Computational Thinking and Acting: An Approach for Primary School Competency Development

Jan Pawlowski[12], Kati Clements[2], Dimitra Dimitrakopoulou[3], Martin Idzik[1], Mikko Muilu[2], Mihkel Pilv[4] and Sofoklis Sotiriou[3]

[1] Ruhr West University of Applied Sciences, Germany
[2] University of Jyväskylä, Finland
[3] Ellinogermaniki Agogi, Greece
[4] Miksike OU, Estonia
jan.pawlowski@hs-ruhrwest.de

Abstract. Computational Thinking has become an important concept for almost all age groups. It describes the purposeful utilization of Information and Communication Technologies for solving problems. The approach of Computational Thinking and Acting combines the computational thinking approach with physical computing, including physical activities to explore real-world problems and tangible outcomes. In this paper, we present a competency framework, design principles, and a sample learning scenario for Computational Thinking and Acting. The evaluation has shown that the approach can be integrated across subjects and is promising for both teachers and pupils.

Keywords: Computational Thinking, Primary School, Computational Thinking and Acting, Physical Computing

1 Introduction

Computational Thinking (CT) comprises of competencies necessary to utilize information and communication technologies to solve problems. In the current wave of Digital Transformation, CT competencies become relevant in many different subjects. In this paper, we extend the concept of CT showing how CT can be developed across subjects in primary schools using a physical computing approach.
The current trend of Digital Transformation leads to new requirements for organizations and individuals [1, 2]. New competencies are needed on all hierarchy levels and in different contexts: basic digital skills for using digital technologies are often expressed within the concept of Digital Literacy [3, 4]. More specific competencies will be required in the future work environments or the digital / digitized workplace [5, 6]. Individuals will need specific competencies for purposefully utilizing emerging or disruptive technologies, such as artificial intelligence, digital security or business analytics [7]. It is common to different frameworks for future competencies / skills [8, 9] that competencies on the use and utilization of information and communication technologies are of urgent importance. It is expected that those skills will improve different aspects, in particular future employability [10, 11].

Different pathways have been initiated to promote and improve Computational Thinking in different parts of education systems: on the policy level, the European Commission strongly promotes CT skills in the school system [12]. On the operational level, many countries have integrated CT into their curricula [13] and teacher education standards [14]. One key issue is that CT is no longer focused on the disciplines of Computer Science or Information systems: CT competencies will be needed in different subjects and (industry) domains [15, 16, 17, 18]. Based on these initial findings, we aim at developing an approach for Computational Thinking integrating latest trends and developments of Digital Transformation. The main research questions are:

1. Which competencies are necessary to utilize information and communication technologies for problem solving across subjects?
2. How to design activity-oriented learning and teaching scenarios to be integrated into curricula across subjects?

We focus on early education in primary schools, in particular for grades 3 to 6. In this paper, we describe the approach of Computational Thinking and Acting which includes a pedagogical framework and competencies as well as a proposal for cross-subject learning scenarios.

2 Background

2.1 Computational Thinking

Currently, there is no common definition of Computational Thinking (CT). The current understanding of the idea is based on the concept of Papert [19] and his idea of teaching computing to children. Previous works [20] have also focused on the ways computers solve problems, in particular algorithmic thinking [21]. The current use of the term was strongly influenced by Wing [22] who defined Computational Thinking as an "approach to solving problems, designing systems and understanding human behavior by drawing on the concepts fundamental to computer science". Cuny et al [23] define CT as "Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent". A shorter definition is given by Berland & Wilensky [24] as "the ability to think with the computer-as-tool". While there is no consensus on the definition, we use the following definition:

"The ability to understand and utilize information and communication technologies and their key concepts, methods and tools to purposefully solve real-world problems"

More important than the definition is the range of competencies included in this concept. As an example, the International Society for Technology in Education (ISTE) and Computer Science Teachers Association (CSTA) define key competencies as "Formulating problems in a way that enables us to use a computer and other tools to help solve them; Logically organizing and analyzing data; Representing data through abstractions such as models and simulations; Automating solutions through algorithmic thinking (a series of ordered steps); Identifying, analyzing, and implementing possible solutions with the goal of achieving the most; efficient and effective combination of steps and resources; Generalizing and transferring this problem solving process to a wide variety of problems." [14]. Grover & Pea [18] describe the following competencies as common for the purpose of curriculum development in the K-12 context: "Abstractions and pattern generalizations (including models and simulations); Systematic processing of information; Symbol systems and representations; Algorithmic notions of flow of control; Structured problem decomposition (modularizing); Iterative, recursive, and parallel thinking; Conditional logic; Efficiency and performance constraints; Debugging and systematic error detection"

