Pedagogical Considerations in Courseware Development: Supporting School "Comeback"

Thanos Hatziapostolou[1], Iraklis Paraskakis[2]

[1]City College, 13 Tsimiski Str, Thessaloniki, Greece, a.hatziapostolou@city.academic.gr

[2]SEERC, 17 Mitropoleos Str, Thessaloniki, Greece, iparaskakis@seerc.info

Distance learning has reached a sufficient level of maturity that it has made it acceptable. Online teaching environments, multimedia educational systems and numerous forms of technology allow distance learners to engage in the learning process. However, adult students who are not very familiar with technology and have been away from school for many years may still experience difficulties especially during the beginning of their studies. This paper describes the design and development of courseware for an introductory course for a distance learning undergraduate computer science program. It discusses the pedagogical strategies and design decisions for providing sufficient and efficient support as well as instructional activities that can engage and motivate the novice learner. It proceeds by presenting the individual components of the courseware and how the design strategies were incorporated and finally it presents the architecture and technologies that were used for the development of the learning environment.

Keywords
Adult learning, distance learning, lifelong learning, courseware development

1. Introduction

In today's knowledge-based economy lifelong learning and continuous update of knowledge and skills is instrumental. One of the most established methods supporting lifelong learning is distance learning. Despite the advances in our understanding of distance learning as well as advances in ICT supporting and facilitating this mode of instruction, certain challenges, especially at learner's start-up stage, still pertain. A recent study [1] that has been conducted at the Hellenic Open University (HOU) concerning student dropout rates and dropout causes of the distance learning undergraduate Computer Science programme, identified the following as primary challenges facing adult distance learning start-ups:

1. lack of time devoted to their studies
2. engaging back in the learning process
3. basic skills in using ICT

HOU in its attempt to assist the students' learning process decided to introduce a self-learning foundation course called 'Introduction to Computer Science' (PLH0). PLH0 comprises of numerous different components that deal with topics that are required by the HOU distance learning undergraduate CS programme. These topics range from simple use of ICT tools to essential problem solving skills, as well as, some basic mathematics. HOU also specified that the courseware material of the PLH0 foundation course had to adhere to the following requirements:

- the courseware should execute from a CD-Rom
- simple technologies should be used for the development of the educational material
- interactive elements should be used whenever appropriate
- self-assessment mechanisms should allow students evaluate their understanding
- practical exercises should be provided in order for students to develop skills
- the interface should be easy to learn and use

Within this context the courseware presented in this paper deals with the component of the PLH0 course that discusses the issues of problem-solving, algorithmic thinking and principles of structured programming. In particular, this paper discusses how challenges 2 and 3 can be tackled by presenting the design principles, implementation issues and the pedagogy in developing a courseware supporting entry level students enrolled for the HOU Computer Science programme.

2. Pedagogical Foundations

PLH0 is a resource-based, self-paced course without the involvement and support of an instructor. As a result, the courseware has to provide a rich and creative learner-centred environment that fosters effective learning by providing the necessary support to the learners and keeping them engaged and interested. At the same time though, the learning environment has to be user-friendly enough to keep the learner from becoming overwhelmed and frustrated in a way that interferes with learning. Since no assumptions can be made about students' knowledge, skills, organizational abilities and commitment, a scaffolding approach should be employed in order to assist learners in interacting with the material at hand, and thus, facilitating its assimilation and accommodation.

Scaffolding is an instructional strategy that dates back to the mid 1970s. The term was initially used by Wood, Bruner and Ross [2], as a metaphor to describe the kind of teaching that recognizes that learning always proceeds from the known to the new and builds on this connection by providing a scaffold to effectively support the construction of knowledge [3]. Over the years however, the term scaffolding has been widely used to describe effective learning support [3]. McLoughlin and Marshall in [5] define scaffolding as "a form of assistance provided to a learner by a more capable teacher or peer that helps the learners perform a task that would normally not be possible to accomplish by working independently". In a distance learning setting scaffolding describes the learning environment's design features and support services that the learner needs and several studies [4, 5, 6, 7, 8] have been conducted for the identification and categorization of distance learning instructional scaffolds.

