A Web-based Case-based Learning Environment – Use in the Didactics of Informatics

Maria Boubouka*

National and Kapodistrian University of Athens
Department of Informatics and Telecommunications
mboub@di.uoa.gr

Abstract. The present dissertation examines the didactic utilization of cases, specifically through digital learning environments. The review of existing digital case-based learning environments indicated the need for the development of an environment that enables the creation of learning activities of different types based on one or more than one cases. Towards this end, the design of the learning environment CASTLE (CASes for Teaching and LEarning) is proposed. Innovative characteristics of CASTLE concern: the distinction of cases and activities that are handled as different entities, the generic design that can be specified for teaching different subject matters and the support of interaction between users. A prototype of CASTLE for teaching Programming has been created and is described, along with a proposed template of Programming cases which is supported by experimental evidence. Finally, the evaluation of CASTLE prototype and an investigation for the specification of CASTLE in Didactics are presented.

Keywords. Cases, Problems, Learning Activities, Educational scenarios, Didactics of Programming

1 Introduction

One of the most salient goals of modern education is the development of problem solving skills [1-2]. Cases which are context-rich problem descriptions are essential ingredients of learning environments aiming at enabling students to deal with problems. Context information included in cases helps students understand the meaningfulness of dealing with the problems and therefore increases their motivation for engagement in the problem solving process [3-4].

In bibliography there is neither a unique definition for the term case nor a consensus for case content or structure [5-7]. This is due to the fact that cases are being used in a great variety of learning activities, some of which demand only the problem description and others necessitate that this description be accompanied by one or more

* Dissertation Advisor: Maria Grigoriadou, Emeritus Professor

solutions [8]. Jonassen [9] characterizes cases as "building blocks of problem-based learning environments" and summarizes seven different ways of case utilization in teaching: 1) cases as problems to solve, 2) cases as worked examples, 3) case studies, 4) cases as analogues [10], 5) cases as prior experiences [11-13], 6) cases as multiple perspectives [7], 7) cases as simulations. The aforementioned applications of cases differ in terms of: a) the structuredness of the problems (well structured or ill structured) [1], b) the number of different cases that are being involved (one or more than one case) and c) how the case is being used i.e. as an example to be studied or as a problem to be solved. Within the wide range of different learning activities that can be designed based on cases there have been included activities adequate for both novices and experts on a subject matter. Moreover, the fact being that case authoring is a rather demanding task, the need to maximize case utilization through the creation of more than one learning activities based on one case is emphasized.

In the last decades a large number of digital case-based learning environments have been developed, covering a variety of learning subject matters. The extended review performed, and analytically described in the dissertation, focused on the structure and content of cases as well as on the types of activities based on the cases in the examined learning environments.

Considerable differences have been noticed in the length of the cases included in the reviewed environments. Cases in the form of story problems in learning environments for Mathematics [14] or Physics [15] are rather short (a couple of paragraphs), while cases describing design problems in Architecture [16] or Computer Engineering [17,18] are quite long (multiple pages, collections of files connected by analytical narratives of the problem situation and the solution process). Moreover, besides the problem description, a case may contain or not the description of its solution process given by an expert.

As far as the case-based learning activities are concerned, each learning environment usually contains one single type of activity. In particular, there are environments where students are asked to solve the case problems [19]; others where students can study cases already solved by experts [16] and others where students have to participate in group discussions based on one case [20]. In another group of environments students working on a case are engaged in a sequence of activities, which is always the same for all cases included in the environment [21]. Finally, only a small number of environments support activities asking for comparisons between cases [22].

Furthermore, few learning environments allow case authoring [23]. In the majority of the environments the case collection is created by their developers and no authoring functionalities are available for the final users (instructors or learners). Finally, none of the reviewed case-based environments have social software functionalities that would allow the exchange of comments on the learning material between users.

