Cultivating Computational Thinking Through Data Practice

Djordje M. Kadijevich(✉)

Institute for Educational Research, Belgrade, Serbia
djkadijevic@ipi.ac.rs

Abstract. After summarising the research context regarding defining, cultivating, and assessing computational thinking (CT), this theoretical paper examines data modelling using interactive displays, a CT practice that may be cultivated across several school subjects. Although working with data is not explicitly mentioned in some CT definitions, this work may activate different CT components, such as abstraction, decomposition, and pattern recognition. Furthermore, interactive displays, which are primarily a means for visualising data, can also be tools for modelling purposes if used within a modelling cycle. Focusing on this modelling in secondary education, we first consider main activities and their underlying skills, and outline what kind of support should be given to modellers, especially novices, in assisting them to complete this as easily as possible. We then consider what computational environment to use, which learning path to follow, and what assessment of learning to apply. Implications for teacher professional development are included.

Keywords: Computational thinking · Data modelling · Interactive charts · K-12 education · Teacher education

1 Introduction

Today, education needs to prepare students to cope successfully with increasingly complex life and work environments, which often rely on technology (i.e. on automated computations). Because of that, following Wing's account of computational thinking (CT) as one of the basic student abilities [1], many studies have dealt with CT in primary and secondary education across a number of school subjects by using various cultivation means. It seems that, in doing so, CT has not been intended to replace other contemporary approaches (e.g. problem-solving, critical thinking, creative thinking), but rather to complement and strengthen them by using concepts, tools, and techniques from computer science (e.g. [2]). As a result, students will be more than just technology-literate [3].

CT was originally used to denote thinking processes applied in problem-solving to formulate solutions in such representations that could be efficiently processed by computers [1]. It was viewed as an important literacy of the 21st century, which would, to some extent, enable everyone (all learners) to: recognise aspects of problems amenable to computation; match those aspects to appropriate computational supports (concepts, tools, techniques, strategies); understand the opportunities and limitations of

Published by Springer Nature Switzerland AG 2019
D. Passey et al. (Eds.): OCCE 2018, IFIP AICT 524, pp. 24–33, 2019.
https://doi.org/10.1007/978-3-030-23513-0_3

those supports; apply the supports in adapted or novel ways; and use computational strategies (e.g. a top-down approach) in any domain [4]. On the other hand, regarding scientists, engineers and other professionals, it was supposed that CT would enable them to reformulate problems to be (more) amenable to computation, develop and use new computational methods, ask and answer questions that rely on large data sets or intensive computations, and use computational terms to explain problems and solutions [4]. It can thus be said that CT may, in general, be viewed as a process whereby we recognise aspects of computations in our surroundings and deal, at introductory or advanced levels, with various systems and processes in these surroundings by applying tools and techniques from computer science [5].

Increasing societal reliance on technology and data calls for connecting CT and data practice in the classroom. In the rest of this theoretical contribution, we first summarise the research context regarding defining, cultivating, and assessing CT. We then examine data modelling in secondary education by using interactive displays. Focusing on this, we first consider main activities and their underlying skills, and outline what kind of support should be given to modellers, especially novices, in assisting them to complete the modelling as easily as possible. We then consider what kind of computational environment should be used (one in which CT components may additionally be fostered), what learning paths could be followed in doing that, and how modellers' progression along this path might be assessed. The paper ends with implications for teachers' professional development.

2 Research Context

2.1 Defining CT

Various definitions of CT have been proposed in the literature (e.g. [2, 6, 7]). Although the term has been used broadly, there has been no widely accepted definition so far [8]. For some researchers, from a general perspective, CT is concerned with algorithmic thinking, critical thinking, problem solving, and working cooperatively [9]. For some others, core CT facets are abstraction (data collection and analysis, pattern recognition, modelling), decomposition, algorithms (algorithm design, parallelism, automation), iteration, debugging, and generalisation [7]. Other researchers, looking at K-12 education, assume that CT is a critical component of problem-solving supported by technology [10, 11], and propose concepts, such as data collection, data analysis, data representation, problem decomposition, abstraction, algorithm and procedures, automation, parallelisation, and simulation as core ideas. It seems that the main goal behind the request to cultivate CT in K-12 education is to prepare students to use computational tools in productive and creative ways within different school subjects [6].

