Model of Learning Computational Thinking

Tauno Palts, *tauno.palts@ut.ee*
University of Tartu, Institute of Computer Science, Estonia

Margus Pedaste, *margus.pedaste@ut.ee*
University of Tartu, Centre for Educational Technology of the Institute of Education, Estonia

Abstract

There is a high demand for qualified information and communication technology (ICT) practitioners in the European labour market, but the problem at many universities is a high dropout rate among ICT students, especially during the first study year. The solution might be to focus more on improving students' computational thinking (CT) before starting university studies. Therefore, research is needed to find the best methods for learning CT already at comprehensive school level to raise the interest in and awareness of studying computer science. Doing so requires a clear understanding of CT and a model to improve it at comprehensive schools. Through the analysis of the articles found in EBSCO Discovery Search tool, this study gives an overview of the definition of CT and presents three models of CT. The models are analysed to find out their similarities and differences in order to gather together the core elements of CT and form a revised model of learning CT in comprehensive school ICT lessons or integrating CT in other subjects.

Keywords

Computational thinking, models, comprehensive school, learning

INTRODUCTION

There is a high demand for qualified ICT practitioners in the European labour market. Recent reports warn of decreasing interest among young people in studying science, technology, engineering, and mathematics (STEM) in many countries (OECD, 2008). Therefore, educational research is needed to introduce the principles of computer science already at comprehensive school level in order to raise the interest in as well as the awareness of studying computer science.

The 21st century calls for an overall redefinition of the forms of knowledge, skills and competences that are necessary for the advancement of our societies. Computations thinking (CT) is a fundamental skill for everyone, not just for computer scientists. Some authors even argue that we should add CT as every child's analytical ability, just like reading, writing, and arithmetic (Wing, 2006).

CT has a long history within computer science. CT is viewed as the way computer scientists use decomposition, recursion, and algorithms to tackle difficult problems (Hoffmann, 2009). Known in the 1950s and 1960s as "algorithmic thinking", Denning defined CT through input and output – CT is a mental orientation to formulating problems as conversions of some inputs to outputs and looking for algorithms to perform the conversions (Denning, 2009). Aho defined CT as the thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms (Aho, 2012).

Wing started a new wave of the usage of the term CT, which has become the most commonly used definition in recent academic publications. Wing defined CT as the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form which can be effectively carried out by an information processing agent. The information processing agent can be a computer, a machine or a human being (Wing, 2006).

Glass opposes that the traditional and still meaningful name for the CT idea is problem solving. According to him, there are several reasons for objecting to the word “computational” (Glass, 2006):

- It has not, for at least several decades, described the work of computers. Computers rarely compute but do manipulate information.
- The implication is that CT is a course computer scientists should teach. Computer science concepts can certainly be part of such a course, but problem solving is a universal activity, and many disciplines are capable of teaching it.
- Problem solving is a centuries old discipline. While computers are a powerful new tool for doing it, the underlying discipline should be focused on problems and solutions, not just on solution approaches.

CT is a part of problem solving, but in this article the main focus is on CT as it is acknowledged as a specific term describing one kind of thought process.

As CT has a long history within computer science, it is important to understand where CT situates in the discipline of computer science. Denning concluded that there are four core practices in computer science:

- programming,
- engineering of systems,
- modelling, and
- applying.

CT can be seen either as a style of thought that runs through the practices or as a fifth practice. Denning (2009) sees it as the ability to interpret the world as algorithmically controlled conversions of inputs to outputs.

In order to find the best way of learning CT already at comprehensive school level a model of learning CT should be created. The problem with the models of CT published in the academic journals so far is the lack of systematic approach of various dimensions to learn CT. A way of organizing these models about CT is to sort them into three categories: concepts, practices and perspective (Lye & Koh, 2014).

This study gives an overview of these three approaches of the models of learning CT, which included principles that were mentioned more than once. All the models of CT were analysed to find out their similarities and differences in order to gather together the core elements of CT and create a revised model of learning CT.