To further compare approaches, we have analyzed different approaches. We identified 15 national and international approaches [12, 13, 14, 18] which represent the broad range of competencies which might be included in the concept Computational Thinking. Four key questions have emerged from the analysis. The first question is whether programming should be an essential learning outcome in CT. Some approaches integrate programming into CT, for example using visual programming languages [18]. At least, the long-term intention of most approaches is that children learn programming languages.

The second key question is the inclusion of emotional aspects, i.e., attitudes and dispositions. As an example, ISTE CSTA [14] propose five dispositions: "Confidence in dealing with complexity, persistence in working with difficult problems, tolerance for ambiguity, the ability to deal with open ended problems, the ability to communicate and work with others to achieve a common goal or solution". Generally, there are still many negative dispositions associated with Computer Science, e.g. as a nerd activity or "not for girls". [25, 26].

The third key question from the analysis regards specific (new) technologies. Almost no approach includes emerging technologies. However, as currently many technologies emerge which will dramatically change education and professional life as well as methods to solve problems (e.g. Machine Learning, Artificial Intelligence, 3D Printing), we would suggest to incorporate those into competency frameworks.

Finally, it is essential to connect real world problems and computer solutions. As Tissenbaum et al [27] criticize the "[...] initial focus on the concepts and processes of computing, leaving real-world applications for "later" runs the risk of making learners feel that computing is not important for them."

2.2 Physical Computing

Computer activities are often seen as passive with even strong negative effects on children's physical condition [28]. However, many CT approaches include the use of tangible, haptic devices such as robots [29, 30] to increase motivation, creativity, engagement and physical activity [31]. Przybylla & Romeike [32] describe Physical Computing as following: "Physical Computing covers the design and realization of interactive objects and installations and allows students to develop concrete, tangible products of the real world, which arise from the learners' imagination". A similar concept is Tangible Computing [33]. Common to those approaches is that computer activities should result in tangible (visual and haptic) experiences. Thus, most approaches focus on the physical output. Given the challenges that 1) computer activities are usually seated and 2) that the transfer from real-world problems to the computer is often neglected [27], it might be useful to extend the concept of Physical Computing. Physical Computing in an educational context should incorporate physical activities to 1) enable the problem identification in the real world and 2) to promote healthy activities and exercise. The “physical” perspective can be seen in different ways:

- **Physical Input:** The problem to be solved in a learning activity is observed and experienced in the real world. This means that children should physically go to the place where the problem can be seen.
- **Physical Transfer:** Every computer activity should be connected with physical real-life activities to avoid that computers are just perceived as a passive activity. The problem identified in the real world should then be transferred to a computer activity in which the solution is elaborated.
- **Physical Output:** Devices such as small robots are used to create haptic, tangible experiences as the outcome of the computer activity.

By extending the concept of physical computing, we address the above mentioned challenges and barriers to create more meaningful, active learning experiences.

3 Computational Thinking and Acting

3.1 Methodology

There is no common methodology for the development of competency frameworks / models or curricula due to the variety of disciplines involved in this field. Typically, competence models are built using deductive approaches (e.g. analysis of existing models) or expert consultations, often in professional communities such as AIS / ACM (e.g. [34]). Further methods include job advert analysis [35] or design based approaches [36]. We follow a Design Science Research method aiming at creating artefacts in a rigorous way [37]. We see the main outcomes of our research (Learning / Teaching Model incorporating physical computing paradigm, associated learning scenarios) as meta-artefacts [38] as well as a method [39] for further system design – resulting systems could be Human Resource Management systems as well as E-Learning courses as corresponding interventions and IT-artefacts.

As guiding theories, we use the competence-based view of the firm [40] which explains the relation of individual competencies to the performance of firms. However, as there is a time-lag of competence development (e.g. in school or university) and their usage (in a later employment). The second guiding theory is the Human Capital Theory which explains employability and competencies [41]. Especially the aspect of employability [42] needs to be extended regarding time and the specific impact of CT within the learning biographies of people. This aspect is not addressed in this paper but it is an overall goal of our research.

