'Learning activity' as the basic structural element for the design of web-based content: A case study

Maria Kordaki

Dept of Computer Engineering and Informatics
Patras University, 26500, Rion Patras, Greece
e-mail: kordaki@cti.gr

Abstract: This paper presents the design and the pilot evaluation study of learning activity-based, multi-media, multi-representational and multi-layered content for the learning of basic concepts of programming and C. This content is integrated into a web-based learning environment named L.E.C.G.O. (Zikouli, Kordaki and Houstis, 2003). This design was based on modern social and constructivist theories of knowledge construction. Thus, the content was organized in a holistic way, using the learning activity as its basic structural element. In particular, the whole information was organized in five layers, presenting: a) complex programming examples (1st layer), b) simple programming examples (2nd layer), c) broad information about programming in C (3rd layer), d) fundamentals in programming (4th layer) and e) broad information at various locations on the WWW (5th layer). The data emerging from a field pilot evaluation study indicated better results in comparison to book-like content used in other learning settings such as paper and pencil and Turbo C.

Introduction

Different learning theories affect, in different ways, the designers of content in both traditional text-book-based learning settings and dynamic web-based learning environments. In fact, traditional leaning theories emphasize impressive presentation of informational material for each specific learning subject, which is usually split into parts, chapters and sections, starting with easier introductory elements and gradually progressing to more difficult advanced topics (Skinner, 1968). According to these traditional perspectives, the content does not emphasize the presentation of the essential concepts of the learning subject but rather presents all concepts in an imbalanced way. This type of content is usually structured in such a way as to begin by introducing learners to the theory related to the subject matter, subsequently progressing to the learning activities assigned for the consolidation of the aforementioned theory. These activities usually take the form of 'drill and practice'; that is, they focus more on the recipes needed to deal with certain trivial problems than on the basic concepts constituting the learning subject and on activities emphasizing problem solving. As a result, the learning subject is mainly viewed as a sum of formulae; to some extent, this limits its conceptual nature. These traditional learning activities are usually boring and meaningless for learners, being mostly outside their sphere of interest. Thus, learners have to memorise the formulae and the entire learning content; as a result, the whole learning process is reduced to a meaningless activity.

In contrast to traditional learning theories, modern social and constructivist theories of learning emphasize a holistic approach to the organization of learning content (Vygotsky, 1978; von Glasersfeld, 1987). According to these theories, holistic, real life learning activities are appropriate as basic structural elements of the content presentation. Constructivist design of learning activities emphasizes the fundamental concepts of the learning subject in question and not its details (von Glasersfeld, 1987; Vygotsky, 1978; Nardi, 1996). Consequently, emphasis is placed on the necessary activities to be performed by students to effectively learn the subject matter. For the design of such activities, analysis of the learning subject in terms of fundamental and timeless concepts is necessary (Kordaki & Potari, 2002). Moreover, holistic learning activities can help learners to acquire a global view of the learning subject in focus. Problem-solving activities that put learners in an investigative mode can encourage them to construct their knowledge actively and acquire some essential problem-solving skills (Nardi, 1996; Fishman, 2000). In addition, it is essential that student difficulties in the understanding of the concepts in focus be taken into account. Furthermore, multiple-solution activities that can help learners express their inter-individual and intra-individual variety is of essential and central interest (Kordaki and Balomenou, in press). Finally, activities stemming from the learners' worlds can motivate them to become involved actively in their own learning.

Hypermedia and multimedia can also play a crucial role in the holistic organization of learning content. In particular, the selected learning activities can be hyperlinked with their multimedia applications, such as animations

and simulations, providing learners with opportunities to study their evolution. In addition, these activities can be hyperlinked with text-based information for an in-depth understanding of their surrounding theoretical context. This theoretical context can be structured in hyperlinked layers where information is presented in different forms, such as: a) brief description b) detailed description c) wide-ranging and various resources, such as e-books, URLs, e-journals etc.

Holistic learning approaches are of great interest in the learning of concepts in Computer Science and especially for the learning of programming (ACM, 1991; Hadjerrouit, 1998; Ellis, 1998; Kordaki, 2001). Despite the fact that, traditional learning theories emphasize the learning of a programming language through its syntactical rules, modern learning theories emphasize the holistic approach of problem solving using algorithmic logic (Soloway & Spohrer, 1989; Komis, 2001; Grigoriadou, Gogoulou, Gouli, 2002).

