Problem-based learning of theoretical computer science

Wilhelmiina Hämäläinen
Department of Computer Science
University of Joensuu
P.O. Box 111, FIN-80101 Joensuu, FINLAND
Email: whamalai@cs.joensuu.fi

***Abstract*— In this paper, we report our first experiment in teaching the theory of computability in the problem-based way. As far as we know, this is the first experiment of applying the problem-based method to a purely theoretical course of computer science.**

Performing the course consisted of three parts: First, the new subjects were learnt according to the classical seven step method, which contains both individual and group work, and problem reports were written. Second, the students participated in a traditional exercise session, in which the new techniques were practised in details. And third, the students kept a learning diary, in which they processed the subjects further, tried to construct an overall schema of things learnt, and supervised their own learning.

The results were really successful: the students committed themselves well and the drop out percentage was very small; they achieved very deep understanding of the subjects measured by their grades and quality of learning diaries; the experience was enjoyable for both the students and the teachers; and finally, the method supported different kinds of learners very well.

Index terms – theory of computation, problem-based learning.

I. Introduction

The theoretical courses of computer science are an important but challenging part of the computer science curriculum. Especially courses on the theory of computability have been found very difficult and boring by the computer science students in the Finnish universities. Majority of the students, who begin the course, drop out of it, and some of them try it several times before passing. In the previous years, at most 1/3 of the students who have registered for our course on the theory of computability, have passed it.

According to student feedback, the main reason for the dislike is the mathematical and theoretical nature of the topic. In the theory of computability the problem is even worse than in the courses on the algorithm theory. The most difficult topics to learn deal with problems which are computationally unsolvable, i.e. there is no algorithm to describe the method to be learnt. The Pumping Lemma for regular languages (how to prove a formal language non-regular) or unsolvability proofs are classical examples. In addition, the traditional material for the topic does not introduce any practical applications or focus the meaning of the issues in real-life problems of computer science.

One reason for the difficulties lies in the way of thinking, which is required. In the most of computer science courses students are taught to reason ”algorithmically”, i.e. they are given a reasoning procedure, which they apply to new situations. Bako [1] has recognized the same problem in the scope of mathematics, too: Algorithmically thinking students can solve similar problems easily, but they have great difficulties to solve any unknown type of problems, even if they have all the knowledge required. As a contrast, in logical reasoning (or ”equational reasoning”, as Page [2] puts it) the students learn to define the solution declaratively, and apply the model or method to new problem instances. These abilities have proved to benefit students in other courses, like in software development [2].

Based on these considerations, it is natural to assume that strong mathematical background helps students in the course on the theory of computability. However, mathematical courses are disliked by most of students, and especially in the University of Joensuu the curriculum of computer science does not contain any obligatory courses in mathematics. Thus, the course should teach and give practice in logical reasoning, too.

When we began to design a new approach for this course, our first goal was to change the role of students to be more active. With difficult courses it is especially crucial that the students will not remain just as passive receivers of new information, but they should become active constructors of the new knowledge, as the *constructivist* hypothesis argues (see e.g. [3]).

In our traditional course setting, the students are very passive in the lectures, but during exercise sessions they are more active and also admit to learning more. The exercise sessions have proved to be very effective learning situations, when the students have first solved the tasks individually, but in the exercise classes the tasks are processed in small groups before representing the solutions to the whole class. However, the scope of exercise sessions is limited: usually the tasks involve practising the given techniques and the students learn only fragmented pieces of knowledge. The holistic understanding of the topics, their mutual relations and meaning is missing. Thus, the natural conclusion was to try to apply this technique for larger-scope problems, which would require both individual work and cooperative processing. These are

exactly the elements of problem-based learning, and the only problem was to invent good and interesting overall problems which would lead to the required learning goals.

In the following sections, we will first introduce the principal idea of problem-based learning. Then we will describe our first experiment of teaching the theory of computability in the problem-based way. Finally, we will analyze our results and draw the final conclusions.

II. Problem-based learning method

A. What is problem-based learning?

The main idea of *problem-based learning* (*PBL*) is to use problems, queries or puzzles as a starting point for learning. In fact problem-based learning is not just a single method or technique, but a variety of problem-based approaches, from lecture-based teaching to pure problem-based learning without any teaching or assessment by the teachers [4], [5]. Ellis et al. [6] divide the problem-based learning methods into three categories. In the modest forms, which Ellis et al. call *problem-based approach* the material is presented in normal lectures, but problems are used to motivate students and demonstrate the theory. In the hybrid models or *guided problem-based learning* problems are solved in groups, but also lectures are used to present the fundamental concepts and conceptually most difficult topics. In *full problem-based learning*, the problems guide and drive the entire learning experience and no formal exposition of knowledge from the ”expert” is given.