In our paper, we focus on the development of the meta-artefacts but also reflect our results towards contributing to theory building [43]. As a contribution towards theory building, we contribute towards the identification of new competencies and related activities as part of Computational Thinking. Regarding the practical contribution, we aim at providing useful learning scenarios for teachers to be used across curricula and subjects. The construction was based on a literature review and the main concept matrix [44]. Overall, we used a naturalistic multi-method evaluation [45]. The initial version of our artefacts was evaluated with 41 persons discussing the importance and target group fit of each category. The version described in the next chapter represents the result of this process.

3.2 Computational Thinking and Acting Framework

The key concept "Computational Thinking and Acting (COTA)" combines Computational Thinking and our extended understanding of Physical Computing. Computational Thinking and Acting (COTA) is defined as the ability to understand and utilize information and communication technologies and their key concepts, methods and tools by identifying and solving real world problems accompanied by physical activities and experiences. The definition describes the scope and specific characteristics of COTA. To further refine the concept, it is necessary to clarify competencies included. Our concept is reflected in the COTA curriculum. It covers seven main categories derived from the competency / curriculum analysis.

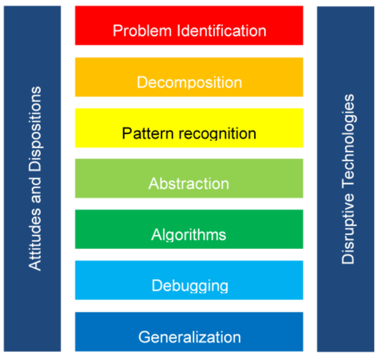

Fig. 1. COTA Currriculum Categories

In contrast to previous curricula (see 2.1), we include 1) *Attitudes and dispositions* containing emotional aspects towards computers and computing. This also contains aspects focusing on the physical perspective, i.e., computer activities should not be seen as solely seated indoor-activities. Furthermore, positive attitudes towards ICT as problem solving tools are intended. 2) *Disruptive Technologies* focus on specific concepts and technologies which (will) change practices in educational, private and professional life. A typical example is Artificial Intelligence which will change the ways problems are solved in business and leisure. Summarizing our approach, we see that Computational Thinking curricula will be highly dynamic and change frequently. However, the core curriculum needs to be mapped to and integrated into other subjects.

3.3 Pedagogical Approach and Learning Scenarios

As suggested by Kong [46], a curriculum needs to be extended describing clear activities and scenarios to be implemented in practice. There are few frameworks for pedagogical / didactical aspects of CT. Chande [47] proposes the following phases: production, recontextualization & problem solving, and reproduction & innovation. The model, however, is too prescriptive for our context. Kotsopoulos et al [48] propose four main types of learning activities: unplugged, tinkering, making and remixing. This model is well suited for physical activities but too strict for teaching CT across subjects. Further models apply well established didactical concepts such as Problem-Based Learning [49], Project-Based Learning [50] or Inquiry Based Learning [51].

Based on the review of existing models and our key concepts, we propose the following phases: 1) **Context setting:** In each scenario, the context needs to be clarified - in particular in cross-subject activities, the overall idea should be clarified. 2) **Problem exploration and identification:** A real world scenario should be explored by the learners - facilitated by a teacher, this should include a physical activity. 3) **Transfer and Elaboration:** In this phase, the problem is transferred to a computer activity. Depending on the type of the problem, different competencies are included. 4) **Production:** In this phase, the developed solution should be visualized / prototyped in the real world. If possible, physical outputs should be produced. 5) **Reflection:** In the final phase, the experience is reflected amongst participants. The phases are not necessarily sequential but can be parallel or repeated multiple times. Secondly, we derived principles for the creation and implementation of learning scenarios. Each learning scenario should be:

- **Real World Problem-Oriented:** Learners should experience and identify problems from the real world. They should be allowed to explore and create own ideas for problem solving / creating projects.
- **Learner-Centric**: Learners should be engaged and empowered to identify solution strategies. The teachers should act as facilitators to moderate the learning process (e.g. scaffolding)
- **Cross-subject:** Learning scenarios should not be restricted to computer science / ICT. Projects across subjects and disciplines should be created to provide richer learning experiences.
- **Physical:** each learning scenario should incorporate physical activities as input and output.
- **Transferable:** Each solution should be reflected to understand the transfer to other problems / subjects / domains.

