1997). *O Método – a natureza da Natureza* (English translation of the title, The Method – the Nature of Nature). Portugal, Publicações Europa–América.
Montangero, J., and D. Maurice–Naville (1998). *Piaget ou a Inteligência em Evolução*(English translation of the title, Piaget or the Intelligence in Evolution). Porto Alegre, Artmed.
Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas*. New York, Basic Books.
Papert, S. (1986). Constructionism: a new opportunity for elementary science education. *A Proposal to the National Science Foundation*. Cambridge, Massachusetts, Massachusetts Institute of Technology, Media Laboratory, Epistemology and Learning Group.
Piaget, J. (1976). *A Equilibração das Estruturas Cognitivas* (English translation of the title, The Equilibration of Cognitive Structures). Rio de Janeiro, Zahar Editores.
Piaget, J. (1977). Recherches sur L'abstraction Réfléchissante. *Études d'épistemologie génétique*. PUF, **2**, Paris (in French).
Piaget, J. (1978). *Fazer e Compreender* (English translation of the title, How to Do and to Comprehend). São Paulo , Edições Melhoramentos and Editora da Universidade de São Paulo.
Piaget, J. (1995). *Abstração Reflexionante: relações lógico-aritméticas e ordem das relações espaciais*(English translation of the title, Reflective Abstraction: Logical-Arithmetic Relations and Order of Spatial Relations). Porto Alegre, ArtMed.
Piaget, J. (1998). *Sobre Pedagogia* (English translation of the title, About Pedagogy). São Paulo, Casa do Psicólogo .
Ralston, A., and C.L. Meek (1976). *Encyclopedia of Computer Science*. First edition. New York, Petrocelli/Charter.
Sussman, G.J. (1975). *Computer Model of Skill Acquisition*. American Elsevier Publishing Company, New York.
Valente, J.A. (1994). Computers in education: shifting the pedagogical paradigm from instructionism to constructionism. *Logo Exchange*, **2** (12), 39–42.
Valente, J.A. (1996). O papel do professor no ambiente Logo (English translation of the title, The Teacher's Role in the Logo Environment). In J.A. Valente (Ed.), *O Professor no Ambiente Logo: formação e atuação* (English translation of the title, Computers in the Knowledge Society). SP: UNICAMP/NIED, Campinas, pp. 1–34.
Valente, J.A. (1999). Análise dos diferentes tipos de software usados na educação (English translation of the title, Analysis of Different Types of Software Used in Education). In J.A. Valente (Org.), *Computadores na Sociedade do Conhecimento.* SP: UNICAMP/NIED, Campinas, pp. 89–110.
Vygotsky, L.S. (1978). *Mind in Society – the Development of Higher Psychological Processes*. MA, Harvard University Press, Cambridge.
Wallon, H. (1989). *As Origens do Pensamento na Criança* (English translation of the title, Origins of Thinking in the Child). Manole, São Paulo.
Weir, S. (1987). *Cultivating Minds: A Logo Casebook*. Harper & Row, Publisher, New York.

J.A. Valente is a professor in the Department of Multimedia, of the Institute of Arts and vice-director of the Nucleus of Informatics Applied to Education (NIED) both at the State University of Campinas (UNICAMP), and collaborator in the Graduate School of Education: Curriculum at the Pontificia Catholic University in São Paulo (PUC-SP), Brazil. He received his PhD in informatics applied to education from the Massachusetts Institute of Technology (MIT), in 1983, developing a computer-based learning environment for physically handicapped children. Currently, his main research focus includes the creation of information and communication technology (ICT) based learning environments and training methodology to be used in factories or in school settings, involving face-to-face or distance educational approaches, as well as the study of the potential of ICT as learning tools.

Logo įtaka informatikos reikšmės švietime supratimui ir ryšiui su mokymosi procesu

José Armando VALENTE

Informatikos taikymas švietime daug kuo pasitarnavo plėtojant mokymo ir mokymosi procesų sampratą. Pirmiausia, šis taikymas įgalino atskirti instrukcionizmą, kuris suvokiamas kaip informacijos perdavimas, ir konstruktyvizmą, kaip žinių formavimo procesą, pasireiškiantį per tai, kad besimokantysis išmoksta naudodamasis kompiuteriu pagaminti prasmingą produktą. Antra, programavimas, ypač naudojant Logo kalbą, daug padėjo, kad būtų suprasta, kaip formuojasi besimokančiojo žinios jam dirbant su kompiuteriu.

Straipsnyje parodoma, kad bėgant metams toks supratimas keitėsi. Pradžioje dėmesys būdavo telkiamas ties žinių reprezentavimo aspektu. Tuo tarpu, naudojantis Logo, vienu kartu mokomasi ir komandų, ir to, kaip jas panaudoti kuriant programas. Be to, sukurtoji programa gali būti traktuojama kaip sąlyčio taškas tarp konkrečių ir abstrakčių sąvokų, kokia pavyzdžiui, funkcijos sąvoka matematikoje. Kvadrato braižymo programą galima traktuoti kaip matematinę funkciją, kuri atvaizduoja visus sveikuosius skaičius atitinkamų dydžių kvadratais, tuo būdu palengvindama abstrakčių sąvokų įsisavinimą. Vėliau į programų kūrimo procesą imta žiūrėti kaip į tam tikrą veiksmų ciklą, susidedantį iš uždavinio sprendimo aprašymo programavimo kalba, šio aprašymo realizavimo kompiuteriu, kompiuterio atliktų veiksmų apmąstymo, ankstesnės programos versijos patobulinimo, sukuriant naują jos versiją. Straipsnyje svarstoma šio ciklo pateikiamų charakteristikų svarba žinių formavimo supratimui.

Geriausiai žinių formavimo procesą programavime reprezentuojančiu modeliu straipsnyje pripažįstamas spiralinis modelis. Ciklo modelis remiasi pakartojimo, uždaro rato, – kurių pradinis ir galutinis taškai sutampa, – idėja. Pastaruoju atveju žinios negalėtų augti, o būtų tik kartojamos. Tuo tarpu spiralės idėja, aiškinant nuolatos augantį žinių formavimo procesą, aiškiau apibūdina tai, kas vyksta besimokančiajam dirbant kompiuteriu. Straipsnyje aptariami ciklo ir spiralės modeliai padeda aiškintis tai, kaip formuojasi žinios naudojantis kompiuteriu, pagalbon čia pasitelkiamos Piaget'o bei Paperto idėjos, ypač jų mintys apie programų apmąstymą ir tobulinimą.