Kong, S.C., Andone, D., Biswas, G., Hoppe, H.U., Hsu, T.C., Huang, R.H., Kuo, B.C., Li, K.Y., Looi, C.K., Milrad, M., Sheldon, J., Shih, J.L., Sin, K.F., Song, K.S., & Vahrenhold, J. (Eds.). (2019). *Proceedings of International Conference on Computational Thinking Education 2019.* Hong Kong: The Education University of Hong Kong.

Employing Computational Thinking in General Teacher Education

Stefan SEEGERER[1*], Ralf ROMEIKE[2*]
[1] Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
[2] Freie Universität Berlin, Germany
stefan.seegerer@fau.de, ralf.romeike@fu-berlin.de

ABSTRACT

The current political discussion about the digital transformation in Germany's educational context is primarily concerned with the use of digital media in schools. However, all disciplines and their related school subjects are significantly affected by digitalization – as can be seen e.g. with the effects of simulation or data analysis. This results in new topics, methods or strategies that schools must also deal with in the future. In consequence, teachers of any subject require Computational Thinking competencies and Computer Science knowledge, not only for the efficient and effective use of digital technology but also to understand and apply the new topics, methods, and approaches. In this paper, the design and implementation of a new course for teacher education in Germany is presented. With a theme revolving around digital transformation, this course aims at preparing pre-service teachers for teaching in the 21st century. Design principles and content selection are based on an analysis of similar courses and requirements arising from digitalization and its effect on the disciplines. First results show that students have gained a clearer understanding of how digitalization influences their subjects and teaching in general. Additionally, they report feeling more confident in employing aspects of digital education.

KEYWORDS

digitalization, teacher education, computational thinking

1. INTRODUCTION

Digitalization facilitates storing, processing and searching massive amounts of data. This accelerates fundamental changes affecting our daily lives. In addition, all disciplines are significantly affected: Digitization leads to far-reaching changes, for example in the collection of data (scope, quantity, and quality) (Grillenberger & Romeike, 2014) or their mostly automatic processing (simulation, collection, and evaluation of large amounts of data) (Hey, Tansley, & Tolle, 2009). With simulation or data analysis often being considered the third and fourth pillars of science (e.g. Riedel et al., 2008), new approaches to knowledge generation evolved. This results in new topics, methods and strategies for all disciplines (e.g. Hegedus et al., 2017).

These new topics, methods and strategies are becoming increasingly relevant for schools, as well. The application of computing technology across subjects requires an understanding of "how, when and where computers and other digital tools can help us solve problems" (Barr, Harrison & Conery, 2011). To embed this application, teachers require Computer Science competencies and knowledge, not only for the efficient and effective use of digital technology, but also to understand and apply the new topics, methods, and approaches while teaching.

In this paper, the questions to be answered are: which design principles can guide the creation of a course aiming at conveying necessary competences, and what content is relevant for teachers of all subjects. Therefore, various individual research results are outlined and intertwined. Building upon these results, five blended-learning modules for pre-service teachers in Germany are presented. The first iteration is then evaluated with regards to the design principles.

2. COURSE BACKGROUND AND RELATED WORK

Digitalization is having an increasing impact on K12 education. Recently, in political contexts, this has been frequently summarized under the term "digital education". While this term is often associated with the use of technology, there are also approaches that focus more on Computational Thinking. CT describes ways of tackling a problem like a computer scientist would (Wing, 2006). Even though there is no agreed-upon definition, there is a mutual agreement that it includes a set of skills to think about problems and their solutions (Kalelioglu, Gülbahar, & Kukul, 2016). It is about applying skills, such as abstraction or decomposition, so that the solution can be effectively carried out by a computer. Initial approaches to incorporate CT into K12 teaching relate primarily to science education, e.g. by learning science through simulation and modeling (Basu et al, 2013). But the approaches and achievements of the digital humanities show that teaching in other subjects could be heavily influenced by digitalization as well.