Along those principles, initial learning scenarios have been designed for both teachers and students. To bring curricula into the classroom, it is necessary to provide concrete learning scenarios related to the competencies [46]. The following scenario is a short sample and outlines the main ideas of COTA.

Table 1. Sample Learning Scenario

Scenario Title	Decomposition
Main ideas and description	This learning scenario introduces algorithmic descriptions of problem solutions as well as decomposition. The problem in this case is how to estimate the rate of damaged / ill trees in a wide area. The scenario is related to the biology and maths curriculum.
Context	Primary school from grade 5

Curricula	Geography: Use of maps, scale Biology: Forests and trees, tree diseases, nature protection Mathematics: Surfaces, scales, units
Compe-tencies	Students can 1 Divide a problem into smaller sub-problems, 2) Use step-by-step instruction to describe a solution, 3) Use variables for calcula-tions, 4) Use conditions within loops
Peda-gogy	Explorative learning
Learning Activities	
LA1 Context	The teacher introduces problems which cannot be solved as a whole – examples are counting all animals in an area, sorting large amounts of things. Students get the task to go out to a close-by forest. The question is asked whether they can count all trees within one lesson. Additionally, tree diseases are introduced. What kind of diseases exist and how can they be observed (e.g. parasites such as birch moth, bark beetle; acid rain, …). This introduction needs to be modified depending on the geographical area.
LA2: Explora-tion	Students get the task to estimate the number of trees in a given forest. They g out to the area to get a visual impression of the problem. First, they calculate the surface of the area. This can be done using a map and estimating the total size. After this, students determine an adequate sam-ple size to decompose the problem (a realistic sample is 50m*50m). The students split up in groups and distribute tasks (one person for measuring the length / width of the area, one person to count all trees, some persons to find and count damaged trees). Students measure the step length to calculate how many steps equal 50m. The worksheet pro-vides step-by-step tasks to solve the problem.
LA3: Elabora-tion	Students go into the woods and find a place from where they can walk down the sample (50 by 50 metres). The corner points are marked (or a student stops there). A counter should count the number of steps (e.g. REPEAT walk_one_step UNTIL counter= number_of_steps) When the square is marked, students start counting the number of trees and agree on the number in case of differences. Afterwards students try to count damaged trees (sick by parasites, breakage etc). Finally, the students calculate the total number of trees and the rate of damaged trees.
LA4: Reflec-tion	Finally the whole algorithm in the four solution steps (calculate area, walk / mark area, count trees, calculate trees) should be written down. Students also can discuss in which other situations they could decom-pose to find a quicker solutions.
Roles	Students , Teacher / additional person for field trip
Tools and Ma-terials	Work sheet: Estimating trees Pen and paper (marking types of trees, conditions, amount Area map

The following figure shows sample tasks from the related guiding worksheets.

How many trees are in the forest below? How many are damaged?

Scale: 1:10000

How big ist he surface, use variables to calculate.
length_map = ____________
width_map = ____________
surface_map = ________ * ____________ = ___________

How long / side is the area in real?
length_real = __________
width_real = __________
surface_real = ________ * ____________ = ___________ m

Task: Go to the forest and mark a 50m * 50m square.
Step_length = __________ cm = _________ m
Number_of_steps = 50m / Step_length =

How many trees have you counted? How many are damaged
Counted_number_of_trees= ____________
Counted_damaged_trees = _____________

Now you can count the number of trees in the whole area. How many squares do you need for the full area?
Our_square * ____X_____ = real_square; X = _____________

Overall_number_of_trees = ___________ * Counted_number_of_trees =
Overall_number_of_damaged_tree = ___ * Counted_number_of_damaged_trees =
Great, you have helped us a lot to understand the situation in the forest!

Task: Summarize all steps you went through to measure the number of damaged tress. Use variables, conditions and loops, e.g.
IF step_counter == 100, THEN turn right by 90 degrees
IF tree = damaged, THEN counted_damaged_trees++1
REPEAT go_one_step UNTIL step_counter == Number_of_steps

Fig. 2. Sample work sheet

The scenario is just one example which was designed together with teacher trainers. It shows the key characteristics of the COTA approach:

- Problem identification in the real world: The physical input (forest excursion, determining damaged plants) and corresponding problem is discovered in the real world without computer usage.
- Transferring the problem towards a CT solution: Different steps and task guide students to the algorithmic description of the solution
- Using CT solutions, in particular pseudo code including object manipulation, conditions, loops.
- Receiving tangible results such as photos and documentation of the solution.