METHODS

This paper presents the models of CT found in the academic journals from three different perspectives. Research was done through a systematic search with the EBSCO Discovery Search tool as it includes a considerable number of educational academic journals referring to the core articles of CT. Research was done using the following steps:

1. The search term "computational thinking" was used to make a search in the EBSCO Discovery Search, which resulted in 84 articles published about CT in academic journals.
2. Abstracts of the 84 articles were read. All the articles which were not directly about computational thinking (e.g., in the field of medicine and journalism) and articles in other languages than English were left aside along with articles which could not be accessed. This left 55 articles to be analysed: 24 viewpoints which were articles that included opinions on the topic of CT; and 31 studies which included reviews, experiments, and case studies.

3. 55 available articles were read to sort out articles which included any kind of models of CT. In this stage 31 articles were selected for the final analysis of the study.

4. For a systematic analysis of the 31 articles a table was created (Appendix 1). It was constructed so as to give an overview of the articles with the following given characteristics for each article (Lye & Koh, 2014):

- author(s) of the article;
- setting of the article (e.g., kindergarten, K-12, higher education);
- research type (survey, experimental, case study);
- intervention (e.g., essence, pair programming, game strategy creation, modelling of mathematical concepts, story-based e-learning, etc.); and
- articles sorted by computational thinking approach:
  i. concepts – concepts that programmers use, e.g., variables, loops;
  ii. practices – problem-solving practices that occur in the process of programming; and
  iii. perspective – students' understandings of themselves, their relationships to others, and the technological world around them, e.g., expressing and questioning about the world of technology.

5. Articles which described models of CT from other studies were used as a starting point of the article. If needed, references in these articles were used to get to the root of the models introduced.

RESULTS AND DISCUSSION

The analysis of the 31 articles found by systematic search showed that 13 different models of CT were introduced. 10 of the articles concentrated on different aspects of CT: the senses involved in teaching CT (Katai et al., 2014); creating profiles of the students (Shell & Soh, 2013); emotional self-awareness, empathy alongside CT (Daily & Eugene, 2013); methods towards analytical skill building (Tsalapatas et al., 2012); convergence of systems biology and CT (Navlakha & Bar-Joseph, 2011); effects of task goals on learning computing concepts (Miller & Settle, 2011); dependency cycle in CT skills (Wolz et al., 2011); personal development trajectory within a sociocultural context (Marina U., 2011); solving a problem by reducing it to another problem (Kilpelainen, 2010); and diagnostics and rubrics for assessing learning across the computational science curriculum (Manson & Olsen, 2010). However, those 10 models are not directly included in the further analysis because none of these approaches to CT has been used in more than one article found by the systematic search.

However, there are three approaches to form a model of CT which were found in several articles: i) Interaction between a Human and Computer (evidence found in 3 articles); ii) Conceptual Model (evidence found in 2 articles); and iii) Engineering Design (evidence found in 3 articles). The next three paragraphs give an overview of those three models. After that a revised model of CT based on the three most widely used models is created and presented.

Model 1: Interaction between a Human and a Computer

Interaction between a Human and a Computer approach for a model of CT is derived from Cooper et al. (2010) looking at computational learning as an iterative and interactive process between the human (the K12 student in that case) and the computer (or, in a more theoretical construct, a model of computation) (Figure 1). This model includes the human cognitive process (capacity for abstraction and for problem formulation) and two strengths of the computer (ability to present complex data sets, often visually, and capacity for storing factual and relational knowledge) (Cooper et al., 2010).

Wing supports the approach with the definition of CT which claims that formulated problems should be carried out by an information processing agent, which can be a computer, machine or human being (Wing, 2006). Denning includes in the definition of CT the role of an input and an output, which can be used by computers and human beings (Denning, 2009).