Consequently, a number of sources discuss the embedding of CT across the curriculum (cf. Barr & Stephenson, 2011, Kale et al., 2018). Examples include curve fitting or doing a linguistic analysis of sentences (e.g. Barr & Stephenson, 2011). The important role "computer (and possibly its abstraction) can play in enhancing the learning process and improving achievement of K–12 students in STEM and other courses" (Cooper, Pérez & Rainey, 2010) is emphasized in the model of Computational Learning (CL). The model combines theories of learning and the computer's ability to handle complexity and visualize results appropriately to improve understanding as well as learning. While this concept includes the usage of computers, e.g. for simulations and modeling, the authors stress that the model explicitly excludes non-cognitive uses of technology such as clickers or blogs.

Computer science skills and knowledge support teachers in engaging their students in CT and CL. While universities started to offer CS courses for a lot of non-CS student groups, there are still many pre-service teacher trainings focusing mainly on improving information and communication technology (ICT) skills (e.g. Goktas, Yildirim, & Yildirim, 2009). Despite the importance of Computational Thinking in all disciplines and throughout education, existing approaches explicitly embedding CT-related competencies in non-CS teacher training are still rare. Existing CS courses for non-CS teachers tend to focus, for example, on everyday phenomena that they analyze and illuminate from a computer science point of view (Müller, Frommer, & Humbert, 2013), on algorithmic concepts (Yadav et al., 2011), or on emphasizing Computational Thinking in the context of Information & Media Literacy (Dengel & Heuer, 2018).

3. ANALYSIS OF COMPUTATIONAL THINKING SKILLS FOR GENERAL TEACHER EDUCATION

In order to develop a CT curriculum in general teacher training, we have drawn on several sources. This section presents how these individual research results were merged and used as the basis for the course. Building upon the results, this section outlines how digitalization affects subjects and illustrates design principles and selection of content.

3.1. How Digitalization Affects Subjects

While new topics or methods, such as simulation, can be found in the context of science teacher training (Smetana, & Bell, 2012), the discussion of these new methods and topics for other subjects, such as the humanities, is fairly new. In order to understand which CT competencies future teachers need, looking at how digitalization affects the regarding subjects and related disciplines delivers helpful insights. These effects result from a survey among educational researchers for subject-matter teaching and learning and have also been discussed in working groups with the participants (Seegerer & Romeike, 2018c) In the following, we will highlight effects of digitalization in the disciplines and discuss how the effects affect school education.

3.1.1. Tools

Software or hardware tools supporting cognitive processes are an integral part of daily work, e.g. computer algebra systems in mathematics or geoinformation systems in geography. This offers possibilities for teachers as well. For example, the use of specialized databases (e.g. regarding life on Earth) offers great opportunities for biology teaching.

3.1.2. Methods and Ways of Thinking

The effects of digitalization in the disciplines are not limited to the mere use of digital media or tools, but fundamentally change cognitive processes (Berman et al., 2018). The methods used in the disciplines are particularly important for determining Computer Thinking competencies. Methods, like modeling, simulation, or data analysis are becoming increasingly relevant across disciplines (e.g. Ananiadou & McNaught, 2006). These methods also allow for a more student-centric access to topics in school. An example from geography is a lesson in which students use GPS data tracked on their daily ways to school. The resulting data is then evaluated collaboratively regarding the meaningfulness of the data.

3.1.3. Topics

A review of publications in various fields of research shows that not only methods are changing. In addition, the digital transformation also leads to “new” topics (e.g. Bharadwaj, et al, 2013) in the disciplines. The survey with experts in subject-matter teaching and learning shows that this also affects topics taught in school. We refer to a new topic when phenomena or artifacts of the digital world can be explored and explained from concepts, ideas or foundations of the subject. Economics, for example, discuss digital business models in contrast to traditional ones. As students are confronted with the implications of these business models when interacting with digital services, such topics also become relevant for schools.

This has consequences for teacher training. Since most teachers have not received a comprehensive general education in CS, they need a foundation of CS knowledge and CT skills. For example, discussing digital business models requires teachers to have a basic understanding of underlying concepts, such as knowing how to attribute meaning to data. However, it is necessary to address the implications of the effects of digitalization on the subjects: Not only do they need to apply digital tools and computational methods in the classroom, they also need to teach new ways of thinking and problem solving and convey new topics. Although courses such as bioinformatics are becoming more and more popular, they have not yet found their way into the course programs of teachers.