The proposed solution was tested with a small group of students and improved based on the students feedback. Then, a full evaluation of the approach was carried out.

4 Evaluation

To further understand teachers' views on computational thinking and physical learning scenarios, we interviewed individual experts and focus groups in Estonia (E), Finland (F), Germany (DE) and Greece (GR). We involved 41 individuals: teachers (n2=30), headmasters (n=5), university lecturers (n=4), teacher training specialist (n=1) and educational technologist (n=1). Out of the teachers, 21 worked on primary level, 4 on secondary and 7 on both.

The first part of the evaluation looked at physical computing as a pedagogical approach. The following solutions were discussed by the interviewees.

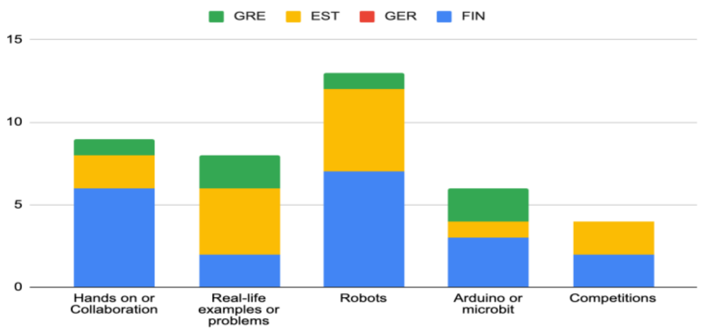

Fig. 3. Physical Computing elements

Real-life problems: Regarding the physical aspects, the most discussed issue was the transfer of problems from the real world to computer solutions which was seen as the key to success (F8, F9, E1, E2, E3, E7, GR2, GR9). "If students solve real-life problems, they are more interested, motivated and focused on finding solutions." (Estonian primary school teacher, age 63)

Robots: The use of robots is also seen positively as it is already current practice in many schools, especially in Finland, Greece and Estonia (F1, F2, F3, F4, F5, F8, F9, E3, E4, E6, E9, E10, GR4). "Students understand deeply physics and mathematics due to robotics lessons." (Greek primary school teacher, age 43)

Microcontrollers: Arduino and MicroBit controllers were popular in Finland and Greece (F3, F4, F5, E10, GR1, GR5). "I used a couple of years ago the breadboard of Arduino and for myself it was a fruitful experience because I got to know more about physics. In addition, I realized that students were motivated, excited and understood more easily due to physical computing." (Greek primary school teacher, age 36)

Collaborative and Hands-on activities: Collaborative and "hands-on" activities were described by most interviewees as liked and most engaging (F1, F2, F3, F6, F8, E3). Physical computing has been used in making art and in games. "Hands on group work seems to offer the easiest way to learn computational thinking." (Finnish primary school teacher, age 43)

Competitions: Different kinds of playful competitions were also mentioned several times (F2, F9, E4, E10). Beaty contests or dance competitions of the lego-robots were used. "We used to do only the 'Hour of code', but there's hardly anything that really grasps and motivates the students. Then again, giving them a task (sumo wrestling lego robots) that includes programming, they can be motivated. Of course they concentrate on the looks of the robot, the flag it carries and everything else 'unimportant' that doesn't have anything to do with the ICT goal. But when they are motivated, they will do work at home and they will compare their work with other students. After the first sumo championship games, the students wanted to have a couple of weeks to modify their robots and their programming and have a rematch. You don't hear that when the kids are just coding with the computer." (Finnish primary school teacher, age 46)

Overall, the idea of combining physical and computer activities was seen very positively to also change attitudes towards computer work.