To sum up, the review of existing case-based digital learning environments has indicated that the creation of distinct learning activities based on one case remains an open research issue. The development of such a learning environment is of critical importance as there are case-based activities adequate for both novices and advanced learners in a subject matter. Moreover, case authoring is an exacting task. Thus, a learning environment that supports case authoring and case reuse in distinct case-

based activities may consist a powerful tool for the instructor by enabling him/her to create learning activities adapted to the educational needs of his/her students.

Towards this end the present dissertation proposes the design of the learning environment CASTLE (CASes for Teaching and LEarning), which aims to support teaching and learning through cases. The design of CASTLE is generic and can be adapted to the specific needs of a learning subject matter through the specification of the design of cases and case-based learning activities. In the dissertation experimental evidence is presented concerning: the development of a prototype of CASTLE for teaching Programming, the evaluation of this prototype by novice and expert Programming teachers who used the environment as authors, and an investigation for the specification of CASTLE in Didactics.

2 The design principles of CASTLE

The main requirement for the design of CASTLE was to support the authoring of cases as well as the authoring and elaboration of different types of learning activities based on cases. Within the different types of activities, the same case may play a role ranging from being an already solved problem offered to be studied as an example, to being presented as a novel problem to be solved. Other types of activities may involve two or more cases. Such an activity asks students to compare the solutions of two isomorphic case problems (different story, same procedure)[24] in order to identify structural similarities, or to study a solved case and engage in solving an isomorphic one. This fundamental requirement is being met in CASTLE by handling cases and case-based learning activities as distinct entities which are interrelated between them.

Three different user roles are being supported in CASTLE: teacher, learner and administrator. In CASTLE a teacher can: author cases and case-based activities; organize learning activities in collections; create sequences of learning activities in order to form educational scenarios; assign these scenarios to learners and follow their elaboration. Learners engage in the elaboration of educational scenarios individually or in groups. Learners are also allowed to author cases themselves. Administrators in CASTLE are responsible for the management of user accounts, subject matters and learning material in the form of cases, activities and scenarios.

Additionally, CASTLE supports the interaction between the groups of teachers and learners thus supporting the formation of communities. In other words, CASTLE has social software functionalities [25]. For example, a case author, after publicizing a case in CASTLE, can receive both verbal comments on the case and an arithmetical grade from 1 to 5 by other users, teachers and learners. The received comments and feedback can facilitate the optimization of the learning material as authors have the possibility to commit improvements. Every case is owned by its author in the sense that only its author can make changes on a case and create an updated version of it. However, no ownership is required on a case in order to create a learning activity based on it. This means that there are interactions between users during the creation of the learning material.

Furthermore, data from user interaction is collected in CASTLE while students elaborate educational scenarios. This data is used as an input to the open learner model [26, 27] of CASTLE and enables the provision of personalized feedback to the learners.

The generic design of CASTLE presented here needs further specification in order to meet the special demands of each learning subject matter. This procedure takes place through the selection of case content and structure as well as the types of case-based learning activities adequate for a given subject matter. In the following paragraph the specification process leading to the prototype of CASTLE for Programming teaching and learning is described.

3 Specification process of CASTLE in Programming .

3.1 Specification of case structure

In order to specify the case structure for the subject matter of Programming, a review of the relevant literature was conducted [28]. Two different Programming *case study* structures have been found in bibliography: Structure A proposed by Linn and Clancy [29] and Structure B proposed by Spooner and Skolnick [30] (see Table 1). These structures have both similarities and differences when compared. For example, both structures propose that a case should contain the problem description (*Programming problem statement* in Structure A, *Motivation*, *Background* in Structure B), an explanation for the problem solution (*Solution process description* in Structure A, *Algorithm development* and *New Programming concepts* in Structure B), the code of the program produced to solve the problem (*Code listing* in Structure A, *Solution program* in Structure B) and questions on the solution (*Study* & *Test questions* in Structure A, *Discussion* & *Further study* in Structure B).