3.2. Design Principles

When CS courses for non-majors are developed, many challenges need to be faced. The students do not necessarily take the course because of an interest in CS, but may take it as a preparation for teaching in a digitalized world. The question is how a course should be designed to give prospective teachers the best possible experience. Based on the effects of the digital transformation on the disciplines, we decided to always embed CS in a theme revolving around digitalization: Topics are discussed in the context of digital transformation to show relevance for the students. Building on the survey results, the constraint of this being the only CT course for nearly all students, and the need to adequately prepare pre-service teachers for a digital age, several key design principles emerged:

3.2.1. Discuss Digital Transformation on a CS Basis

As described in the previous section, Computer science knowledge and skills are necessary to understand the digital transformation and advances in the disciplines. In addition, the digital transformation also leads to new issues becoming relevant for being discussed in class (cf. Brinda & Diethelm, 2017). Due to their history of education, many prospective teachers still do not have a proper foundation in Computer Science. A course must, therefore, be based on fundamental ideas of computer science that underlie the digital transformation and also highlight impacts on society. In order to be a good fit for prospective teachers of all disciplines, a corresponding course must also take the different levels of prior knowledge into account.

3.2.2. Show Relevance to Students of All Disciplines

German teacher education has a strong focus on the individual subjects. Typically, teachers study two subjects in-depth, which they will later teach. As seen in the section before, digitalization leads to new topics, methods and tools in their related disciplines. Thus, it also affects subject teaching. Although the course is generally be designed for students of all subjects and school types, transferability of the topics into the subjects must be ensured. Thus, cross-references to other disciplines should be emphasized repeatedly. This is addressed via different exercises, ideas and examples provided throughout the course. In some cases, students are required to research or discuss additional examples transfer to their subjects. In other cases, we provided a list of ideas they can elaborate on.

3.2.3. Profit from CT Inside and Outside the Classroom

CT skills, such as solving a problem using a simulation, help students in their disciplines. As prospective teachers, these students are also in the position to teach CT. By integrating CT and CL into the course, pre-service teachers learn how to profit from computing in their disciplines, as well as inside and outside the classroom. To this end, the course needs corresponding tasks including programming, as it exposes students to CT (Lye, & Koh, 2014). While hands-on activities enable teachers to learn CT, the connection to teaching should be highlighted whenever possible.

3.2.4. Show the Importance of Skills Like Collaboration or Creativity

Many experts believe that traditional topics will be less important in the future. Instead, teachers should be enabled to promote creativity, collaboration and critical thinking in the context of digital education (e.g. Bellanca, 2010). These approaches are also part of CT models, such as the CAS Model (Computing at School, 2014). This mindset should also be reflected in course design: Exercises allow students to develop their own ideas or to create personal artifacts. Many tasks are designed to encourage collaboration among students (e.g. via pair programming or collaborative writing). The course material also provides suggestions on how to employ these skills in class. Teachers should be made aware of the importance of these skills so they can foster them in their teaching.

3.3. Content Selection

Computer science knowledge and skills are necessary to understand changes and advances in the students' disciplines. To allow for the discussion of this digital transformation on a basis of CS, a well-founded selection of course content is important. When designing a CS course for students who will only take one such course, questions arise about which aspects of Computer Science are essential for everyone. When it comes to selecting the topics, we, as computer science educators, have different possible ways of selecting the content. Course designers rely on idea catalogs, such as the CS Principles (Astrachan & Briggs, 2012). Another idea catalog is the non-exhaustive list of Big Ideas behind K12 Computing Education (Bell, Tymann & Yehudai, 2018), including data representation, algorithms, complexity, comput-ability, digital representations, time-dependent operations, digital systems are designed to serve humans needs, and communication protocols. Others, with a slightly different focus, include the Principles of Computing (Denning, 2013) or the Fundamental Ideas of CS (Schwill, 1994).