The second part of the evaluation looked at the **Competency Framework**. Most teachers recognised potential to achieve the competencies within the COTA framework (Fig. 1). More specifically, The German teachers ranked different competences differently, the most important were: 1) Problem solving, 2) Algorithmic thinking, 3) Digital / media literacy, 4) Utilizing programs for problem solving, 5) ICT as tools for learning. The view of the experts was rather different on the competencies of programming in a specific language. While one expert found this a regular competency "(Scratch) Programming is regularly taught from grade 2-3" (GR1), another expert clearly said that "I don't think that in primary school students learn programming." (GR7) The category of debugging was also controversial. As a consequence, we will continue to revise the curriculum in further trials with school children and teachers. Overall, there was consensus that CT is necessary in this age group or as one expert said: "It is the best age to learn the basics". Overall, the curriculum was perceived very positively as a starting point for profile building for different requirements (in each country).

As a final aspect, we asked for **support needs** towards enabling teaching of computational thinking and physical computing. Key findings are quite obvious on resources (time, funding, learning resources, competences, training) but also the teachers hope they could have hands-on playing and testing time before introducing physical computing activities in their teaching. There are some differences between the European countries, but they all agree on the importance of increasing teachers' competences and providing learning resources ready, easy-to-use, easy-to-find, age appropriately available. Finnish teachers are not currently worried about schools' infrastructure. Their need is more towards teachers' time. Chart X shows the country differences in teachers' needs to see whether our approach of providing Open Educational Resources (OER) and Practices (OEP) is feasible. The following figure shows the teachers' requests.

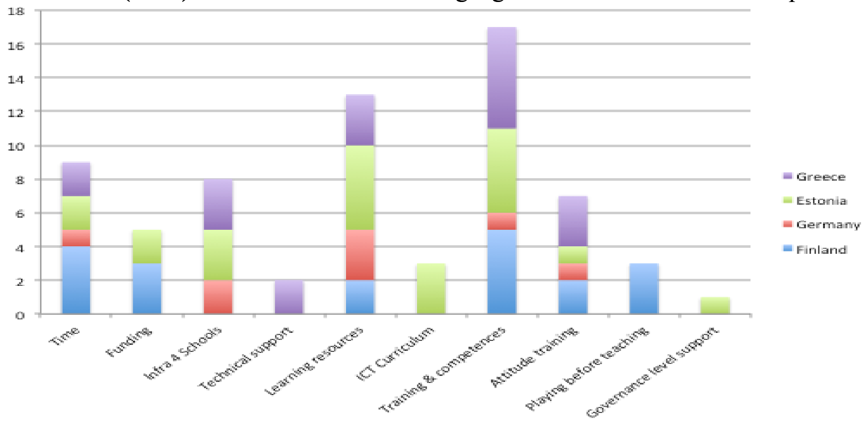

Fig. 4. Support needs of teachers

Needs for competences through training, guidance and support: Most popular answer in this interview was that teachers agreed that they need guidance, training and support to increase their competences. In Finland, teachers' digital peer-mentoring projects have been well received and have provided a systematic way for teachers to learn from each other. Teachers also need "Clear and simple goals to achieve to avoid overwhelming them from the start."(Finnish Maths teacher, Age 42). A lot of the answers were focusing on teachers' attitudes and motivation rather than technical skills in using robots or computers. It was emphasised that the support mechanisms should make teachers excited (German Primary Schoo teacher, Age 32) and get the teachers motivated (Finnish Principal, age 44 and Greek Primary School teacher 39). Teachers should be thought to have an open mind. (Estonian Seconary level teacher, age 45)

Needs for easy-to-use ready-made learning resources: Second most popular need focused on learning resources in the forms of online resources as well as conventional books and worksheets (German Headmaster, Age 50). The resources should be designed to be age-appropriate and some mentioned playful, gamified elements. "Learning materials should be age appropriate. Playful materials for primary school and more complex materials for the older kids." (Estonian IT Specialist 51, Estonian secondary level teacher, 60)

The following list shows the requirements which will be applied in the next steps when designing further learning materials which should be: easy to use; easy to find; age appropriate; simple goals clearly defined; structured; offline worksheets for those who need them; playful, gamified materials.

Overall, the results of the evaluation have shown that the approach is useful for both teachers and experts. As a next step, the COTA approach will be tried out in experiments in schools with teachers and pupils.

5 Conclusion

In this paper, we have outlined the concept of Computational Thinking and Acting (COTA) which combines the concepts of Computational Thinking and Physical Computing. The concept aims at providing new learning and teaching experiences with a focus on contributing to competency development in the field of Computer Science as well as future employability. We developed meta-artefacts as well as a method to implement the competency framework, i.e., design principles and learning scenarios. The initial evaluation has shown that our competency framework covers the most important aspects of computer science and is feasible for schools and cross-subject teaching.