Table 1. Programming Case study structures

Structure A	Structure B
Programming Problem statement	Motivation
Solution process description	Background
Code listing	Algorithm development
	New Programming concepts
	Solution program
Study questions	Discussion
Test questions	Further study

In CASTLE, *Cases* and case-based *Activities* are considered different entities and are handled independently. In the *Case study* structures described above, there are parts that correspond to the *Case* entity of CASTLE and parts that correspond to the *Activi-*

ties (e.g *Study & Test questions* in Structure A and *Discussion & Further study* in Structure B). The comparison between the two structures shows that differences reside in the *Case* part of the *Case studies* and, more specifically, in the parts referring to the explanation of the problem solution. In particular, Structure A proposes an explanation of the critical decisions reached by an expert programmer while writing the solution program. On the contrary, Structure B suggests a gradual introduction of the solution, starting with the description of the algorithm development and proceeding with the new programming concepts required in the program code.
In order to select the structure of Cases in the CASTLE prototype for Programming teaching, the two structures have been compared in two empirical studies.

First empirical study.

The first empirical study aimed to compare the two structures in terms of their efficiency and their acceptance by the learners.

In particular, the research questions were:

- does the structure of a programming case affect the ability of learners to develop programs for resolving similar problems?
- does the structure of a programming case affect the time required to study it?
- what are the opinions of the learners about the structure and the usefulness of cases?

Participants.

102 first-year students participated in the study. They had enrolled to the course "Introduction to Informatics and Telecommunications" at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens. The students formed two groups of 51 members each: Group 1 and Group 2.

Procedure.

The empirical study consisted of the following phases:

- Phase A, *Pre-test* (30 min): the students of both groups worked on a programming problem and were asked to develop a program in order to solve it.
- Phase B, *Case study* (60 min): the students of Group 1 worked on a Case study structured according to Structure A and the students of Group 2 worked on a Case study structured according to Structure B. The case problem for both groups was the same.
- Phase C, *Post-test* (30 min): the students of both groups worked on a problem isomorphic to the problem of the pre-test phase. As in phase A, their task was to develop a program.
- Phase D, *Filling the questionnaire* (15 min). All students were asked to fill in a questionnaire on their opinions on the Case study assigned to them in phase B.

Students were also asked to write down in their response sheet the actual time they had spent on the completion of phases A, B and C.

Results.

The programs developed by the students during the pre-test and post-test phases have been evaluated independently according to criteria set by two experienced Programming teachers who assigned a five-point scale grade (0-4) for each criterion. The total grade of each program has been calculated as the mean value of the distinct criteria grades. Disagreements between evaluators were resolved through discussion.

Pre-test performance between the two groups has been compared using an independent samples t-test. No significant difference was found in the t-test performed (t(100)= 0,107, p=0,915) (mean pre-test scores Group1=2,5620, Group2=2,5375).

Subsequently, repeated General Linear Model (GLM) measures analysis of variance on students' performances in pre- and post-test, with Case (pre-test vs. post-test) as the within-subjects factor, and Case structure (Group1 vs. Group2) as the between-subjects factor has been conducted. The corresponding multivariate tests revealed that the Case factor contributes to the improvement of the post–test performance of both groups'. The "Case x Case structure" interaction yielded no significant effects (mean post-test scores Group1=2,995, Group2=2,743). The results of the repeated measures analysis are presented in Table 2.

Table 2. Multivariate tests for time factor and for the interaction Case x Case structure

Factor	F(1)	*p*
Case	11.68	0.001
Case x Case structure	1.48	0.226

As can be seen in Table 2, no evidence was found supporting the assumption that Case structure may affect the ability of students to develop programs that resolve similar (to the case) problems.

Data concerning the other two research questions has also been analyzed. In specific, this data concerned: a) the time students spent studying the Case in Phase B, b) the students' opinions on the structure and usefulness of Case. The data analysis showed that: a) there was a significant difference in the time students of the two groups spent studying the Case, with students of the Group 1 spending less time than those of Group 2, b) there was a significant difference of opinions of students of the two groups about the redundancy of the information contained in the Cases, with students of the Group 2 claiming more often that some parts of the Case they worked on should be omitted.