This paper presents a holistic, activity-based, multi-media, multi-representational and multi-layered design of web-based content for the learning of programming and C. This content has been designed to assist learners to become active and constructive in the learning process while interacting in a computer learning environment called L.E.C.G.O. ('A Learning Environment for programming and C using Geometrical Objects', Zikouli, Kordaki & Houstis, 2003). L.E.C.G.O. is an interactive multi-representational environment designed to provide secondary level education students with opportunities to learn programming and C. L.E.C.G.O. provides learners with opportunities to construct algorithmic solutions to problems in a variety of representation systems. Learners can exploit the multiple media and the various forms of the information integrated into L.E.C.G.O. to succeed in the tasks at hand. This content has been tested in the field using real students with positive results; such design of content presentation has not yet been reported.

In the next section of the paper, a brief description of LE.C.G.O. as a learning environment is followed by the description of the design of the proposed content with reference to specific examples used. A pilot evaluation of this content follows and, subsequently, its importance is discussed and conclusions drawn.

L.E.C.G.O. as a learning environment

The design of L.E.C.G.O. was the result of a synthesis of three models (Zikouli, Kordaki & Houstis, 2003): a) the learning model, based on modern social and constructivist theories of learning acknowledging the active, subjective and constructive character of knowledge and the crucial role of tools and of Multiple Representation Systems (MRS) in knowledge construction (von Glasersfeld, 1987; Noss & Hoyles, 1996). In particular, it is acknowledged that computer learning environments providing a variety of RS of different cognitive transparency could encourage students to select from among them the most appropriate tools to express their knowledge. These different RS can provide students with opportunities to express their inter-individual and intra-individual variety (Dyfour-Janvier, Bednarz & Belanger, 1987; Kordaki, 2003). It is noteworthy that most learner difficulties are found in the gap between their intuitive knowledge and the knowledge they need to express themselves in the RS proposed for use (Janvier, 1987). For example, prepositional, symbolic and abstract RS prevent some learners (usually beginners) from expressing their knowledge, the same systems being intended for use by advanced learners. Contrariwise, metaphors of everyday life and visual RS are more suitable for beginners; b) the subject matter model, based on the literature on basic aspects and structures of programming and C and c) the learner model, based on the literature on how students learn essential aspects of programming. L.E.C.G.O. was designed to be a possible learning environment for twelfth grade students (18 years old) and for first-year University students. The programming language C was selected as a learning subject as this is a modern language with great capabilities which could also become a solid background for the learning of object-oriented programming.

L.E.C.G.O. provides students with opportunities to: a) express their solution strategies in MRS starting from 'anthropocentric', non-necessary programming solutions and gradually moving to more computer-oriented programming solutions, b) acquire hands-on experience in providing graphic solutions to the problems at hand, using tools that support the direct manipulation of computational objects on a computer screen, c) deal with graded difficulty learning activities within the context of drawing using basic geometrical objects (Zikouli and Kordaki, 2004). Drawing was selected as a context for the learning activities as it would motivate learners to become actively and passionately involved in their own learning. In addition, drawing using geometrical objects was selected to give students the chance to learn about the graphic functions in C. The aforesaid activities can be solved without the extra cognitive load that might stem from the demand to perform complex geometrical constructions. d) offer assistance with their problem-solving strategies. This help is provided in three modes: i) as ready specific expressions that could be used to describe, in natural language, a specific algorithmic solution to the task at hand, ii) as ready structures and functions in pseudocode, provided in the form of buttons and iii) as ready structures and functions in

C, also provided in the form of buttons. The design of L.E.C.G.O. could be modified for the learning of any programming language.

The general architecture of L.E.C.G.O. is diagrammatically presented in Figure 1 and also demonstrated on the L.E.C.G.O. home page. The aforementioned architecture is divided into two main parts: a) that presenting the appropriate content for learning fundamentals in programming and C. The design of this part is presented in this paper, and b) that dedicated to the learning activities that have to be performed by students in order to learn fundamentals in programming and C. This part includes tools for algorithmic solutions of problems in multiple Representation Systems (RS) such as: i) graphical RS providing opportunities to solve problems graphically using Cabri-Geometry II tools (Laborde & Strasser, 1990). Cabri-Geometry II is well-known educational software designed to perform geometrical constructions within the context of Euclidean Geometry; ii) text-based RS providing possibilities for translating the graphical solution given by a student in Cabri into natural language; iii) imperative RS, providing specific expressions in the imperative which could be used for translation of the solution at hand into the imperative; iv) pseudo-code RS, to translate the solutions expressed in the previously mentioned RS into pseudocode; v) C language-based RS and vi) the graphic output of the written programs.