As the next steps, learning scenarios will be evaluated in different schools and subjects. We aim at better understanding the impact of CT competencies and improve the design guidelines and learning scenarios. In the long-term, it will be of high importance to understand the relation of CT and employability over time.

6 Acknowledgement

This research has been co-funded by the European Commission within Erasmus+ programme, project COTA, grant no. 2019-FI01-KA203-060877.

7 References

1. Hoberg, P., Krcmar, H., Oswald, G., & Welz, B. (2017). Skills for digital transformation
2. Sousa, M. J., & Rocha, Á. (2019). Digital learning: Developing skills for digital transformation of organizations. Future Generation Computer Systems, 91, 327-334.
3. Buckingham, D. (2010). Defining digital literacy. In Medienbildung in neuen Kulturräumen (pp. 59-71). VS Verlag für Sozialwissenschaften.
4. Eshet, Y. (2004). Digital literacy: A conceptual framework for survival skills in the digital era. Journal of educational multimedia and hypermedia, 13(1), 93-106.
5. White, M. (2012). Digital workplaces: Vision and reality. Business information review, 29(4), 205-214.
6. Dery, K., Sebastian, I. M., & van der Meulen, N. (2017). The Digital Workplace is Key to Digital Innovation. MIS Quarterly Executive, 16(2).
7. Andriole, S. J. (2018). Skills and Competencies for Digital Transformation. IT Professional, 20(6), 78-81.
8. Van Laar, E., Van Deursen, A. J., Van Dijk, J. A., & De Haan, J. (2017). The relation between 21st-century skills and digital skills: A systematic literature review. Computers in human behavior, 72, 577-588.
9. Ehlers, U. D., & Kellermann, S. A. (2019). Future Skills: The future of learning and higher education, 2-69. Karlsruhe.
10. Frey, C. B., & Osborne, M. A. (2017). The future of employment: How susceptible are jobs to computerisation?. Technological forecasting and social change, 114, 254-280.
11. Southama, D. C., Rohlb, A. L., & Balserc, T. C. (2017). STEM GRADUATES AS DIGITAL CREATORS: COMPUTATIONAL THINKING FOR TWENTY-FIRST

CENTURY EMPLOYABILITY. In The 23rd Australian Conference for Science and Mathematics Education, 110.
12. Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., & Punie, Y. (2016). Developing computational thinking in compulsory education. European Commission, JRC Science for Policy Report.
13. Balanskat, A., & Engelhardt, K. (2014). Computing our future: Computer programming and coding-Priorities, school curricula and initiatives across Europe. European Schoolnet.
14. ISTE CSTA (2011). Operational Definition of Computational Thinking for K–12 Education. National Science Foundation.
15. Chung, W., Yang, S., Fox, E. A., Sheetz, S. D., Chung, W., Yang, S., & Fox, E. A. (2010). Integrating Computational Thinking into Information Systems and Other Curricula. In Proceedings of the SIGED Conference.
16. Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. Journal of Computing Sciences in Colleges, 25(5), 66-71.
17. Czerkawski, B. C., & Lyman, E. W. (2015). Exploring issues about computational thinking in higher education. TechTrends, 59(2), 57-65.
18. Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. Educational researcher, 42(1), 38-43.
19. Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc..
20. Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. In Proceedings of the 16th Koli Calling International Conference on Computing Education Research, 120-129. ACM.
21. Knuth, D. E. (1985). Algorithmic thinking and mathematical thinking. The American Mathematical Monthly, 92(3), 170-181.
22. Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35.
23. Cuny, J., Snyder, L., & Wing, J. M. (2010). Demystifying computational thinking for non-computer scientists. Unpublished manuscript
24. Berland, M., & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology*, *24*(5), 628-647.
25. Weisgram, E. S., & Bigler, R. S. (2006). The Role of Attitudes and Intervention in High School Girls' Interest in Computer Science. Journal of Women and Minorities in Science and Engineering, 12(4).
26. WISE (2015). Women in Science,Technology, Engineering and Mathematics: The Talent Pipeline from Classroom to Boardroom. https://wise.statementcms.com/uploads/wise/files/WISE_UK_Statistics_2014.pdf
27. Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From computational thinking to computational action. Communications of the ACM, 62(3), 34-36.
28. Carson, V., Hunter, S., Kuzik, N., G. et al (2016). Systematic review of sedentary behaviour and health indicators in school-aged children and youth: an update. Applied Physiology, Nutrition, and Metabolism, 41(6), S240-S265.
29. Blikstein, P. (2013). Gears of our childhood: constructionist toolkits, robotics, and physical computing, past and future. In Proceedings of the 12th international conference on interaction design and children (pp. 173-182). ACM.
30. Rees, A., García-Peñalvo, F. J., Jormanainen, I., Tuul, M., & Reimann, D. (2016). TACCLE3, An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers.
31. Hodges, S., Sentance, S., Finney, J., & Ball, T. (2020). Physical computing: A key element of modern computer science education. Computer, 53(4), 20-30.