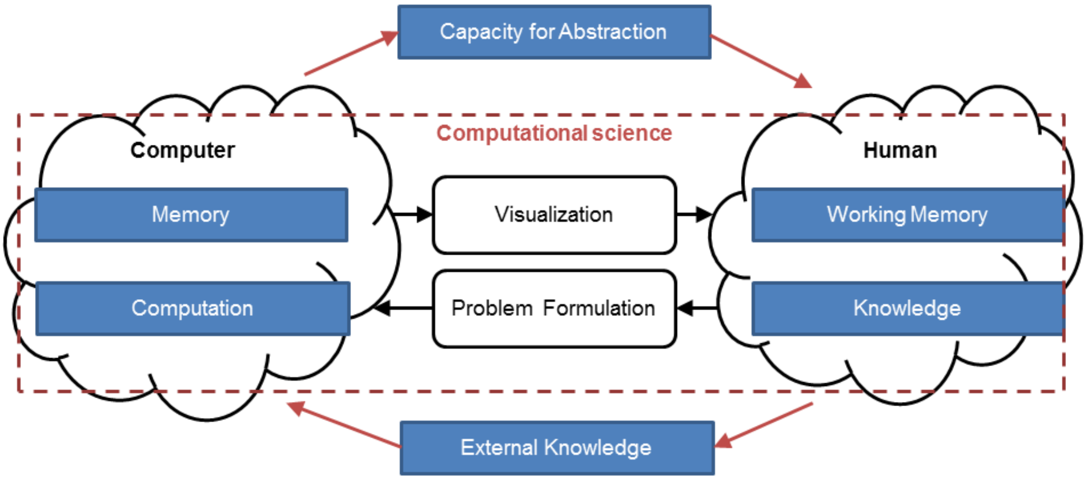

Figure 1: Model of Computational Learning (Cooper et al., 2010)

This model is rather similar to the definitions of CT presented in the introduction but does not include any information about the core concepts of CT (e.g., abstraction, automation) which are recognized as important parts of learning CT (Wing, 2006). Although the problem formulation is shortly mentioned in the model, not enough information is presented about the pedagogical side of learning CT. Interaction between a human and a computer should be presented in the revised model.

Model 2: Conceptual Model

In the Conceptual Model, CT is viewed as a link between discipline thinking and computing philosophy (Wenchong et al., 2014) including five core concepts of CT (Figure 2).

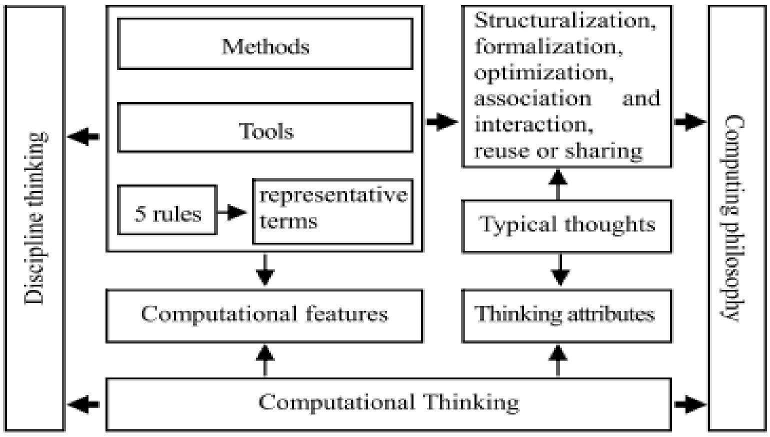

Figure 2: Conceptual Model of CT (Wenchong et al., 2014)

This model is based on the five typical thoughts of CT that form the core of CT:

1. Structuralization, which includes formatting and standardization. Examples of objects that are structural objects in the context of CT include data type, table, and file.

2. Formalization, which reflects ideas such as abstraction and visualization so as to be understood and universal. Examples of formalization include graph models, network protocols, and system formal specifications.

3. Optimization, which is reflected by concepts such as redundancy and complexity. Examples of optimization include lowering the time complexity by reducing the number of loops, from 1st normal form to 5th normal form of the database theory, choosing the data type appropriately.

4. Association and interaction including integrality, which is helpful for improving access speed, realizing automation and ensuring the validity of data. Examples of association and interaction: hyperlink in a web page embodies the association and interaction of files, media and location of information; object-oriented programming embodies the association and interaction of objects and data; transaction management of database theory embodies the association and interaction of events.