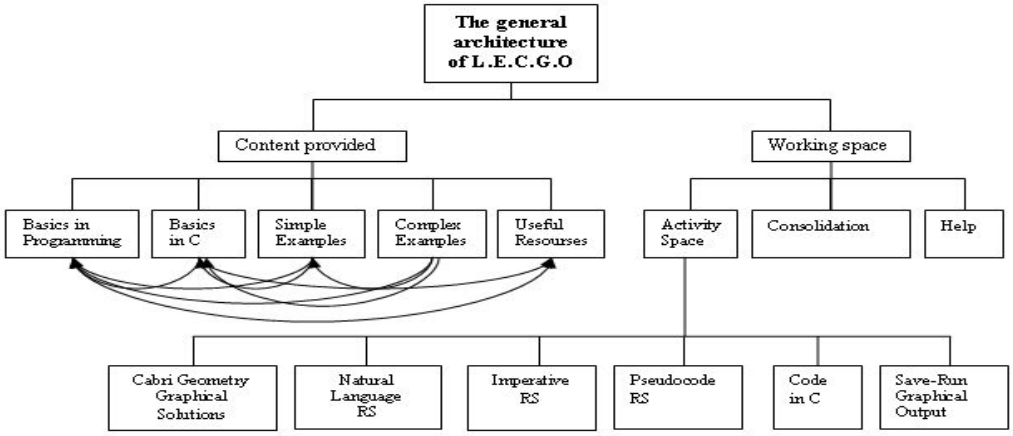

Figure 1. The general architecture of L.E.C.G.O.

The design of the proposed learning content

The design of the proposed learning content for the learning of programming and C was based on modern social and constructivist theories of learning. ? he focus of this content was on the learning of basic algorithmic structures in C. This was chosen because a considerable amount of research has reported serious student difficulties in understanding the algorithmic logic in problem solving (Soloway & Spohrer, 1989; Komis, 2001; Grigoriadou, Gogoulou, Gouli, 2002). In fact, it seemed that students persist in referring to their previous knowledge of problem solving, emphasizing other methodologies from learning other subjects, such as mathematics, physics, etc. In these subjects, the nature of problems is different than in programming; consequently, there is no reason to use algorithmic logic as the problem-solving methodology. In programming, students face difficulties in understanding the whole dynamic context of an algorithmic methodology that focuses on the formation of an appropriate process to obtain the preferred results. Students also encounter difficulties in their transition from intuitive or non-intuitive algorithmic solutions to algorithmic ones.

Taking into account all of the above, we have structured the proposed content in a hierarchical order of five layers, including: a) complex examples (1st layer), b) simple examples (2nd layer) c) broad information about programming in C (3rd layer), d) fundamentals in programming (4th layer) and e) broad information at various locations on the WWW (5th layer). All layers except for the last one are hyperlinked. All examples and topics integrated into this content can also be accessed through a specially-designed index. Complex examples provide holistic knowledge about algorithmic logic while simple examples are usually focused on how a specific algorithmic structure works. All web pages included in all layers have the following common characteristics:

- Each page displays the index mentioned above for easy transition to the preferred example or topic

- Each page provides possibilities to return to the home page
- The presentation of the solution of a problem starts from its graphic representation and moves gradually to its representation in the programming language C, through the various RS mentioned in the previous section. This was chosen to help students move from intuitive graphic solutions to solutions using pseudo-code and, ultimately, to solutions using the commands of a programming language such as C.

In the next section of this paper, specific examples of the proposed content in the above layers are presented.

1st layer: Presentation of complex examples for the learning of programming and C. In these pages (see Figure 2) a student is given the possibility to study examples for programming in C with holistic characteristics. Some of those examples are presented in the following section:

Example 1: *Write a program for drawing a small boat on your computer screen (as you prefer).* This example aims to familiarize the student with graphic co-ordinates and graphic functions in C as well as assist in their primary approach to forming a program using the basic serial algorithmic structure in C.