Identifying the scaffolding strategies that would address the learner support needs was the first step for the development of the PLH0 courseware. The diversity of the students' background knowledge and skills of the HOU CS programme in relation with the requirements set by the HOU led to the following scaffolding strategies:

- provide content and resources in a variety of ways
- support comprehension through guidance and assistance
- provide and balance learner control and autonomy
- support learners in using available tools and resources
- minimize cognitive overload

The next step was to determine how these scaffold strategies will best be implemented in software and how they can be integrated in the learning environment.

3. PLH0: Design and Development Issues

Constructing the PLH0 courseware required designing and developing a range of components. These components should be able to deliver instruction, facilitate interaction, enhance the quality of learning, and most of all support the learner. This section presents the individual components that are included in the PLH0 courseware and describes and how they integrate and support the identified scaffolding strategies.

3.1 Interactive Learning Activities

Interactivity is defined as the instructional strategy that provides opportunities for the student to participate actively in the learning activity [9]. Research has shown that learners learn most effectively when they are actively engaged in learning, as opposed to passively reading or listening [10]. The PLH0 courseware contains numerous interactive elements that aim to actively engage students in the learning process and to support deeper and more meaningful learner-centered learning. These interactive elements include pop-up boxes, multiple-choice quizzes (explained in section 4.2) and user-controlled animations.

3.1.1 Supporting Comprehension with Pop-up Boxes

Pop-up boxes provide students the ability to view explanatory or complementary pieces of information which can consist of descriptive text and/or images. While there exist numerous pop-up boxes throughout the courseware in order to facilitate the interaction with the system and the presentation of the material, there are two categories of pop-up boxes that are systematically used: code pop-up boxes and term pop-up boxes.

Code pop-up boxes

Code pop-up boxes support comprehension by providing explanations on any program or code fragment that is included in the courseware. They are invoked simply by positioning the mouse pointer over any line within a program's code and are deactivated when the mouse pointer is removed. The line or block of lines of code that the explanation pop-up box is referring to is/are highlighted in order to assist the student in relating the explanation with the code (figure 1). The purpose of providing detailed explanations for the code is twofold: first, to assist the learners in better understanding the specific commands used and second, to present the algorithmic approach of the solution or parts of the solution. The diversity of the background knowledge of the students though, required to discover a way that facilitates any type of learner. More specifically, some students may be able to understand a program without the detailed clarifications provided in the pop-up boxes while some other students may need line-by-line detailed explanations. If these additional clarifications were included within the reading material, fast-learners would have to read long, tiring and possibly uninteresting information. Providing the choice to dynamically invoke the explanations constitutes the best possible solution for any type of learner.

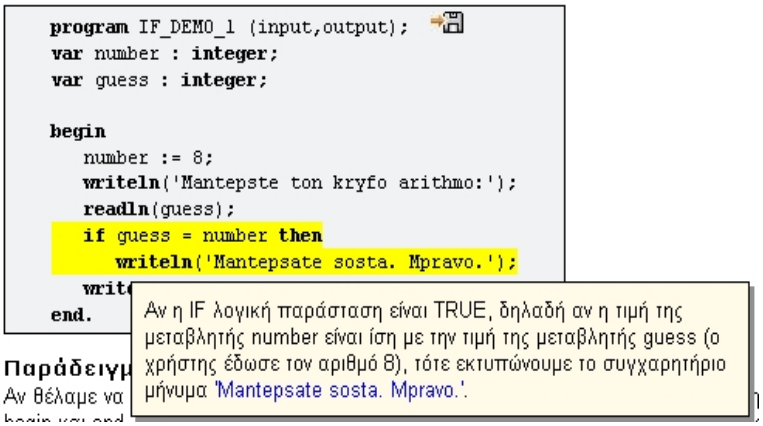

Figure 1 A code pop-up box providing explanatory information for the highlighted code.