5. Reuse or sharing, which is one of the most important thoughts of computing science. It is beneficial to reduce the repetitive development of resources or to improve resource utilization. Examples of reuse or sharing: database technology, which makes data files separate themselves from the source programme and exist independently so that they can be applied to other programmes; resource sharing promoted by the wider coverage of Internet; the clipboard of Windows, the Cut, Copy and Paste operation of various software are all helpful for reusing the data or the files.

Compared to the previous model, this model includes some core elements of CT. Although there can be other ways of presenting the core concepts of CT, all other concepts (e.g., decomposition, pattern recognition (Grover & Pea, 2013)) can be recognized in various parts of the current model. All of the elements in the current model can be integrated into learning CT in various lessons, which means that instruments can be developed to analyse the students' understandings of the conceptual elements of CT.

This model lacks information about the steps of problem solving in learning CT: planning, prototyping, etc. All of these steps relate to CT in important ways that could be significant for education. The core concepts of the conceptual model of CT should be presented in the revised model.

Model 3: Engineering Design

The third way of approaching CT can be including the steps that are important when solving CT problems. Massachusetts Department of Education has created an Engineering Design process model which includes eight steps and is based on the cycle of problem solving (Figure 3) (Massachusetts Department of Education, 2006).

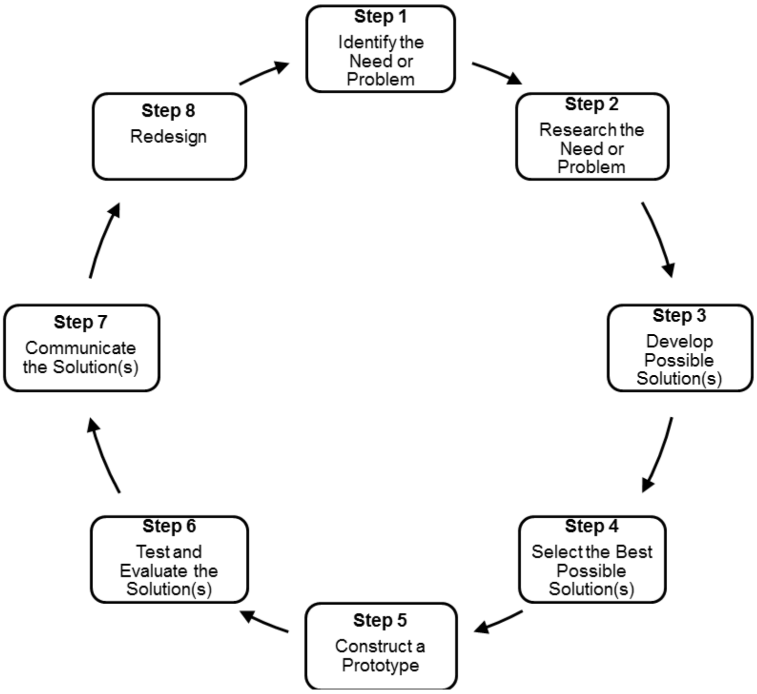

Figure 3: Steps of the Engineering Design Process (Massachusetts Department of Education, 2006)

This design process model consists of eight steps, which mostly follow each other in a linear order but can, in certain circumstances, be arranged in various types of order if needed. The steps of engineering design are as follows:

1. Identify the need or problem.
2. Research the need or problem: examine the current state of the issue and current solutions; explore other options via the Internet, library, interviews, etc.
3. Develop possible solution(s): brainstorm possible solution(s); draw on mathematics and science; articulate the possible solution(s) in two and three dimensions; refine the possible solution(s).
4. Select the best possible solution(s): determine which solution(s) best meet(s) the original need or solve(s) the original problem.
5. Construct a prototype: model the selected solution(s) in two and three dimensions.
6. Test and evaluate the solution(s) asking questions: Does it work? Does it meet the original design constraints?
7. Communicate the solution(s): make an engineering presentation that includes a discussion of how the solution(s) best meet(s) the initial need or the problem; discuss the societal impact and trade-offs of the solution(s).
8. Redesign: overhaul the solution(s) based on information gathered during the tests and presentation.