These approaches are important to emphasize which topics make up computer science as a subject. But they do not necessarily help to make statements about which aspects are crucial as a basis for every non-CS teacher. Non-CS teachers need very specific knowledge about the basics of digitization and the effects on their subjects in a short amount of time. Therefore, our approach is two-fold. Aspects covered in university level non-major courses can form the basis for analyzing which aspects of CS can support pre-service teachers. In a study we analyzed the most common topics relevant in CS courses for non-majors (Seegerer & Romeike, 2018a, Seegerer & Romeike, 2018b). While the rising number of non-major CS courses may serve as an indicator for identifying key competences of CS considered important in the context of a digital transformation, the effects of digitalization in the subjects need to be considered as well. Accordingly, content was selected not only based on other CS courses for non-majors, but also based on tools, new methods and topics relevant across disciplines identified through literature, and the study with subject-matter teaching and learning experts.

Accordingly, we included basics about the digital representation of data, programming and algorithms. Due to the heterogeneity in prior knowledge of students, we saw a need to include computer systems and networks as well. Cryptography and limits of computing supplement the collection of topics. These topics are interweaved with other topics that gained importance in all related disciplines, namely simulations and different uses of data, such as data analysis.

4. CURRICULUM

The course is designed as a blended learning course. In the beginning, a face-to-face meeting takes place. This meeting is all about tinkering: students explore programming in Snap! using a MakeyMakey Board (Beginner's Mind Collective & Shaw, 2012). They build a banana piano and remix an existing video game to be controlled with the MakeyMakey. Most of the course is then done in an online learning environment. The online learning environment is designed to be engaging for students. Therefore, students often get the chance to gain hands-on experience by using the programming environment Snap!, or multiple interactives, e.g. adapted from the CS Field Guide (csfieldguide.org.nz.) The course ends with a short project phase, where students develop their own ideas in small groups and present them in the last session.

The course consists of 12 modules in total, 5 of them focusing primarily on CT (see Table 1). The CT part includes a module focusing on the impact of digitalization on disciplines, society and education as a whole, one module focusing on more technical aspects exploring the layers of abstraction inside computing systems, one on algorithms and one module concentrating on data analytics and simulations, respectively. Other modules related to CS focus on creativity or information gathering. They discuss different ways of fostering creativity in modern classrooms and emphasize the importance of creative thinking and working.

Table 1. Curriculum Contents and connection to CS.

Title of module	CS content
Fundamentals of Digitalization	Representation of data, programming
Secure use of computer systems and networks in professional teaching and learning	Computer systems, networks, cryptography, limits of computing
Solving subject-specific tasks with algorithms	Algorithms, programming
From data to domain knowledge	Working with data, programming
Modeling and simulations in domain-specific contexts	Modeling, simulations, programming

4.1. *Fundamentals of Digitalization*
This module marks the introduction into CT and takes place in week two. As the entire course is dedicated to the theme “Teaching in a digitalized world“, it starts off by identifying and explaining the differences between digital and analog representations. Photography is used as an example for transforming analog information into digital data. Subsequently, binary numbers are introduced as the basis of digital data storage of a computer. Programming languages are introduced as the language of digitalization. To engage students in problem-solving with programming, students create turtle art in Snap!. They are guided by tutorials introducing sequences and loops. The last task is to remix an existing project, allowing for creative expression.

4.2. *Secure Use of Computer Systems and Networks in Professional Teaching and Learning*
This module showcases the structure and operating principles of computers and computer networks. Students investigate the layers of abstraction in modern computer systems. They explore package transport via the internet, and discuss why antivirus programs are subject to fundamental limitations. Part of this module was also a section about cryptography, basic techniques, and its importance for society.

4.3. *Solving Subject-specific Tasks with Algorithms*
The module on algorithms concentrates on how automation is used in, or affects students’ subject areas. Students learn how to use conditional statements and apply it to creating a learning app with Snap!. Afterward, the term algorithm is introduced. Students explore what an algorithm is (and what it is not). Finally, students have to implement a certain algorithm. They can choose from different algorithms they already know from their subjects, or that have relevance in their discipline, e.g. calculate square roots, or determine word frequencies.