Term pop-up boxes

Term pop-up boxes include definitions and explanations of important terms and concepts (e.g. compiler, high-level language, control structure, data structure etc). As the material progresses and more terms and concepts are introduced building on previous knowledge, some students may need a quick refresh of previously explained terms. This type of pop-up boxes (figure 2) provide students with an efficient and very quick method of reviewing the definition of a term or a concept, simply by positioning the mouse pointer over the name of the term without having to spent to navigate and find the section that the term was first explained. Thus, comprehension of studied material is supported by providing assistance while at the same time operational conflicts and frustration are minimized.

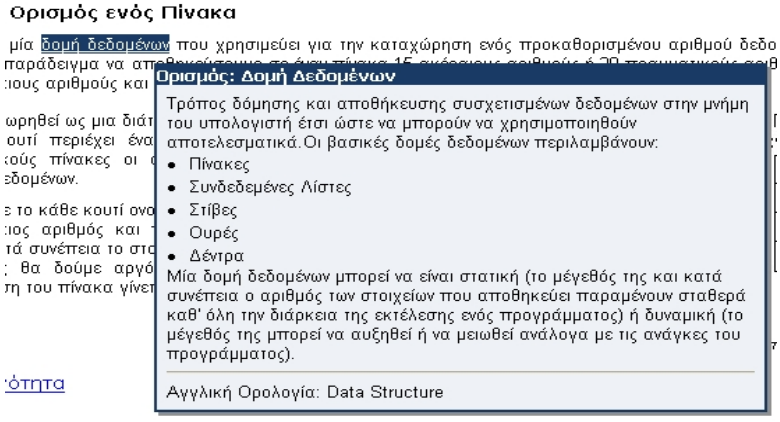

Figure 2 A term pop-up box that provides the definition of a term

3.1.2 Animations

Teaching and learning of algorithmic problem-solving is a difficult task since it is an abstract activity [11]. The steps of an algorithm that solve a specific problem may include branching and looping that may impose conceptual difficulties to the students. An instructional method that can greatly enhance the understanding of these conceptual difficulties is visualization and more specifically animations. The PLH0 courseware includes very carefully designed animations that aim to augment the algorithmic thinking approach and support the comprehension of the concepts that are involved. Each animation (figure 3) includes two views of an algorithm (pseudocode and flowchart), the contents of the memory, the output of

the algorithm and textual explanations. Students can control the pace of the animation and see the execution of the algorithm on a step-by-step basis. More specifically, at each step of the execution, the student is presented with a clear relationship between the two views of the algorithm, the changes of the contents of the memory, the output of the specific step and finally, a detailed explanation.

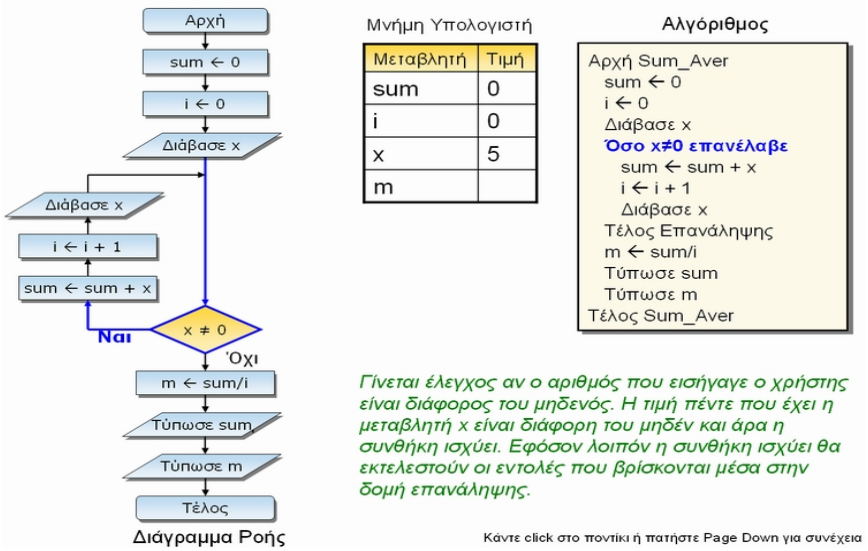

Figure 3 An animation consisting of four different but related elements.