32. Przybylla, M., & Romeike, R. (2014). Physical Computing and Its Scope--Towards a Constructionist Computer Science Curriculum with Physical Computing. Informatics in Education, 13(2), 241-254.
33. Horn, M., & Bers, M. U. (2019). Tangible computing. The Cambridge handbook of computing education research, 1, 663-678.
34. Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K. M., Nunamaker Jr, J. F., Sipior, J. C., & De Vreede, G. J. (2010). Curriculum guidelines for undergraduate degree programs in information systems. ACM/AIS task force.
35. Todd, P. A., McKeen, J. D., & Gallupe, R. B. (1995). The Evolution of IS Job Skills: A Content Analysis of IS Job Advertisements From 1970 to 1990. MIS Quarterly, 19(1).
36. Wesselink, R., Biemans, H., Gulikers, J., & Mulder, M. (2017). Models and principles for designing competence-based curricula, teaching, learning and assessment. In Competence-based Vocational and Professional Education (pp. 533-553). Springer, Cham
37. Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of management information systems, 24(3), 45-77.
38. Iivari, J. (2015). Distinguishing and contrasting two strategies for design science research. European Journal of Information Systems, 24(1), 107-115.
39. Offermann, P., Blom, S., Schönherr, M., & Bub, U. (2010). Artifact types in information systems design science–a literature review. In International Conference on Design Science Research in Information Systems (pp. 77-92). Springer, Berlin, Heidelberg.
40. Freiling, J., Gersch, M., & Goeke, C. (2008). On the path towards a competence-based theory of the firm. Organization Studies, 29(8-9), 1143-1164.
41. Berntson, E., Sverke, M., & Marklund, S. (2006). Predicting perceived employability: human capital or labour market opportunities? Economic and Industrial Democracy, 27(2).
42. Cai, Y. (2013). Graduate employability: A conceptual framework for understanding employers' perceptions. Higher Education, 65(4), 457-469.
43. Baskerville, R., Baiyere, A., Gregor, S., Hevner, A., & Rossi, M. (2018). Design science research contributions: finding a balance between artifact and theory. Journal of the Association for Information Systems, 19(5), 3.
44. Webster, J., & Watson, R. T. (2002). Analyzing the past to prepare for the future: Writing a literature review. MIS Quarterly, 26(2).
45. Venable, J., Pries-Heje, J., & Baskerville, R. (2016). FEDS: a framework for evaluation in design science research. European journal of information systems, 25(1), 77-89.
46. Kong, S. C. (2016). A framework of curriculum design for computational thinking development in K-12 education. Journal of Computers in Education, 3(4), 377-394.
47. Chande, S. (2015). A Conceptual Framework for Computational Thinking as a Pedagogical Device. International Journal of Innovative Research in Computer and Communication Engineering, 3(11).
48. Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A pedagogical framework for computational thinking. Digital Experiences in Mathematics Education, 3(2), 154-171.
49. Nuutila, E., Törmä, S., & Malmi, L. (2005). PBL and computer programming—the seven steps method with adaptations. Computer Science Education, 15(2), 123-142.
50. Sierra, A. J., Ariza, T., & Fernandez, F. J. (2013). PBL in programming subjects at engineering. Bulletin of the IEEE Technical Committee on Learning Technology, 15(2).
51. Psycharis, S., & Kotzampasaki, E. (2017). A didactic scenario for implementation of computational thinking using inquiry game learning. In Proceedings of the 2017 International Conference on Education and E-Learning (pp. 26-29). ACM.