Boud [4] has listed some general characteristics which are typical for problem-based courses:

- Acknowledgement of learners' experience.
- Emphasis on students taking responsibility of their own learning.
- Crossing of boundaries between disciplines.
- Focus on the processes of knowledge acquisition rather than the products of such processes.
- Change in staff role from instructor to facilitator.
- Student self- and peer assessment of learning.
- Focus on communication and interpersonal skills.

B. Evaluation of the method

Problem-based learning has produced excellent results in medical science [7], which have encouraged other faculties to adopt it in higher education. In computer science problem-based methods have been experimented mainly in the areas of programming and software engineering, but in Linköping University the entire curriculum of Information Technology is problem-based [8]. A lot of good effects have been reported: The students have deeper understanding of the issues and they are better motivated. PBL improves also communication and cooperation skills as well as meta-cognitive skills like problem solving and ways of thinking. Individual learning goals support different learners and also poor students manage well in PBL. In addition PBL gives practice in research and information retrieval. [8], [9], [10]

Ellis et al. [6] have argued that problem-based learning is especially well suited to computer science. Computing discipline matches the characteristics of problem-based learning: computing itself is mostly problem-driven; life-long learning is a necessity in the area and the professionals must constantly update their skills; the project work in groups is the dominating working mode in industry; and finally, computing crosses discipline boundaries.

The evaluation studies agree that problem-based learning produces better skills than traditional education. However, it has been speculated, whether the students acquire as good theoretical knowledge as in traditional education with lectures and exams. Dochy et al. [11], Albanese et Mitchell [12] and Vernon et Blake [13] have made independent meta-analyses of the reports, which compare the problem-based method to traditional education. The conclusions are uniform: immediately after the course the problem-based learners have better skills but slightly poorer knowledge than the traditional learners. However, the difference in knowledge level is statistically quite insignificant and even better results have been reported. The most interesting observation is that after some time the theoretical knowledge of the PBL learners is at least as good as of the traditional learners – i.e. the PBL learners remember better what they have learnt.

The results of problem-based learning match well with constructivist view of learning: during the problem-solving process the students actively construct their knowledge on the basis of their own experience and reflections. It is quite obvious that this kind of processing leads to deep learning instead of memorizing facts for the exam.

However, even a good method can fail, and it is good to be conscious of possible problems. McCracken et Waters [14] have reported about some problems in their experiment on software engineering course: 1) Real world problems are essential for PBL, because the students do not like artificial ”toy problems”, but such problems are often too worksome and difficult even for the facilitators. 2) In the PBL the students may not develop deep learning issues, because they cannot set efficient learning goals, or they concentrate in a particular solution, instead of the general method. 3) The students may abandon the PBL process as they move further into the problem. 4) The students can have difficulties in developing the desired meta-cognitive skills. McCracken and Waters notice that partly the problems are due to staff recourses: enough tutoring by experienced facilitators is essential, especially when the method is new for the students.

As a solution for the first problem Ellis et al. [6] suggest well-structured problems for novice students. For advanced students PBL course should emphasize more open-ended and ill-structured problems, which are typical in real life. For the other pitfalls, the learning diary can be an especially valuable complement [6]. In the learning diary the students are asked to process further what they have learnt, and what was unclear, construct overall pictures, analyze their own learning process and develop the ”meta-skills” of learning. Learning diary offers also an effective forum for feedback between the

facilitator and the student. [15], [6]

III. The course description

A. Arrangement

In our experiment, we applied the problem-based method for teaching theory of computability in the Department of Computer Science, University of Joensuu during spring term 2003. In Joensuu there is only one course on this subject, "Theoretical Foundations of Computer Science" (*TFCS*), which covers the theory of computability from finite automata and regular expressions to context-free grammars, pushdown automata, Turing machines and solvability issues.

The course was given during ten weeks, with one week break, which was reserved for an art exhibition. In the art exhibition the students represented pieces of art they had made with the aid of formal languages: animations, pictures and music by L-systems and poems generated by finite automata.