4.4. *From Data to Domain Knowledge*
Visualizations can be helpful for the interpretation of data, the generation of hypotheses, or generally for the clarification of correlations. Unfortunately, data is typically not accessible in such a visualized form. Usually, the data is stored as numbers or strings coded in databases, spreadsheets or certain files. One of the most important applications of computers in science is, therefore, the evaluation and analysis of digital data. Accordingly, one module concentrates on analyzing, visualizing, and interpreting data. Teachers engage in sense making of data using representations and learn the importance of being critical when interpreting data, not jumping to conclusions, and always deciding on the basis of the available data. In addition, students are encouraged to evaluate the importance of data analysis in their disciplines. In one exercise, for example, they use Snap! to fathom London's cholera outbreak from 1854.

4.5. *Modeling and Simulations in Domain-specific Contexts*
The weather, the solar system or particle motion are just some examples of complex science concepts that may require a mental model for learners. Modeling is particularly helpful in such contexts, as it allows us to focus on specific aspects of a real-world phenomenon (Weintrop et al., 2016). This includes the use of abstraction, as unnecessary details are left out. Prospective teachers learn about the importance of modeling for learners. In addition, they learn when and how to incorporate modeling and simulation into teaching. After using existing models that can be manipulated using parameters, the students learn about different aspects of a model itself, and use concept mapping as an easy for applying modeling in the classroom. Afterward, we present them with different exercises where they learn to model and simulate different contexts, including a predator-prey, a market, and an epidemic situation with Snap!.

5. EVALUATION
In the following, insights of an initial qualitative analysis for the first iteration of the modules will be outlined. These were provided in writing by five students at the end of each module. The questions included aspects like perceived personal benefit, personal effort, or perceived relevance. As most of the course was online, we were curious as to how certain topics can be taught. Fortunately, the online format was well received: “Interesting and ‘different’. It is more diversified, clearly more casually arranged than other online modules, and loosened up by exercises and practice.”

5.1. *Discuss Digital Transformation on a CS Basis*
As the students should gain a foundation in CS, we were interested in how well the content was received. One of the biggest challenges was the time constraint. Executing a meaningful module combining both typical lecture and lab content within the possible workload for one week proved difficult. Indeed, students have recognized and commented on a heavy workload for some modules. In order to let individual modules not become too extensive, the content of selected modules was streamlined after the first weeks. Still, students rated all content as personally meaningful.

5.2. *Show Relevance to Students of All Disciplines*

The course design aims to teach students how digitalization affects their teaching and their corresponding disciplines. Therefore, we tried to emphasize the relevance to the different subjects and disciplines on as many occasions as possible. Initial reactions show that the course can deliver on the promises. According to the students, they feel more confident to incorporate digital tools and new methods or topics in the classroom after completing the course. "I particularly liked the connection drawn to the subjects, where I've become aware of the specific topics that can be addressed in the classroom using Snap!."

The sections on data analysis and simulations, in particular, did particularly well in this aspect. The biggest potential for improvement is found in the very general section on Computer systems and networks.

5.3. *Profit from CT Inside and Outside the Classroom*

As with most Computer Science courses for non-majors, there are specific aspects that are crucial. Programming tasks in non-major courses are often considered problematic (e.g. (Banerjee & Kawash, 2009)), despite being important for CT. Nevertheless, in many university courses for students of other disciplines, programming is an essential component that is also addressed at a very early stage (e.g. (Garcia, Harvey & Barnes, 2015)). As pre-service teachers of all disciplines should profit from CT inside and outside the classroom, the question arises how an early reference to programming tasks affects the motivation and self-efficacy expectations of the students. We had similar constraints when designing the materials. We found that while students feel overwhelmed by the possibilities, the programming environment Snap! offers, they saw it as a fresh take on the topic and engagement in programming. The first contact with Snap! has been described as: "Websites like Snap! are great to try out things."