Second empirical study.

The second empirical study focused on the case authoring process aiming at the investigation of the difficulties pre-service Computer Science teachers face while acting as case authors. In this framework, the proposed case structures (Structure A and Structure B) have also been compared. Significant differences have been noticed in the frequency pre-service teachers select Structure A and Structure B templates, with Structure A being preferred by the majority of authors. Moreover, significantly less

problems in the case structure have been observed to cases structured according to Structure A than those structured according to Structure B.

Case template in CASTLE Programming prototype.

Evidence from both empirical studies described above has been considered in order to design the Case template for CASTLE Programming prototype. Namely, the selected template followed the Structure A outline containing the parts: *Problem* where the case problem is described, *Solution* where the solution program code is listed and *Explanation* where the most important decisions in the development of the solution program are commented.

3.2 Specification of case-based activities

Next, the specification of CASTLE Programming prototype proceeded with the selection of the case-based activities types. Ideas for the types of activities were found in the Didactics of Programming bibliography, including the Programming case studies structure proposals described above as well as digital learning environments specialized in Programming instruction.

Some of the selected types of activities engage learners to respond to questions after studying an entire case and others demonstrate only some of the case parts and require the completion of the case by the learners. For example, in *perturbation* activities, the entire case template (including the parts *problem*, *solution* and *explanation*) is presented to learners. Learners have to study the proposed solution and explanation thoroughly and solve a new problem themselves. The new problem is created by the alteration of the case problem conditions. On the contrary, in *explanation* activities only the *problem* and *solution* parts of the case are being presented to the learners who have to complete the explanation part themselves. In another set of activities, two or more cases are involved. For example, in *comparison* activities students have to study two entire cases and find similarities and differences in their solutions.

3.3 CASTLE Programming prototype outline

CASTLE layout.

The main screen of CASTLE is divided into three areas: the concept tree (Figure 1, A), the information frame (Figure 1, B) and the presentation tabs (Figure 1, C). Cases appear as leaves in the concept tree, appended from the concept they refer to. Once a user clicks on the name of a given case, the case unfolds into a new presentation tab. Simultaneously, a list of all activities based on this case appears in the information frame. A user may select an activity from the list in order to view it in a new presentation tab.

Case management.

Case authoring in CASTLE is a two-step activity. In the first step the author fills in the case name, the problem description, the problem constraints (if any), the main concept and (optionally) the group of cases containing isomorphic problems. In the second step the problem solution (code listing) together with the explanation of critical decisions are completed. This enables the submission of multiple solutions to a given problem description. Every solution is divided into parts which are labeled according to their function. Explanation is listed below the solution, following the same labels.

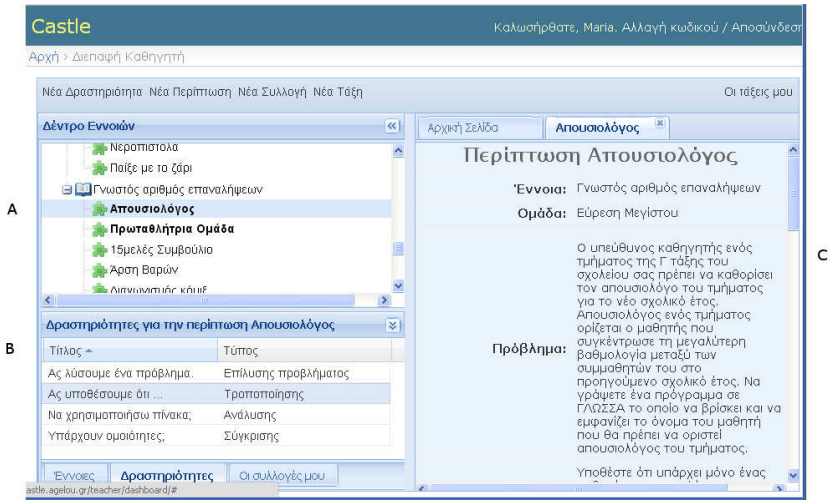

Fig. 1 CASTLE Snapshot where A. the concept tree, B. the information frame & C. the presentation tabs

Activity management.