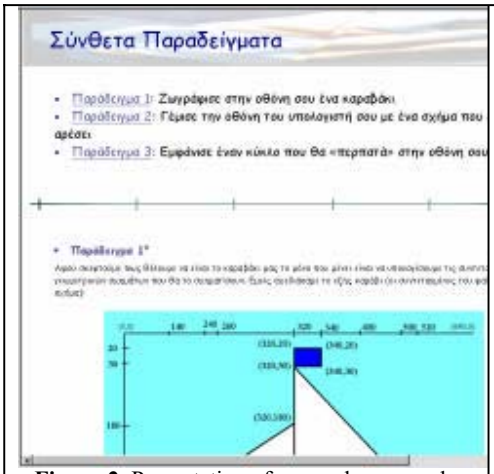

Figure 2. Presentation of a complex example

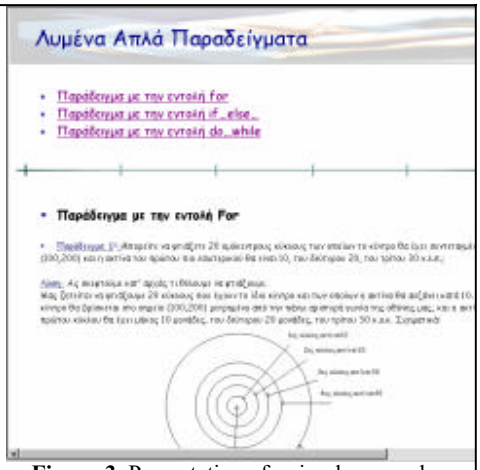

Figure 3. Presentation of a simple example

Example 2: *Select/draw a shape/pattern and write a program that fills your computer screen.* By studying this example, students have the chance to gain some knowledge about data Input/Output and the use of nested 'for…next ' loops. However, one could face this problem by using nested 'do…while'. This example was selected to provide students with opportunities to study multiple-solution based problems that can help them to enhance their knowledge of the learning concepts in focus (Kordaki and Balomenou, in press). Students also can explore the slutions provided for this task and subsequentlyt invent their own solutions. For better understanding of this example, this is hyperlinked to other simple examples, namely the 2nd and 3rd simple examples presented in the next section (2nd layer).

Example 3: *Write a program that illustrates a circle moving on your computer screen.* Here, the program illustrates a circle on a point of the computer screen; a few seconds later, this circle is erased and then illustrated at another point on the screen etc. By studying this example, students have the opportunity to learn about graphic functions in C, and about the delay function of header file dos.h, as well as about the iteration structure 'do…while' which controls , if the circle is drawn within the right limit of the computer screen.

2nd layer: Presentation of simple examples for the learning of programming and C. In the pages included in this layer, students can study simple examples. These examples are presented to assist students in their understanding of fundamentals in programming and C as well as in their understanding of complex examples presented in the 1st layer mentioned in the previous section. Some simple examples included in 2nd layer of the proposed content are presented below:

Example 1: *'Ask student to select between a circle and a rectangle. If the student selects a circle, draw a red circle, then a green rectangle. The point on the computer screen where the selected shape is to be drawn is chosen by the student.'* The aim of this example is to familiarize the students with the 'conditional statement: if…then…else' and about data Input/Output.

Example 2: *'Write a program that draws 20 concentric circles. The centres of these circles are to be drawn on the point with co-ordinates (100, 200), the radius of the first circle being 10 pixels, of the second circle 20 pixels, etc.'* By studying this example, students have the opportunity to gain some knowledge of the 'for…next' statement (Figure 3). The large number of concentric circles - 20 circles - was selected to provide students with the opportunity to understand the difference between two programs: one constructed using a sequential structure and another using an iteration structure. In particular, it should be emphasized that when using both of these programs, although the computer does the same task, the programmer's work is different; it is more sophisticated, more convenient, easier to understand and easier to correct the whole procedure.
Example 3: *'Fill the first line of your screen with a shape of your preference'.* The aim of this example is to provide students with the chance to learn about the appropriate use of the iteration structure 'do…while' when the number of iterations is unknown beforehand.

3rd layer: Presentation of basic aspects and topics for the learning of programming in C. Through the pages included in this layer, students are provided with opportunities to access detailed information related to basic concepts and functions in C, namely: a) description of the language C, b) basic steps in forming an executable program in C, c) the structure of a program in C, d) variables in programming and in C, e) control statements in C, f) data Input/Output in programming in C, g) essential graphic characteristics of the programming environment using C, and h) graphic functions in C. These web-pages feature an index that assists learners to access the specific information they prefer easily. The sub-set of the language C selected to be presented through these web-pages has been evaluated as being appropriate to support students in solving essential and fundamental problems in programming using C so that they may present drawings using basic algorithmic structures and simple geometrical objects.