Nevertheless, there have been some entry barriers for students. As in most courses for non-CS majors that involve some kind of programming exercises, students face certain difficulties. Most difficulties have been solved by face-to-face meetings or via online communication. Especially the face-to-face meeting after students have worked on the material on their own has been regarded as very helpful. However, coding exercises still represent a hurdle, especially for online courses. We have tried to strike a balance between unrestricted and guided tasks. Students noted missing hints in some places, so they tended to use a trial-and-error approach by trying out different blocks. Regarding students' feedback, for future iterations, the inclusion of additional video material concerning programming is planned.

5.4. *Show the Importance of Skills Like Collaboration or Creativity*

Approaches like tinkering, collaborating and creating, are an important part of CT. Students apply these approaches themselves to learn how to think computationally, and to understand the importance of CT approaches for their own students. Additional readings provided students with the theoretical background and strategies for the application of these approaches in class. Students emphasized their understanding of the importance of these approaches, e.g. creativity, in the evaluation: "I became aware of the relation of creativity and teaching and will pay more attention to foster the creativity of students in my lessons."

6. CONCLUSION AND FUTURE WORK

In summary, we have presented a course for pre-service teachers at a German university, which introduces computing using the background of digital transformation. In the future, teacher training will have to deal increasingly with new topics and the effects of digitalization. Therefore, it will not be a question of whether but of how to incorporate Computer Science and Computational Thinking education into general teacher training. The paper describes design principles that can support the creation of similar courses and provides an example course. Its effort is to form a foundation for all teachers regardless of their subject and/or school type. Building on this foundation, teachers can discuss different aspects of digitalization within their subjects. So, while teachers might compare and debate data usage by different digital business models in economics, they may create or adapt a simulation for science teaching. By combining different aspects of Computational Thinking, we help prepare teachers to incorporate computing inside and outside of the classroom. Computational Thinking and Learning practices can enhance learning for students in all subjects. The course was initially piloted with a small number of students from a limited range of subjects. In the future, it will be opened for further teaching positions. Thus, we plan on integrating additional examples from a diverse range of disciplines. The material is now also adapted for in-service teacher training.

7. REFERENCES

Ananiadou, S., & McNaught, J. (2006). *Text Mining for Biology and Biomedicine*. London: Artech House.

Astrachan, O., & Briggs, A. (2012). The CS Principles Project. *ACM Inroads, 3*(2), 38-42.

Banerjee, S., & Kawash, J. (2009). Re-thinking Computer Literacy in Post-secondary Education. *Proceedings of the 14th Western Canadian Conference on Computing Education.* New York: ACM, 17-21.

Barr, D., Harrison, J., & Conery, L. (2011). Computational Thinking: A Digital Age Skill for Everyone. *Learning & Leading with Technology, 38*(6), 20-23.

Barr, V., & Stephenson, C. (2011). Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community*? ACM Inroads, 2*(1), 48-54.

Basu, S., Dickes, A., Kinnebrew, J., Sengupta, P., & Biswas, G. (2013). CTSiM: A Computational Thinking Environment for Learning Science through Simulation and Modeling. *Proceedings of the CSEDU.* Aachen, 369-378.

Beginner's Mind Collective, & Shaw, D. (2012). Makey Makey: Improvising Tangible and Nature-based User Interfaces. *Proceedings of TEI '12, Stephen N. Spencer (Ed.).* New York: ACM, 367-370.

Bell, T., Tymann, P., & Yehudai, A. (2018). The Big Ideas in Computer Science for K-12 Curricula. *Bulletin of EATCS, 1*(124).

Bellanca, J. (Ed.). (2010). *21st century skills: Rethinking How Students Learn*. Bloomington: Solution Tree Press.

Berman, F., Rutenbar, R., Hailpern, B., Christensen, H., Davidson, S., Estrin, D., Franklin, M., Martonosi, M., Raghavan, P., Stodden, V., & Szalay, A. (2018). Realizing the Potential of Data Science. *Communications of the ACM, 61*(4), 67-72.

Bharadwaj, A., El Sawy, O., Pavlou, P., & Venkatraman, N. (2013). Digital Business Strategy: Toward a Next Generation of Insights. *MIS Quarterly*, *37*(2), 471-482.