Steps 1–3 form the analytical, 4–5 the synthetical and 6–7 the evaluation part of the model. There are similar research oriented models created by Sengupta et al. (2013) and Lee et al. (2014), which include slightly different dimensions of CT, but those elements of CT are all included in the current model. Based on the current model, a simpler model has been created for youth (Lee et al., 2011). The current model has been effectively used for teaching CT in robotics (Bers et al., 2014).

The Engineering Design model has many specific steps for describing the process of learning CT, such as planning, prototyping, etc., which can be effectively carried out when learning CT in various comprehensive school lessons. The current model lacks information on the core concepts of CT, such as optimization, formalization, etc. The steps that this model represents can be used as essential parts of the revised model of learning CT.

Revised Model of Learning CT

All of the three applied CT approaches above (Interaction between a Human and a Computer, Conceptual Model, and Engineering Design) look at CT from different perspectives. When gathering together the core of all the models, a new model for learning CT can be created which includes the various dimensions of CT for learning CT at comprehensive school level (Figure 4).

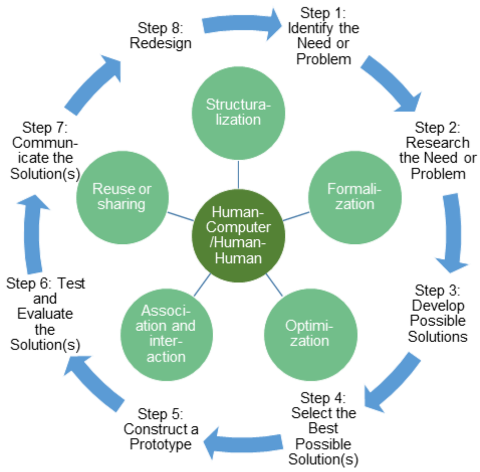

Figure 4. Revised Model of Learning CT

The revised model of learning CT includes three main components derived from previous models of CT:

1. Interaction between a Human and a Computer in the model of learning CT has a vital role and is presented in Figure 4 as the centre of the model.
2. Five core elements of CT (structuralization, formalization, optimization, association and interaction, reuse or sharing) are included in the model because those five components of CT can be taught in various lessons, in various key stages and involve the core elements of CT. In Figure 4 those five elements can be seen surrounding the centre of interaction of humans and computers. The five core elements can be rotated around the centre and used dynamically without any fixed order.
3. 8 steps are included in the model to go through all the steps that occur during the process of learning CT. The first step in the process of learning CT should be identifying the problem. The arrow pointing from step 1 to step 2 in Figure 4 indicates that after the problem is identified, research needs to be done, and after that, possible solutions are developed. The steps are following each other in a linear way and the various steps more or less include the core elements of CT. The circular arrows indicate that when one problem is solved (step 8), a new one can be started from the beginning (step 1).

The revised model of learning CT includes the definitions of CT, the core dimensions of CT, and the problem solving approach of learning CT. This model can be practically used to create scenarios for developing various CT skills by rotating the middle layer of CT concepts and then choosing one element from each layer. For example, in order to create a scenario for teaching structuralization in step 1 (Identify the need or problem), i) an interaction has to be chosen from the centre of the model (e.g., human-computer), and ii) the core concept (structuralization) has to be rotated towards step

1 (Identify the need or problem). As a result, for example, information about the need or the problem can be structured in a file with the valuable metadata and kept in a structured folder system. But in another scenario (e.g., step 5 – constructing a prototype) structuralization can be used to create a database to support the data collection in the solution.

CONCLUSION

In this study a model of learning CT was designed for creating scenarios learning CT in comprehensive school level ICT lessons or in other subjects.

Through the analysis of the articles found through EBSCO Discovery Search, this study gave an overview of the definition of CT and presented three models of CT. The core concepts from the models were integrated into a revised model for learning CT in comprehensive school ICT lessons or for integrating CT in other subjects:

- From the first model a principle of interaction between a human and a computer was added to the revised model.
- From the second model five core CT concepts (structuralization, formalization, optimization, association and interaction and reuse or sharing) were added to the revised model.
- From the third model engineering design steps (identify the need or problem; research the need or problem; develop possible solution(s); select the best possible solution(s); construct a prototype; test and evaluate the solution(s); communicate the solution(s); redesign) were added to the revised model.