3.2 Self-Assessment Methods

Assessment, both formal and informal, is an important part in an education system since it can provide feedback on teaching and learning, and diagnose learner strengths and weaknesses [12]. Well-designed formative and summative assessment methods, directly connected with the learning outcomes, with timely and constructive feedback, can improve student learning and enhance the educational experience [12]. The PLH0 courseware provides students with the ability to evaluate their understanding of the studied material and their overall progress through two types of self-assessment: practical exercises and interactive quizzes.

3.2.1 Practical exercises

Practical exercises are a form of formative self-assessment method. For every important concept that is examined, there exists at least one related practical exercise that aims to assist students in realizing their critical understanding of the concept. These continuous exercises which provide the opportunity for hands-on experimentation are divided into two types:

- The first type of exercises displays to the learners an algorithm (flowchart or pseudocode) or a piece of source code and asks them to interpret and explain what is being carried out.
- The second type of exercises, building on the first type, goes one step further by asking learners to develop a solution and implement a program that corresponds to a given problem description.

Finally, every exercise is accompanied by a model solution and a detailed explanation. These are included for two important reasons: first, to provide learners with the opportunity to have immediate feedback by comparing their solution with the model answer and second, to assist students who encounter difficulties at a certain point of their attempted solution and wish to overcome it by taking a glimpse of the model answer.

3.2.2 Interactive Quizzes

The second type of self-assessment method of PLH0 is the use of interactive quizzes. They can be considered as a summative form of assessment since they can be found the end of each chapter. Each quiz consists of fifteen multiple-choice questions which are gradually advancing in level of difficulty. These quizzes however, do not have the formal meaning of the term i.e. they do not provide an assessment mechanism by counting the correct responses and presenting a score to the students. Rather than this, they are mostly used as an instruction tool since the answers to a multiple choice question include fine differences of the concept that is being examined. As a result, the quizzes provide the opportunity to students to further enhance their understanding of a concept by exploring these fine differences. Immediate, constructive feedback which pinpoints and explains these fine differences is provided whenever a wrong answer is selected. Last but not least, while students are allowed to answer a question more than once, the number of unsuccessful attempts is recorded. This rationale is conveyed to the students by explicitly visualizing unsuccessful attempts as depicted in figure 4.

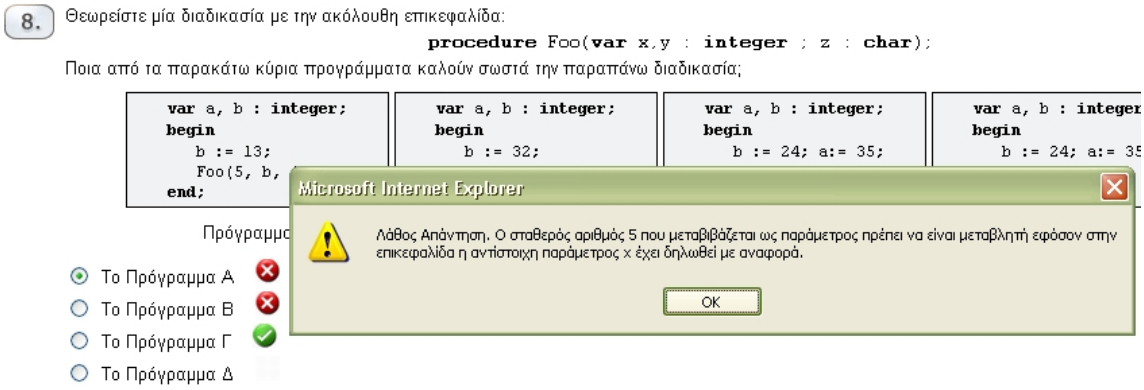

Figure 4 System's response to a multiple choice question.