Brinda T., & Diethelm, I. (2017). Education in the Digital Networked World. *Proceedings of WCCE 2017.* Cham: Springer, 653-657.

Computing at School (2014). *Barefoot: Computational Thinking*. Retrieved January 3, 2019, from http://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking

Cooper, S., Pérez, L., & Rainey, D. (2010). K-12 Computational Learning. *Communications of the ACM, 53*(11), 27-29.

Denning, P. (2003). Great Principles of Computing. *Communications of the ACM, 46*(11), 15-20.

K-12 Computer Science Framework Steering Committee (2016). *K-12 Computer Science Framework. Technical Report*. New York: ACM.

Garcia, D., Harvey, B., & Barnes, T. (2015). The Beauty and Joy of Computing. *ACM Inroads, 6*(4), 71-79.

Goktas, Y., Yildirim, S., & Yildirim, Z. (2009). Main Barriers and Possible Enablers of ICTs Integration into Pre-service Teacher Education Programs. *Journal of Educational Technology & Society, 12*(1), 193-204.

Grillenberger, A., & Romeike, R. (2014). Big data – Challenges for Computer Science Education. *Proceedings of International Conference on Informatics in Schools.* Cham: Springer, 29-40.

Hegedus, S., Laborde, C., Brady, C., Dalton, S., Siller, H. S., Tabach, M., & Moreno-Armella, L. (2017). Uses of Technology in Upper Secondary Mathematics Education. *Proceedings of ICME-13*. Cham: Springer, 1-36.

Hey, T., Tansley, S., & Tolle, K. (2009). *The Fourth Paradigm: Data-intensive Scientific Discovery (vol. 1).* Redmond: Microsoft Research.

Kale, U., Akcaoglu, M., Cullen, T., Goh, D., Devine, L., Calvert, N., & Grise, K. (2018). Computational What? Relating Computational Thinking to Teaching. *TechTrends*, *62*(6), 574-584.

Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A Framework for Computational Thinking based on a Systematic Research Review. *Baltic Journal of Modern Computing, 4*(3), 583.

Lye, S., & Koh, J. (2014). Review on Teaching and Learning of Computational Thinking through Programming: What is Next for K-12? *Computers in Human Behavior, 41*, 51-61.

Müller, D., Frommer, A., & Humbert, L. (2013). Informatik im Alltag– Durchblicken statt Rumklicken [CS in Everyday Life - Understanding Instead of Clicking Around]. *Proceedings of HDI '13*, 98-104.

Riedel, M., Streit, A., Wolf, F., Lippert, T., & Kranzlmüller, D. (2008). Classification of Different Approaches for E-science Applications in Next Generation Computing Infrastructures. *Proceedings of 4th Inter-national Conference on eScience.* US: IEEE, 198-205.

Schwill, A. (1994). Fundamental Ideas of Computer Science. *Bulletin-European Association for Theoretical Computer Science, 53*, 274-274.

Seegerer, S., & Romeike, R. (2018a). Goals, Topics and Tools of Computer Science for All University or College Courses. *Proceedings of SIGCSE '18, 1090.* New York: ACM.

Seegerer, S., & Romeike, R. (2018b). Was Jeder über Informatik Lernen Sollte – Eine Analyse Von Hochschulkursen Für Studierende Anderer Fachrichtungen [What Everyone Should Learn About CS - An Analysis of University Courses for Students of Other Disciplines]. *Proceedings of HDI '18.* Potsdam: Universitätsverlag.

Seegerer, S., & Romeike, R. (2018c). Computer Science as a Fundamental Competence for Teachers in Other Disciplines. *Proceedings of WiPSCE '18, 149-150.* New York: ACM.

Smetana, L., & Bell, R. (2012). Computer Simulations to Support Science Instruction and Learning: A Critical Review of the Literature. *International Journal of Science Education*, *34*(9), 1337-1370.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.

Yadav, A. Zhou, N., Mayfield, C. Hambrusch, S., & Korb, J. (2011). Introducing Computational Thinking in Education Courses. *Proceedings of SIGCSE '11*. New York: ACM, 465-470.

Wing, J. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33-35.