4th layer: Presentation of fundamentals for the learning of programming. This layer was created to assist students to re-think basic aspects of programming and computer science and includes the necessary background for the learning of fundamentals in C. The information provided is organized in alphabetical order and presented through an index hyperlinked with the appropriate data.

5th layer: Presentation of broad information for the learning of programming and C at various locations on the WWW. A number of online resources are provided to assist learners to develop a broad view of programming and C. These resources can be divided into the following categories: a) e-books, b) e-manuals, and c) URLs.

The pilot evaluation study of the content integrated into L.E.C.G.O.

The proposed content was evaluated using real students within the context of a pilot evaluation of L.E.C.G.O. In particular, 9 twelfth grade students participated in this evaluation experiment. These students at a technical school located in Patras, Greece, were attending specific classes dedicated to providing them with special knowledge of Computer Science suitable for their employment in related jobs. Students were set four learning tasks, in three learning settings, namely: a) the paper and pencil classroom setting, b) the typical environment of Turbo C and c) the L.E.C.G.O. environment. Students were provided with paper based information to confront the tasks at hand, in the paper and pencil environment and in Turbo C. Students used the proposed content to deal with the set tasks in the context of L.E.C.G.O. The aforementioned three sets of four tasks were holistic and open. These tasks were also within the context of drawing, using simple geometrical objects. The learning tasks across the three learning settings were similar but not identical. In the following section, the tasks set to be performed in the context of L.E.C.G.O. are presented.

- *Task 1.* Write a program that, when executed, will display on your computer screen a drawing you can hang on the wall of your room. Use geometrical shapes of your own choice.
- *Task 2*. Write a program that, when executed, will display on your computer screen a train with as many carriages as you like.
- *Task 3.* Write a program that, when executed, will display on your computer screen a grid with its diagonal cells painted black.
- *Task 4.* Write a program that, when executed, will display a shape of your choice moving on your computer screen and leaving a specific trail behind it. Create a program that will enable this trail to be customized.

In terms of methodology, this research is a qualitative study (Cohen and Manion, 1989). The data resources included the field notes of the researcher recording student interactions with the content provided in the three learning settings of this experiment, the students' work sheets and the computer programs that students constructed during this pilot study. The researcher participated in this study as an observer with minimum intervention so as not to affect student actions. A more detailed description of this evaluation experiment is presented in Zikouli and Kordaki (2004). In this paper, the focus is on the evaluation of the learning content provided.

The data analysis revealed that all students visited all the kinds of examples integrated into the content of L.E.C.G.O. All students visited the animations provided more than once in order to attempt the set tasks successfully. These animations were used to explain the solution of the specific examples by displaying the execution of the corresponding programs step by step. Students usually visited the proposed examples when they faced difficulties in understanding: a) how a statement and an algorithmic structure works, b) how to combine basic algorithmic structures to deal with a complex problem, c) the kind of iteration structures needed and the number of iterations required, and d) the nature of the variables needed and their initialization.

Fewer students visited the pages that presented basic concepts of programming in C (3^{rd} layer) and few students studied basic concepts of programming (4^{th} layer). As the time of this evaluation study was limited, none of the students visited the various locations on the WWW suggested in the 5^{th} layer of the proposed context. Students studied the proposed content in the manner described above in order to attempt the set tasks successfully. Table 1 displays the number of students who successfully managed the set tasks in each of the three settings of this learning experiment.

Learning setting	**Number of students who successfully managed the set tasks**			
	1^{st} task	2^{nd} task	3^{rd} task	4^{th} task
Paper and pencil (p-p)	1	2	0	1
Turbo C	5	2	0	4
L.E.C.G.O.	8	6	3	4

Table 1. Number of students who successfully managed the set tasks in p-p, Turbo C and L.E.C.G.O.

As is shown in Table 1, more students successfully attempted the set tasks in the context of LE.C.G.O. than those who succeeded in the paper and pencil and Turbo C environments. This is due to the difference between the content integrated into L.E.C.G.O. in combination with the other features it provides and the book-like content and features provided by the paper and pencil and Turbo C environments. Students were also inspired - by the content integrated into L.E.C.G.O. - to use more types of graphic functions than those they used in the paper and pencil and Turbo C environments. In fact, the graphic functions used in L.E.C.G.O. were three times more than those used in paper and pencil and two times more than those used in Turbo C. Finally, students asked more complex questions when interacting within LE.C.G.O. than those they asked while acting in the paper and pencil or Turbo C environments.