CASTLE facilitates the case activity authoring by appropriately presenting and hiding parts of a case according to the selected activity type. An activity author should complete the following steps in order to create an activity:

1. Name the activity
2. Select the activity type out of the list of activities included in CASTLE
3. Select the case (or the cases) the activity is based on
4. Write the activity question
5. Provide information about: the estimated difficulty level (1 to 5) and the required elaboration time (in minutes).

An activity author may base his/her activities on cases previously authored by him/herself, but also on cases authored by others. If the adequate case is not already included in CASTLE, the author should first create the case and then proceed with the creation of the activity.

Comment exchange functionalities.

CASTLE users may exchange comments on the cases and activities included in CASTLE. This commenting mechanism provides authors with feedback on the learning material they have created thus enabling them to make improvements. In specific, a case author may receive both verbal comments and a numerical evaluation in a five-point scale (1 to 5). Activities authors, in addition to the verbal and numerical evaluation, receive feedback concerning the estimation of other users on the activity difficulty level (1 to 5) and the required elaboration time.

Other functionalities.

CASTLE supports the creation of *educational scenarios*, which are sequences of learning activities. In order to help teachers create their scenarios, CASTLE provides them the opportunity to organize sets of activities in *collections*. Moreover, a teacher is able to monitor and support the elaboration of scenarios by his/her students who work with CASTLE.

3.4 CASTLE Programming prototype evaluation

The prototype of CASTLE for Programming has been used and evaluated by a group of 66 pre-service programming teachers and a group of 17 expert programming teachers. Pre-service teachers were four-year students at the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens who had enrolled to the course “Introduction to Informatics and Telecommunications”. Expert teachers were in-service teachers of Informatics with many years of teaching experience in secondary education. Teachers had to work with CASTLE prototype in order to create cases and case-based learning activities. Next, they were asked to submit comments on cases and case-based learning activities authored by their peers. At the same time, they had to fill in an evaluation questionnaire containing both Likert-scale and open-ended questions. The analysis of the teachers’ answers in the questionnaires revealed that both groups of teachers had expressed positive opinions on CASTLE. In specific, teachers agreed that CASTLE manages to handle successfully cases and activities as distinct entities, supporting users adequately in finding activities based on a given case. Moreover, teachers of both groups found the comment exchange functionalities of CASTLE really useful, recognizing how important it is not only to receive but also to submit comments themselves. Few significant differences on the opinions of the two groups of students have been observed. For example, pre-service teachers responded that CASTLE is more adequate for beginner learners in Programming than it is for more advanced learners. Expert teachers do not share this opinion as they consider CASTLE equally useful for learners of all levels. Suggestions for the next versions of CASTLE have also been collected from the evaluation of the prototype. Evaluators indicated that help information should be upgraded. Finally, interesting ideas for layout improvements have been reported and will be taken into consideration.

4 Specification process of CASTLE in Didactics.

The specification of CASTLE in Didactics has also been investigated in the framework of this dissertation. Many examples of case utilization in teacher education have been reported in the last years. An empirical study was conducted to examine whether the way a case is being studied (through an online learning environment or in class) affects the ability of teachers to design learning activities [31]. No significant differences have been reported. The examination of how case-based activities of Didactics can be implemented through a digital learning environment provided evidence that CASTLE design is adequate for this subject matter as well.

5 Conclusions

The review of relevant bibliography indicated the need and usefulness of a learning environment that enables the authoring of learning activities of different types based on one case, as this remains an open issue in the domain of digital case-based learning environments. There is a large variety of learning activities that may be created on a given case, including activities both for novices and for advanced learners. Additionally, case authoring is a demanding task. These facts stress the importance of case reuse in different types of activities.