In our experiment we have adopted the seven step model, which is quite often used in the PBL education of medical science [16], [17]. The seven steps of each PBL cycle are:

1) Defining unclear concepts: The students look for concepts, which are unclear and try to define them.
2) Defining problem: The students discuss about the problem.
3) Brain storming: Students try to construct, test and compare different hypotheses and explanations.
4) Constructing hypotheses: The problem is analyzed carefully by comparing different hypotheses. The ideas are argued and organized into an integrated whole.
5) Defining learning goals: The students write down their learning goals for the self-studying phase.
6) Self-studying: The students acquaint themselves independently with the topic. In this phase also lectures can be offered to support the self-studying.
7) Sharing the results: The students compare their solutions and try to help each other to understand the topic. The learning goals are checked and the final conclusions are drawn.

In this model the steps 1-5 constitute an opening session and step 7 works as a closing session. At least some tutors or facilitators should be available during the group sessions. For the self-studying phase students can be given some reference material, but they are encouraged to use different sources. Also other group members or teacher consultation can work as an information source, if the process gets stuck.

In our experiment, the students had weekly a two hours traditional exercise session, but the lecture time was scheduled in a new way. Half of the lecture time was used in opening and processing the problems in small groups according to the seven step model, and the other half was reserved for lecturing. A couple of times this normal program was replaced by playing problem solving games. During the group work the lecturer wandered in the groups and tutored them. In a couple of the most difficult problems the course assistants were also voluntarily available.

In the first half of the lecture the students first processed the last problem for about half an hour. Then they were given a new problem to be tackled. They were also encouraged to meet the group members during their free time, and at least some groups had very active communication through chat. When the problem reports were returned we still had some general discussion with the whole group and the final conclusions were drawn.

In addition to the problem reports the students wrote learning diaries, in which they were asked to process the learnt subjects, set questions, introduce their own applications, and especially reflect their own learning process. The learning diaries and the problem reports were evaluated, and the course grade composed of the points they had got in problems, learning diary and exercises together.

The students were given freedom to participate either problem-based learning or perform the course by the traditional way with exercises and exams. However, the amount of lectures was only half of normal and they were given only in Finnish. That is why the course was more demanding for the foreign students, even if they had an English-speaking exercise class and problem-solving group.

B. Problems

During the course the students solved a total of 14 problems. Additionally, the first problem was processed along the whole course and especially in the last lecture, when the students played a problem solving game. The problem cases and their learning goals are represented in Table I. It should be remarked that the "official" learning goals were not told to the students, but they were asked to determine their own individual learning goals. In the end of the course the students could compare, what they have learnt against the official learning goals, and thus perform self- and course evaluation.

For example the first problem, which was quite central for the whole course, was the following:

"You are working in a problem solving company, the motto of which is 'What we cannot solve, cannot be solved'. The company has all kind of computer science experts, for example top programmers of all existing programming languages, and super computers with all possible utilities. Your task is to receive the problems of clients and decide, if they are solvable. If the problem is solvable, you give it to a suitable software engineering group and tell, how 'difficult' problem it is, so that they can allocate the required recourses."

The full descriptions of the other problems can be found in `http://www.cs.joensuu.fi/pages/whamalai/tepe/problems.htm`.

In Table II the problems are classified according to whether the initial problem setting, the goal and the methods are *open* or *closed*, although sometimes it is not obvious, into which category they belong. In the open cases the starting point and the problem are vaguely defined and no solution method is suggested. As a contrast in the closed cases the initial setting and stopping point are clearly defined and also the solution method can be hinted at. [18]

TABLE I

PROBLEM CASES AND THEIR LEARNING GOALS.

Problem	Topic
Case 0	Morse alphabet – Basic concepts
Case 1	Problem solving company – Modelling problems, their difficulty and solvability
Case 2	Search for mail addresses and a precompiler for a programming language – Regular expressions
Case 3	Coffee automaton – Deterministic finite automata
Case 4	Nondeterministic editor – Nondeterministic finite automata and determinization
Case 5	Roman checking exams – Finite automata vs. regular expressions, ϵ-automata
Case 6	Grandma's rhyme – Pumping Lemma
Case 7	L-systems – Idea of grammars
Case 8	Parentheses parsing – Pushdown automaton
Case 9	Arithmetic calculator – LL(1)-grammars, recursive parser
Case 10	General parser – The CYK-algorithm and Chomsky normal form
Extra	Attribute grammar and parser tools
Case 11	Summing machine – Basics of Turing machines
Case 12	Library functions for TM's – Deeper practice on Turing machines
Case 13	Programming competition – Universal machines and universal languages, solvability proofs
Case 14	Philosophical considerations – Church-Turing theses, limits of computation.
Case 1 revisited	Problem solving company again – Overview of the principles learnt

TABLE II

CLASSIFICATION OF THE PROBLEM SETTING, GOALS AND METHODS AS OPEN **o** OR CLOSED **c**.