3.3 Minimizing Overload through Real-Life Examples

Numerous textbooks that teach programming concepts and algorithmic thinking utilize mathematical examples. While it is true that mathematics and geometry present endless possibilities for examples appropriate for developing algorithmic thinking, real-life examples help to set the stage for easier and more stimulating problem-solving. When concepts are presented in the context of real-life situations, students become more interested, engage better in the learning process and thus, acquire and retain information most effectively [13] Furthermore, the use of practical, everyday scenarios in order to explain concepts allows adult learners to build from background knowledge and thus avoiding cognitive overload. In the PLH0 courseware, algorithmic problem-solving concepts such as selection, repetition, subprograms etc. are explained and demonstrated through examples and case studies related to everyday activities such as banking transactions, payment of bills, etc. Adult learners should relate easier to these real-world examples minimizing in this manner a

possible cognitive overload that would exist if classic examples where used from the areas of mathematics and geometry.

3.4 Providing Multiple Representations

Learning with multiple representations has been recognised as a potentially powerful way of facilitating understanding [14]. Learners, for many reasons, have vastly different learning styles and instructional material must take into consideration different learning styles and the possibilities offered in and by the multiple representation of concepts [15]. Furthermore, students learn a complex concept if they experience the information in various formats [16]. As a result, multiple representations can complement each other, resulting in a more complete representation of an application domain than a single source of information does [17].

Algorithmic problem-solving structures and programming concepts can be presented and explained through multiple representations: in textual format, with a diagram (e.g. flowchart), with pseudocode or with a program. In order to support the learners and to allow them to produce more complete images of the concept being examined, the PLH0 courseware includes multiple representations of concepts (figure 5).

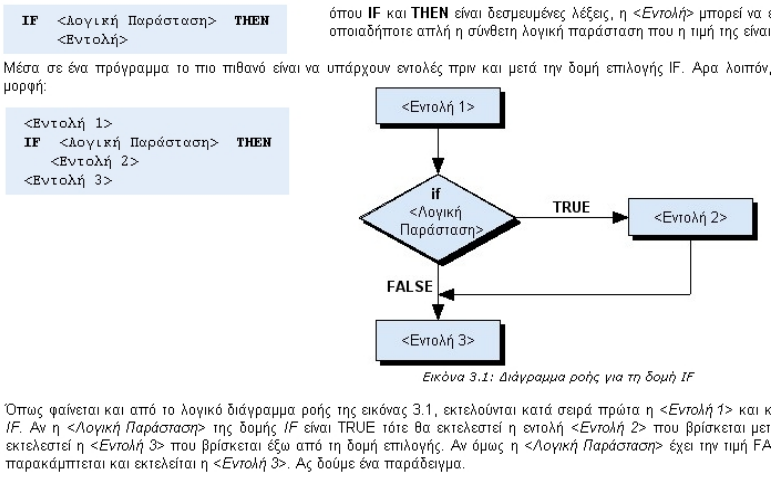

Figure 5 Multiple representations of a selection structure.

3.5 Interface Design

An effective interface design sets the stage for meaningful learning to take place and motivates the learner to stay engaged [18]. The PLH0 courseware was very carefully designed in order to minimize operational conflicts and reduce cognitive overload. Emphasis was given to the following three aspects:

Navigation: An intuitive and unambiguous navigational module offers learners with the freedom of choosing their own path through the content and thus providing control and autonomy, but also provides a clear and well-defined structure to those learners who need support.

Content Organization: The material is organized into logical teaching units with clearly defined aims, learning outcomes and prerequisites, in order for the students to have an

unambiguous picture of each learning unit in terms of what is included, what is anticipated to be gained after completion and previous knowledge that is required.

Layout: A hierarchical organization of the teaching unit and a consistent pattern of the layout which dynamically adapts to the user's monitor settings, ensure maximum functionality and legibility.

4. Technologies Used

Many technologies are available for developing interactive educational material, some of which are very powerful and can produce a highly interactive and visually appealing environment. However, based on the specifications that were set by the HOU and discussed in section 1, it was decided that the most appropriate approach was to develop a web-based environment with HTML documents. The popularity of the Web means that many students may already know how to use a web browser but even if they don't, the simple and intuitive environment of a web browser ensures that the effort involved for understanding the interface is kept to minimum. Additional technologies that were used for the development of the courseware include Javascript, a simple, powerful, fast, efficient and yet inexpensive technology, that adds dynamicity and interactivity to HTML documents, and Cascading Style Sheets (CSS), a stylesheet language that describes the presentation of the content and ensures consistency.