In the framework of the present dissertation the architectural design of the web-based case-based learning environment CASTLE (CASes for Teaching and LEarning) is presented. CASTLE handles cases and case-based learning activities as distinct entities. An author may create his/her own cases in CASTLE or find interesting cases created by others and use them to create different types learning activities. Some types of learning activities in CASTLE may refer to two or more cases (e.g. comparison activities). Additionally, CASTLE enables the creation and elaboration of educational scenarios that are formed as sequences of case-based learning activities. Finally, comment exchange on cases, learning activities and educational scenarios is supported, enabling authors to constantly improve their learning material.

The specification of the generic design of CASTLE in two subject matters has been also examined. Programming and Didactics have been selected as representative subject matters that contain mainly well-structured and ill-structured problems respectively. The investigation showed that CASTLE may be specified in both subject matters. In Programming, the investigation went further and a prototype of CASTLE for Programming teaching has been developed. The Programming case template used in CASTLE prototype has been selected after the experimental comparison of two templates found in bibliography and is proposed. Finally, CASTLE prototype has been evaluated by pre-service and expert teachers. Both groups of teachers expressed overall satisfaction with CASTLE, found that it succeeds to support users to find the learning activities based on a given case and recognize the importance of receiving and submitting comments on the learning material.

References

1. Jonassen, D. H.: Instructional design model for well-structured and ill-structured problem-solving learning outcomes. Educational Technology Research and Development, vol 45(1), 65-95 (1997)
2. Jonassen, D. H.: What is problem solving. In Jonassen, D. H. (Ed.) Learning to solve problems: instructional design guide, pp 1-18. NJ: Lawrence Erlbaum Associates (2003b)
3. Brown, J.S., Collins, A., Duguid, P. : Situated cognition and the culture of learning. Educational Researcher, 18, 32-42 (1989)
4. Vosniadou, S. How children learn. Educational Practices Series, The International Academy of Education (IAE) and the International Bureau of Education (UNESCO) (2001).
5. Herreid, C. F.: Start with a story: The case study method of teaching college science. NSTA Press: Arlington, VA (2006)
6. Kolodner, J.: Case-Based Reasoning. San Mateo, CA: Morgan Kaufmann Publishers Inc. (1993)
7. Spiro, R. J., Coulson, R. L., Feltovich, P. J., Anderson, D. K.: Cognitive flexibility theory: Advanced knowledge acquisition in ill-structured domains. Technical Report No. 441. Champaign, IL: University of Illinois, Center for the Study of Reading (1988)
8. Golich, V.L., Boyer, M., Franko, P., Lamy, S.: The ABC of Case Teaching, Pew Case Studies in International Affairs. Institute for the Study of Diplomacy, Georgetown University, Washington D. C. (2000)
9. Jonassen, D.H.: Research Issues in Problem Solving. In 11th International Conference on Education Research. (2010)
10. Gentner, D., Loewenstein, J., Thompson, L.: Analogical encoding: Facilitating Knowledge transfer and integration. In Forbus, K. D., Gentner, D., Reiger, T. (Eds.) Proceedings of the 26th Annual Conference of the Cognitive Science Society, pp 452–457.Cognitive Science Society, Austin, TX (2004)
11. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications. IOS Press, vol7 (1), 39-59 (1994)
12. Kolodner, J.L.: An introduction to case-based reasoning. Artificial Intelligence Review, vol 6(1), 3-34 (1992)
13. Maher, M. L., Balachandran, M. B., Zhang, D.: Case-Based Reasoning in Design. Lawrence Erlbaum, Hillsdale, NJ. Lawrence Erlbaum Associates(1995)
14. Derry, S. J., the TiPS Research Group: Development and Assessment of Tutorials in Problem Solving (TiPS): A Remedial Mathematics Tutor. Final Report to the Office of Naval Research (N00014-93-1-0310), Wisconsin Center for Education Research. University of-Wisconsin-Madison, Madison, WI (2001)
15. Jonassen, D. H.: Designing research-based instruction for story problems. Educational Psychology Review, vol 15(3), 267-296 (2003a)
16. Zimring, C.M., Bafna, S., Do, E.: Structuring cases in a case based design aid. Vanegas J., Chinowsky. P. (eds), pp. 308-313 (1996)
17. Carroll, J.M., Rosson, M.B.: Case studies as minimalist information. IEEE Transactions on Professional Communication, vol49(4), 297-311 (2006)
18. Xiao, L., Carroll, J., Rosson, M.: Support Case-Based Authentic Learning Activities: A Collaborative Case Commenting Tool and a Collaborative Case Builder. Paper presented at the Human-Computer Interaction. HCI Applications and Services, pp. 371-380. Beijing, China (2007)