Problem	Setting	Goal	Methods
Problem 0	**c**	**c**	**o**
Problem 1	**o**	**o**	**o**
Problem 2	**o/c**	**c**	**o**
Problem 3	**c**	**c**	**c**
Problem 4	**o/c**	**o/c**	**o**
Problem 5	**o/c**	**c**	**o**
Problem 6	**o**	**o/c**	**o**
Problem 7	**o**	**o**	**c**
Problem 8	**c**	**c**	**o/c**
Problem 9	**c**	**c**	**o/c**
Problem 10	**c**	**c**	**o/c**
Problem 11	**c**	**c**	**c**
Problem 12	**o**	**o/c**	**c**
Problem 13	**c**	**o**	**o**
Problem 14	**o**	**o**	**o**

IV. EVALUATION

A. Results

Totally 77 students registered for the course, of which 75 participated at least the first exercise session. 63 students began the course in the problem-based way, but two of them changed to the traditional method in the beginning. 11 students dropped from the course, seven traditional learners and four problem-based learners. So the proportion of dropouts in the PBL approach was about 7% and in the traditional approach about 50%. 55 students passed the course in the problem-based way, and seven students in the traditional way. I.e. 90% of PBL students, and 50% of the traditional learners passed the course. The exact numbers are summarized in Table III.

It is remarkable that those problem-based learners who dropped from the course did it in the first weeks, when they had seen what the course was dealing with. The problematic dropouts of the traditional courses, who waste their own and staff's resources until the first exam, were missing. Only two problem-based students dropped from the course in the middle of it. In the traditional courses of theoretical computer science usually much less than 50% of students pass the course. Thus we can conclude that the students commit themselves better to a problem-based course than a traditional one.

TABLE III

NUMBERS OF STUDENTS IN THE PROBLEM-BASED (PBL) AND THE TRADITIONAL (TRAD) APPROACHES. TWO STUDENTS CHANGED FROM THE PROBLEM-BASED APPROACH TO THE TRADITIONAL ONE DURING THE COURSE.

	PBL	**TRAD**
registered	63	14
female	23	4
male	40	10
participated	63 (61)	12 (14)
female	23	4
male	40 (38)	8 (10)
dropped	5	7
female	-	2
male	5	5
failed	1	-
female	1	-
male	-	-
passed	55	7
female	22	2
male	33	5

The average grades and grade distribution of the problem-based learners and the traditional learners of the course are represented in Table IV and Figure 1. (Grade 1- is the lowest accepted performance and 3 is the best grade.) There is no significant difference in the grades between the two groups, but the amount of traditional learners is too small for any conclusions. However, it is considerable that the population, which passed the course was about three times more than in the previous years. Thus also the weak students managed well in problem-based learning, even if the evaluation was quite strict and reaching the fundamental learning goals was checked. Another interesting phenomenon is the amount of the highest grades in the distribution. Many of those students did not only learn everything required for the best grade, but

became real experts in the area.

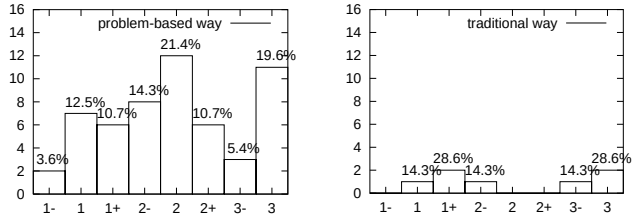

Fig. 1. The grade distribution of the problem-based and the traditional learners in the TFCS course.

TABLE IV

AVERAGE GRADE OF THE STUDENTS IN THE PROBLEM-BASED (PBL) AND THE TRADITIONAL (TRAD) APPROACHES.