The revised model also explains the relations between the various aspects mentioned above, which is an important part of the design of the learning process.

Further research is needed to investigate in more depth each of the five core CT concepts and to create instruments for evaluating the various concepts of CT. The relationships between the various elements and steps could be further researched. The model of learning CT is designed in such a way that it could be used in different subjects tackling computational problems. The activities for integrating the concepts of CT into the comprehensive school level classroom using various scenarios in lessons could be created and the effectiveness of the activities developing the awareness and knowledge of the CT principles could be researched. The revised model of learning CT is a theoretical one and we suggest creating scenarios of learning CT to test parts of it empirically in the future.

REFERENCES

Aho, A. V. (2012). Computation and computational thinking. Computer Journal, 55(7):832 – 835.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.

Cooper, S., Prez, L. C., and Rainey, D. (2010). Education k-12 computational learning. Communications of the ACM, 53(11):27.

Daily, S. B. and Eugene, W. (2013). Preparing the future stem workforce for diverse environments. Urban Education, 48(5):682 – 704.

Denning, P. J. (2009). The profession of it: Beyond computational thinking. Communications of the ACM, 52(6):28. Artificial Intelligence, Philosophy, and Cognitive Science, 21(2):301 – 322.

Glass, R. L. (2006). Call it problem solving, not computational thinking. *Communications of the ACM*, 49(9):13.

Grover, S. and Pea, R. (2013). Computational thinking in k12: A review of the state of the eld. *Educational Researcher*, 42(1):38 – 43.

Hoffmann, L. (2009). Qa: The upper limit. Communications of the ACM, 52(1):112.
Katai, Z., Toth, L., and Adorjani, A. K. (2014). Multisensory informatics education. Informatics in Education, 13(2):225.
Kilpelainen, P. (2010). Do all roads lead to rome? (or reductions for dummy travelers). Computer Science Education, 20(3):181 – 199.
Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., & . Werner, L. (2011). Computational thinking for youth in practice. ACM Inroads, 2(1), 32. doi:10.1145/1929887.1929902
Lee, T. Y., Mauriello, M. L., Ahn, J., and Bederson, B. B. (2014). Ctarcade: Computational thinking with games in school age children. International Journal of Child-Computer Interaction, 2:26 – 33.
Lye, S. Y., & Koh, J. L. (2014). Review: Review on teaching and learning of computational thinking through programming: What is next for K-12?. Computers In Human Behavior, 4151-61. doi:10.1016/j.chb.2014.09.012
Marina U., B. (2011). The tangiblek robotics program: Applied computational thinking for young children. Early Childhood Research and Practice, (2).
Manson, J. R. and Olsen, R. J. (2010). Diagnostics and rubrics for assessing learning across the computational science curriculum. Journal of Computational Science, 1(1):55.
Massachusetts Department of Education. (2006). Massachusetts science and technology/engineering curriculum framework. Retrieved from. Massachusetts Department of Education http://www.doe.mass.edu/frameworks/scitech/1006.pdf.
Miller, C. S. and Settle, A. (2011). When practice doesn't make perfect: E
ects of task goals on learning computing concepts. ACM Transactions on Computing Education, 11(4).
Navlakha, S. and Bar-Joseph, Z. (2011). Algorithms in nature: the convergence of systems biology and computational thinking. Molecular Systems Biology, 7:546.
OECD annual report. http://www.oecd.org/newsroom/40556222.pdf, 2008.
Repenning, A. (2012). Programming goes back to school. Communications of the ACM, 55(5):38.
Sengupta, P., Kinnebrew, J., Basu, S., Biswas, G., and Clark, D. (2013). Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework. Education and Information Technologies, 18(2):351.
Shell, D. F. and Soh, L.-K. (2013). Profiles of motivated self-regulation in college computer science courses: Differences in major versus required non-major courses. Journal of Science Education and Technology, 22(6):899 – 913.
Thiruvathukal, G. K. (2009). Computational thinking and doing. *Computing in Science and Engineering*, 11(6):4.
Tsalapatas, H., Heidmann, O., Alimisi, R., and Houstis, E. (2012). Game-based programming towards developing algorithmic thinking skills in primary education. Scientific Bulletin of the Petru Maior University of Targu Mures, 9(1):56 – 63.
Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3): 33 – 35.
Wenchong, S., Maohua, L., & Hendler, P. (2014). Computational Features of the Thinking and the Thinking Attributes of Computing: on Computational Thinking. Journal Of Software (1796217X), 9(10), 2507. doi:10.4304/jsw.9.10.2507-2513
Wolz, U., Stone, M., Pearson, K., Pulimood, S. M., and Switzer, M. (2011). Computational thinking and expository writing in the middle school. ACM Transactions on Computing Education, 11(2):1.