19. Riedel, J., Fitzgerald, G., Leven, F., Toenshoff, B.: The Design of Computerized Practice Fields for Problem Solving and Contextualized Transfer. Journal of Educational Multimedia and Hypermedia, 12(4), 377-398. Norfolk, VA: AACE (2003).
20. Rocha, A.R., Moreira, G., Campos, F., Rabelo, L.: CardioCaseDiscussion: a cooperative learning environment for patients' cases discussion in Cardiology. In Barker, P., Rebelsky, S. (Eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2002, pp. 1371-1373. Chesapeake, VA: AACE (2002)
21. Choi, I., Lee, S. Jung, J.: Designing Multimedia Case-Based Instruction accommodating Students' Diverse Learning Styles. Jurnal of Educational and Educational Multimedia and Hypermedia. Vol17(1), 5-25 (2008)
22. Leclet, D., Joiron, C.: Design of a Distance Learning Environment - DIACOM : An Interactive Forum Based on Collaborative Learning for Continuing Medical Eduaction. In Barker, P., Rebelsky, S. (Eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2002, pp 876-881. Chesapeake, VA: AACE (2002)
23. Rudy, M., Jaksch, S.: Building Science by Design: Developing a Building Information System and Teaching Architecture with Analytical Case Studies. ED-MEDIA 2004: World Conference on Educational Multimedia, Hypermedia & Telecommunications, Association for the Advancement of Computing in Education, pp. 4815-4820. Norfolk (VA) (2004)
24. Reed, S. K.: A structure-mapping model for word problems. Journal of Experimental Psychology: Learning, Memory, and Cognition, 13, 124–139 (1987)
25. Shirky, C.: A Group is Its Own Worst Enemy. Networks, Economics, and Culture mailing list, Retrieved 21 2012 from:. http://www.shirky.com/writings/group_enemy.html (2003)
26. Bull, S.: Supporting Learning with Open Learner Models. In Proceedings of 4th Hellenic Conference in Information and Communication Technologies in Education, pp. 47-61. Athens (2004)
27. Bull, S., Kay, J. : A Framework for Designing and Analysing Open Learner Modelling. Proceedings of Workshop on Learner Modelling for Reflection. In International Conference on Artificial Intelligence in Education, pp. 81-90 (2005)
28. Boubouka, M., Verginis, I., Grigoriadou, M. & Zafiri, I. (2009) "Towards the enhancement of the learning process with different types of case based activities" in Proceedings of the 8th IEEE International Conference on Advanced Learning Technologies (ICALT2009), Riga, Latvia, 15-17 July 2009, pp 652-654.
29. Linn, M., Clancy, M.: The case for case studies of programming problems. Communications of the ACM, vol 35(3), 121-132 (1992)
30. Spooner, D. & Skolnick, M.: Science and Engineering Case Studies in Introductory Computing Course for Non-Majors, pp. 154-158. SIGCSE CA (1997)
31. Boubouka M., Verginis I., Grigoriadou M.. Supporting the Implementation of Case Activities using e-learning. In Spector M. J., Ifenthaler, D., Isaías, P., Kinshuk, & Demetrios, G. Sampson (Eds.), *Learning and Instruction in the Digital Age Technologies*, Springer, 2009, ISBN 978-1-4419-1550-4 (2009).