	PBL	**TRAD**
all	1.97	2.00
female	2.08	3.00
male	1.85	1.60

There seems to be a recognizable difference between the grades of the female and the male students. We recall that only two female students took the course in the traditional way, and the difference in the traditional approach has no statistical meaning. However, there is a clear difference in the problem-based approach. It is possible that the problem-based method attracts more female students, as it has been reported in Linköping University [8]. Unfortunately, we do not have any reference about gender-grade correlation in other courses to make further conclusions.

Another interesting phenomenon was the influence of mathematical studies on the course performance. The students who passed the course in the traditional way had at least some (1/7) or a lot of (6/7) credit units in mathematics. In the PBL approach, the mathematical background did not affect so much on the performance. There was a noticeable correlation between the grades and amount of mathematical studies, but 17 students performed the course without any courses in mathematics and their grade distribution was still equal. So the PBL method helped also those students, who had poor starting points for the course.

B. Summary of the student feedback

After the course the students were asked to anonymously fill the official course evaluation form of the department. Only 36 students answered the query and the answers of the problem-based and the traditional learners were mixed. The students were also asked to write some free comments about the problem-based method. 26 students considered this question. Most of them were either very satisfied or quite satisfied with PBL and only three were unsatisfied (from whom one did not even participate PBL). Four of the students found the problem-based method heavy and total 14 students mentioned that the course would need more time. Six of those missed more traditional lectures. Only two students were unsatisfied with the learning diary. The conclusions about satisfaction with the course and PBL are represented in Figure 2.

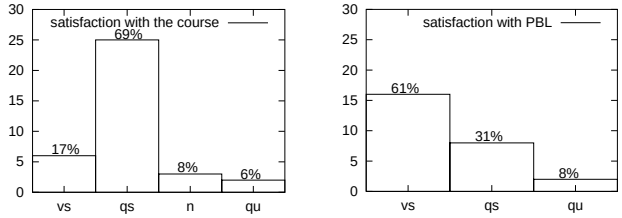

Fig. 2. The students' overall satisfaction with the course and satisfaction with the PBL method (vs=very satisfied, qs=quite satisfied, n=normal, qu=quite unsatisfied).

An interesting observation was the difference in satisfaction and overall feedback between the Finnish and the foreign students. 12 foreign students began the course, all of them in the problem-based way, but two dropped from the course and two changed to the traditional method. Those eight who remained in the problem-based method had very diverged opinions: they were either very excited or blamed the whole method. Partly this is due to the lack of English lectures, but we have had similar experiences with English courses, too. The amount of foreign students in our department is so remarkable that the reasons for dislike should be further studied.

C. Feedback in the learning diaries

Much more feedback was given in the students' learning diaries, and most of it was very positive, although the method was found quite heavy (c.f. narrative study by Naumanen [19]). The most delightful was the change in attitudes towards the theoretical topics. Here are some extracts from the learning diaries:

"Once again I have recognized that the theoretical issues can be approached in an interesting way, this week by the means of a game, pictures, and animations,"

"The fear I had felt for the topics of this course has disappeared. The topics were not after all so insuperably difficult!!"

"Now, afterwards, I feel as educated as after a normal course, but in another way. Maybe after a normal course you feel that when the exam is passed that's it. Now when we don't have any such culmination point the topics maybe remain in our memory better."

"Sometimes I felt that we are not learning anything in this way, but when I now look back everything seems surprisingly clear – i.e. they are learnt. Learning must have happened so discreetly that I haven't recognized it even myself."

"Now I can argue that even if the course has been one of the most disgusting in our curriculum, still my interest and motivation have not decreased even in the final meters."

"In the beginning of the course my learning goal was simply to pass the course. When I recognized the new learning method I leapt with joy, and when the course passed I realized that I really have possibility to learn something in this course. And so it happened that after all, all the Greek sounding concepts

like Turing machines, automata, regular languages, context-free languages, etc. have cleared at least partly for me, and becoming familiar with them and using them is no more as frightening as it was before."

V. Conclusions

In this paper we have reported our first experiment in teaching the theory of computability in the problem-based way. The results were successful, according to both quantitative (portion of dropouts, grade distribution, course feedback query) and qualitative (feedback and descriptions of personal learning experiences in the learning diaries) measures. We wish that this precedent encourages also others to try PBL in the education of theoretical computer science.