5. Conclusions

This paper describes the pedagogical issues and the design strategies for the development of distance learning courseware. Targeted to adult learners of the HOU distance learning CS programme, who are deprived of or have very little previous experience in computer use, and have been away from school for quite a few years, the developed courseware attempts to facilitate students' comeback to the educational process. Based on scaffolding strategies, the paper describes the learning environment and the individual components of the courseware that address the support needs of the learners. The courseware is currently being deployed at the HOU and is to be used with students in 2008. We plan to carry out an evaluation study of the courseware to measure its effectiveness both in terms of pedagogy and skill development as well as acceptance level by the students.

References

1 M. Xenos, C. Pierrakeas and P. Pintelas, A Survey on Student Dropout Rates and Dropout Causes Concerning the Students in the Course of Informatics of the Hellenic Open University, Computers and Education Journal, 39(2002) pp 361-377

2 Wood, D., Bruner, J. S., and Ross, G. (1976). The Role of Tutoring in Problem Solving. Journal of Child Psychology and Psychiatry, 17, 89 – 100

3 Wilhem, J., Baker, T., Dube, J., Strategic Reading: Guiding Students to Lifelong Literacy, Heinemann (Division of Reed Elsevier Inc.), New Hampshire, USA, 2001

4 McLoughlin, C., Learner Support in Distance and Networked Learning Environments: Ten Dimensions for Successful Design, Distance Education, Vol. 23, No. 2, 2002

5 McLoughlin, C., Marshall, L., Scaffolding: A model for learner support in an online teaching environment, Proceedings of the Teaching and Learning Forum, 2000

6 Ludwig-Hardman, S., Dunlap, J. C., Learner Support Services for Online Students: Scaffolding for success, International Review of Research in Open and Distance Learning, Volume 4, Number 1, 2003

7 Oliver, R., Exploring Strategies for Online Teaching and Learning, Distance Education, 20(2), pp. 240-254, 1999
8 Kao, M., Lehman, J., Cennamo, K., Scaffolding in Hypermedia Assisted Instruction: An example of integration, Paper presented at the convention of the Association for Educational Communications and Technology, ERIC, 1996
9 Hannafin, M., Hill, J. R., Oliver, K., Glazer, E., Sharma, P., Cognitive and Learning Factors in Web-Based Distance Learning Environments, Handbook of Distance Education, pp.245-272, IEA Publishers, 2003
10 Brooks, D. W., Web-teaching: A guide to designing interactive teaching for the World Wide Web, Plenum Press, New York, 1997
11 Pareja-Flores, C., Velázquez-Iturbide, J. Á., Program Execution and Visualization on the Web, Web-based Education: Learning from Experience, pp. 165-188, Information Science Publishing, 2003
12 Born, A. D., Web-Based Student Assessment, Web-based Education: Learning from Experience, pp. 165-188, Information Science Publishing, 2003
13 Willoughby, J., Finding Science in the Real World, Glencoe/McGraw-Hill, New York, 2000

14 Ainsworth, S., Van Labeke, N., Using a Multi-Representational Design Framework to Develop and Evaluate a Dynamic Simulation Environment, Proceedings of the International Workshop on Dynamic Visualizations and Learning. Tübingen, 2002
15 Sankey, M., Multiple Representations in Instructional Material: An Issue of Literacy, Proceedings of Educause, Australia, 2003
16 Jacobson, M. J., Spiro, R. J., Hypertext learning environments, cognitive flexibility, and the transfer of complex knowledge: An empirical investigation, Journal of Educational Computing Research, 12, 301–333
17 Bodemer, D., Ploetzner, R., Encouraging the Active Integration of Information During Learning with Multiple and Interactive Representations', Proceedings of the International Workshop on Dynamic Visualizations and Learning, Tübingen, 2002
18 Klassen J., Preproduction Stages in Multimedia Development: Conceptualization and Scriptwriting, Web-based Education: Learning from Experience, pp. 90-104, Information Science Publishing, 2003