Discussion

An alternative approach to the design of content presentation, including information materials to be used in web-based learning contexts, has been presented in this paper. This design has been inspired by modern social and constructivist theories of learning emphasizing the role of holistic, real life, multiple-solution-based problem solving, based on the influence of activities from the students' own world on their learning. Thus, the basic structural element of the proposed content was learning itself. In the design of the proposed content, the features of multimedia and hypermedia were also taken into account. In this way, the design of the content in question emphasized the learning activity presented by using multimedia and multiple representation systems hyperlinked in multiple layers. The proposed content was designed for the learning of fundamentals of programming and C by twelfth grade students and first year university students. This content was integrated into a web-based learning environment for the learning of programming and C named L.E.C.G.O. As students face difficulties in understanding how a computer works and move from their own intuitive solutions to more sophisticated solutions, L.E.C.G.O. provides learners with specific tools to represent solutions in different representation systems. Consequently, students can start by forming 'anthropocentric' solutions, by describing how they could perform the task at hand. Then, students could gradually move to computer-oriented solutions using pseudo-code and the programming language C. To do this, students need the assistance of a number of tools but also from the way the examples integrated in the content provided are presented. In fact, the presentation of these examples is performed in multiple representation systems.

The whole learning information is organized in five layers, including: a) complex examples (1st layer), b) simple examples (2nd layer) c) broad information about programming in C (3rd layer), d) fundamentals in programming (4th layer) and e) broad information at various locations on the WWW (5th layer). All layers bar the last are hyperlinked. The proposed content was evaluated in the field through a pilot evaluation of L.E.C.G.O. In particular, L.E.C.G.O. was evaluated through a comparative qualitative study using real students in three learning settings where students faced similar tasks: a) the paper and pencil learning environment, where the information given in paper and pencil was organized in a traditional manner emphasizing presentation of definitions and theories before implementation of this theory in specific examples b) the Turbo C environment, providing the same information materials and c) the L.E.C.G.O. environment. The analysis of the data collected during this experiment showed that all students were interested in the examples provided and visited them more than once to deal with the set tasks successfully. Students also seemed to prefer to study the information presented in the form of examples (complex and simple examples) while they seemed not to be attracted by the presentation of information in terms of definitions and descriptions (included in the 3rd and 4th layers). It is worth noting that, while all students visited the examples provided, fewer students visited the other kinds of information provided. Students also seemed to understand the information better when the execution of the program is presented dynamically step by step than when it is represented in a static way, as in the paper and pencil environment. Comparing the effects of the proposed content with book-like organized content, the data analysis revealed very different results: more students successfully performed in L.E.C.G.O. than in the paper and pencil and Turbo C environments. This was also due to the fact that L.E.C.G.O. generally provided different capabilities from those of the static paper and pencil environment and the typical compiler of Turbo C, where students have no opportunities to express their intuitive knowledge but are forced to use the commands of the programming language C directly.

Conclusions and future work

This paper presented the design and the pilot evaluation phase of a multimedia, multi-representational and multi-layered content, structured using the learning activity as its basic structural element. This content was developed for the learning of fundamentals in programming and C and integrated into a multi-representational web-based environment named L.E.C.G.O. The design of this content was inspired by modern social and constructivist theories of learning. In fact, five hyperlinked layers of multimedia information were used. Holistic, complex examples for the learning of programming and C were presented in the 1st layer; simple examples were included in the 2nd layer; basic aspects and topics for the learning of programming in C were integrated into the 3rd layer; the 4th layer was comprised of fundamentals for the learning of programming and, in the 5th layer, broad information for the learning of programming and C located at various web-sites was presented. The data analysis from the L.E.C.G.O. pilot field evaluation study revealed that students mostly used the first three layers to accomplish the set tasks successfully. Students visited all types of solved examples presented, more than once and with interest. Few students visited general information about programming and none of the students accessed the suggested material located at various websites. Students extracted information from the content integrated into L.E.C.G.O. and managed the set tasks. More students succeeded in managing these tasks in L.E.C.G.O. than in the paper and pencil or Turbo C environments, where they faced similar tasks but used book-like information. In addition, the number of graphic functions used by these students in L.E.C.G.O. was three times that used in the paper and pencil environment and twice that used in Turbo C. Finally, as the data emerging from this pilot evaluation study are very promising, the content of L.E.C.G.O. can be enriched with more examples in the near future.