Appendix 1

No	Author	Setting	Research type	Intervention	Concept, practice, perspective		
					C	Pr	Pe
1	(Lye & Koh, 2014)	High s.	Survey	Conc. model of CT		X	
2	(Israel et al., 2014)	Teacher	Survey	Teaching CT		X	
3	(Berkaliev et al., 2014)	Uni.	Case study	CT in mathematics		X	
4	(Wenchong et al., 2014)	General	Survey	Conc. model of CT		X	
5	(Gynnild, 2014)	General	Survey	CT in journalism			X
6	(Katai et al., 2014)	General	Experiment	Multi-sensory approach to CT	X		
7	(Miller, 2014)	Uni.	Case study	Programming	X		
8	(Harlow & Leak, 2014)	Elem. s.	Case study	Programming	X	X	
9	(Bers et al., 2014)	Kinderg.	Case study	CT in robotics		X	
10	(Lee et al., 2014)	Prim. s.	Case study	Game design	X	X	
11	(Vakil, 2014)	Mid. s.	Experiment	Cognitive aspects in CT		X	
12	(Shell & Soh, 2013)	Uni.	Case study	Motivation of students		X	
13	(Libeskind-Hadas & Bush, 2013)	Uni.	Case study	CT in biology		X	
14	(Daily & Eugene, 2013)	Kinderg.	Experiment	Story based CT	X	X	
15	(Sengupta et al., 2013)	High s.	Experiment	Agent-based CT		X	
16	Kostadinov, 2013	Uni.	Survey	CT in math		X	
17	(Aiken et al., 2013)	Mid. s.	Case study	CT in physics		X	
18	(Rex & Ruben, 2013)	Uni.	Survey	Programming		X	
19	(Caballero et al., 2012)	Uni.	Survey	Programming	X		
20	(Tsalapatas et al., 2012)	Prim. s.	Experiment	CT in game creation	X		
21	(Navlakha & Bar-Joseph, 2011)	Uni.	Survey	CT in biology		X	
22	(Miller & Settle, 2011)	Uni.	Case study	Project-based learning		X	
23	(Wolz et al., 2011)	Mid. s.	Case study	Story based learning		X	
24	(Dodig-Crnkovic, 2011)	General	Survey	CT in computation			X
25	(Marina U., 2011)	Kinderg.	Survey	CT in robotics	X		
26	(Lee et al. 2011)	Elem. s.	Survey	Model of CT	X		
27	(Kilpelainen, 2010)	Uni.	Survey	Reduction in CT	X		
28	(Manson & Olsen, 2010)	Uni.	Survey	Rubrics in CT		X	
29	(Cooper et al, 2010)	Elem. s.	Survey	Model of CT	X		
30	(Molina & Mason, 2009)	Elem. s.	Experiment	CT in math	X		
31	(Ioannidou et al., 2009)	Mid. s.	Case study	Game creation in CT		X	

Table 1: Articles found from EBSCO search

Tauno Palts is an Assistant in Didactics of Informatics in the University of Tartu and has started doctoral studies in informatics in the Institute of Computer Science.

Acknowledgments

This research was supported by the European Union through the European Regional Development Fund. It is financed in the project 'Conceptual framework for increasing society's commitment in ICT: approaches in general and higher education for motivating ICT-related career choices and improving competences for applying and developing ICT'. In addition, this research was supported by European Social Fund's Doctoral Studies and Internationalisation Programme DoRa, which is carried out by Foundation Archimedes.