There are still interesting issues for further research. The problem-based course could be analyzed according to the *pedagogical stances* [20] the students tend to adopt in the course. Naumanen [19] has taken a step into this direction in her narrative study of the learning diaries the students produced during our course. Still there is a vast amount of valuable information for further research about students' experiences and personal development.

Another interesting research issue for the future is the influence of the cultural background on learning experiences. The amount of foreign students in our department is considerable: about one fifth of the students are from Eastern European countries and the contradiction between their previous learning experiences and the problem-based learning is evident, although the reasons are still unstudied.

References

[1] M. Bako, "Why we need to teach logic and how we can teach it?" 2002, http://www.ex.ac.uk/cimt/ijmtl/bakom.pdf.

[2] R. Page, "Software is discrete mathematics," *ACM SIGPLAN Notices*, vol. 38, no. 9, pp. 79–86, September 2003.

[3] M. Ben-Ari, "Constructivism in computer science education," *Journal of Computers in Mathematics and Science Teaching*, vol. 20, no. 1, pp. 45–73, 2001.

[4] D. Boud (ed.), *Problem-based learning in education for professions*. Sydney: Higher Education Research and Development Society of Australasia, 1985.

[5] H. Barrows, "A taxonomy of problem-based learning methods," *Medical Education*, no. 20, pp. 481–486, 1986.

[6] A. Ellis, L. Carswell, A. Bernat, D. Deveaux, P. Frison, V. Meisalo, J. Meyer, U. Nulden, J. Rugelj, and J. Tarhio, "Resources, tools, and techniques for problem based learning in computing," in *Working Group reports of the 3rd annual SIGCSE/SIGCUE ITiCSE conference on Integrating technology into computer science education*. ACM Press, 1998, pp. 41–56. [Online]. Available: http://doi.acm.org/10.1145/316572.358296

[7] H. Barrows and R. Tambly, *Problem-based learning: An approach to medical education*. New York: Springer, 1980.

[8] P. Lambrix and M. Kamkar, "Computer science as an integrated part of engineering education," in *Proceedings of the 6th annual conference on the teaching of computing and the 3rd annual conference on Integrating technology into computer science education.* ACM Press, 1998, pp. 153–156.

[9] R. Kirsch, "Teaching OLE automation: a problem-based learning approach," *SIGSCE Bulletin*, vol. 28, no. 1, pp. 68–72, 1996.

[10] T. Greening, J. Kay, and J. Kingston, "Results of a PBL trial in first year computer science," in *Proceedings of the Second Australasian Conference on Computer Science Education*. ACM, 1997, pp. 201–206.

[11] F. Dochy, M. Segers, P. Van den Bossche, and D. Gijbels, "Effects of problem-based learning: a meta-analysis," *Learning and Instruction*, no. 13, pp. 533–568, 2003.

[12] M. Albanese and S. Mitchell, "Problem-based learning: a review of literature on its outcomes and implementation issues," *Academic Medicine*, no. 68, pp. 52–81, 1993.

[13] D. Vernon and R. Blake, "Does problem-based learning work? A meta-analysis of evaluative research," *Academic Medicine*, no. 68, pp. 550–563, 1993.

[14] M. McCracken and R. Waters, "Why? When an otherwise successful intervention fails," in *Proceedings of the 4th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and technology in computer science education*. ACM Press, 1999, pp. 9–12.

[15] S. Lindblom-Ylänne and A. Nevgi, *Yliopisto- ja korkeakouluopettajan käsikirja*. Vantaa: WSOY, 2003.

[16] T. David, K. Burdett, and P. Rangachari, *Problem-based learning in medicine*. Worcester: Royal Society of Medicine Press Ltd., 1999.

[17] K. Hakkarainen, K. Lonka, and L. Lipponen, *Tutkiva oppiminen.* Porvoo: WSOY, 1999.

[18] E. Sutinen and J. Tarhio, "Teaching to identify problems in a creative way," in *Proceeding of FIE '01, 31st ASEE/IEEE Frontiers in Education Conference, IEEE*, 2001, pp. T1D8–13.

[19] M. Naumanen, "A narrative perspective to students' experiences in problem based learning in theory of computation – New deals with learning," in *Proceeding of Kolin Kolistelut – Koli Calling. 3rd Annual Finnish / Baltic Sea Conference on Computer Science Education*, 2003, pp. 66–74.

[20] M. Savin-Baden, *Problem-based learning in higher education: Untold stories*. Buckingham: The Society for Research into Higher Education & Open University Press, 2000.