References

ACM (1991). ACM Curricula Recommendations, Volume 1: Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force. http://www.acm.org/education/ curr91/ homepage.html.

Cohen, L. & Manion, L. (1989). *Research Methods in Education*. London: Routledge.

Dyfour-Janvier, B., Bednarz, N.,& Belanger, M. (1987). Pedagogical considerations concerning the problem of representation. In C. Janvier (Eds), *Problems of representation in teaching and learning of mathematics* (pp. 109-122). London: Lawrence erlbaum associates.

Ellis, A. (1998). Development and Use of Multimedia and Internet Resources for a Problem Based Environment. Proceedings of

the 3rd Conference on Integrating Technology into Computer Science Education and on 6th Annual Conference on the Teaching of Computing, (269) Ireland.

Fishman, J.B. (2000). How Activity Fosters CMC Tool Use in Classrooms: Reinventing Innovations in Local Contexts. *Journal for Interactive Learning Research*, 3(1), 3-28.

Grigoriadou, ? ., Gogoulou, ?. & Gouli, ?. (2002). Alternative approaches in teaching introductory aspects of programming: Teaching proposals. *3rd Pan-Hellenic Conference 'ICT in Education'(with International Participation). Rhodes, Greece, September, 2002, pp*. 239-248.

Hadjerrouit, S. (1998). A Constructivist Framework for Integrating the Java Paradigm into the Undergraduate Curriculum. *Proceedings of the 3rd on Integrating Technology into Computer Science Education and on 6th Annual Conference on the Teaching of Computing*, (105-107). Ireland.

Janvier, C. (1987). Representation and understanding: The notion of function as an example. In C. Janvier (Eds), *Problems of representation in teaching and learning of mathematics* (pp. 67-72). London: Lawrence erlbaum associates.

Komis V. (2001). A study of basic programming concepts within a constructivist teaching approach. *Themes in Education*, 2 (2-3),243-270.

Kordaki, M. (2001). Special characteristics of Computer Science; effects on Teaching and Learning; Views of Teachers. 8th Panellenic Conference of Greek Computer Society, Nicosia, Cyprus.

Kordaki, M., & Potari, D. (2002). The effect of area measurement tools on children's strategies : the role of a computer microworld. *International Journal of Computers in Mathematical Learning*, 7(1), 1-36.

Kordaki, M. (2003). The effect of tools of a computer microworld on students' strategies regarding the concept of conservation of area. *Educational Studies in Mathematics,* 52, 177-209.

Kordaki, M. & Balomenou, A. (2006, in press). Challenging students to view the concept of area in triangles in a broader context: exploiting the tools of Cabri II. *?nternational J?urnal of Computers for Mathematical Learning*.

Laborde, J-M., and Strasser, R. (1990). Cabri-Geometre: A microworld of geometry for guided discovery learning. *ZDM*, 5, 171-177.

Nardi, B.A. (1996). Studying context: A comparison of activity theory, situated action models, and distributed cognition. In B.A. Nardi (Ed.), *Context and consciousness: Activity theory and human-computer interaction*, Cambridge, MA: MIT Press.

Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning Cultures and Computers*. Dordrecht : Kluwer Academic Publishers.

Skinner, B. F. (1968). *The Technology of Teaching*, New York : Appleton, 1968.

Soloway E. Spohrer J.C. (1989). *Studying the novice programmer*. Hillside, N.J. Erlbaum.

von Glasersfeld, E. (1987). Learning as a constructive activity. In C. Janvier (Eds), *Problems of representation in teaching and learning of mathematics* (pp. 3-18). London: Lawrence Erlbaum associates.

Vygotsky, L. (1978). *Mind in Society.* Cambridge: Harvard University Press.

Zikouli, K., Kordaki, M. & Houstis, E. (2003). A Multi-representational Environment for Learning Programming and C. *3rd IEEE International Conference on Advanced Learning Technologies*, (pp. 459), July, 9-11, Athens, Greece, 2003.

Zikouli, ? . and ? ordaki, ? . (2004). Forming an evaluation context for the learning of basic concepts of programming and C through the use of educational software. *4rth Pan-Hellenic Conference 'ICT in Education'(with International Participation).. Athens, Greece, September, 2005, pp*. 598-606.