Apart from general frameworks, CT has been examined within subject-specific frameworks. For example, in the context of programming with Scratch, Brennan and Resnick [12] applied a CT framework with three dimensions, namely: *CT concepts* (e.g. data, operators, loops), *CT practice* (e.g. abstracting, modularising, debugging), and *CT perspectives* (e.g. questioning, connecting). In the STEM (Science, Technology, Engineering and Mathematics) context, focusing on high school mathematics and science education, a CT definition was given in the form of a taxonomy comprising four main practice categories [13]: *data practices* (e.g. collecting, visualising), *modelling and simulation practices* (e.g. building and using computational models), *computational problem-solving practices* (e.g. programming, troubleshooting), and *system-thinking practices* (e.g. defining systems, managing complexity). Another general CT framework [14], exemplified for mathematics pedagogy, comprises four overlapping activities with various objects (of digital, tangible, or conceptual nature). These activities are: *unplugging* (not using computers), *tinkering* (taking objects apart and changing/modifying their components), *making* (constructing new objects), and *remixing* (appropriating of objects or their components to use them at other places or for other purposes).

Clearly, a standard definition of CT is lacking. However, because of its pedagogical utility, it seems promising to define CT using various CT practices and activities, examined in terms of underlying CT concepts and skills (extrapolated from the NRC [15]). Although working with data is not explicitly mentioned in some CT definitions, one CT practice, namely data practice [13], should be included, since it may activate a number of CT components (e.g. abstraction, decomposition, and pattern recognition). Not only was the relevance of this practice for CT development (in particular, of data collection, representation, and analysis) recognised by the Computer Science Teacher Association (CSTA) and International Society for Technology in Education (ISTE) [11], but work with data has also been included in an international assessment of students' computer and informational literacy, which assumed that, apart from programming, the CT domain deals with structuring and manipulating data sets as well (for more details, see https://www.iea.nl/icils).

2.2 Cultivating and Assessing CT

Despite a relevant educational goal "CT for all" (initiated by Wing [1]), our knowledge of how to integrate CT in K-12 education is still in its infancy, because research on integration is scarce [6]. However, teacher education may, for example, benefit from examining examples of the use of CT in daily life. In other words, CT concepts (e.g. algorithm, abstraction, debugging) may be illustrated with concrete examples from teachers' daily experiences [16]. This approach, basically exemplifying various CT activities, has, for example, been applied by CSTA and ISTE [11]. If teacher education is based on the framework of technological pedagogical content knowledge [8], the main focus should be on developing knowledge of CT-related concepts, tools, and practice (technological knowledge) and combining them with disciplinary content (i.e. content knowledge) and pedagogical strategies (i.e. pedagogical knowledge). Additionally, to promote appropriate CT within specific subject domains, teachers should be encouraged to avoid using just a few tools (e.g. concepts mapping tools, interactive whiteboards) and CT concepts and practices (e.g. automation, problem decomposition).

Research also evidences that, in general, we should cultivate CT within rich computational environments (in different domains such as game design and development, and with various CT instances such as abstraction and automation), and, in doing so, apply a use-modify-create learning path [17]. Of course, having in mind different school subjects or university courses, CT practice should support or empower relevant scientific practice involving disciplinary knowledge and skills. Although CT may be promoted through activities without the use of computers (e.g. CS Unplugged [18]), the use of computing tools is nevertheless indispensable as they help learners test and revise their solutions involving CT concepts and practices (i.e. CT is primarily promoted through problem-solving with computing tools [10]).

Regarding CT assessment, there seems to be a vacuum in measuring and assessing CT achievement, which makes it difficult to judge the effectiveness of CT-based instruction [19]. In particular, because a standard definition of CT is lacking, measurements of this construct are diverse, which, as Shute and colleagues [7] underlined, not only raises questions regarding results obtained, but also makes them difficult to compare. These researchers also stressed that assessing CT in classrooms is challenging and that to support a teacher's instruction, real-time assessments that monitor students' progress may be required.

Having in mind Brennan and Resnick [12], appropriate assessments could be based on the analysis of students' project portfolios (involving artifact-based interviews with them), assuming that novice students progress in developing projects along, in our terms, an understand-debug-extend trajectory (i.e. from understanding a developed project via debugging this project to extending it). With more experienced students, a use-modify-create learning path [17] might be applied and assessed. To assess instruction that promotes CT among students by using computational tools in conjunction with content and pedagogy, we might use a technology integration rubric, whose criteria, as in [8], evaluate choosing and using tools and practices with respect to curriculum goals and instructional strategies, simultaneously aligning content, pedagogy, and technology.

3 Data Practice Using Interactive Displays

Despite the fact that a standard definition of CT is lacking, data practice, data analysis, or work with data can, as already mentioned, be recognised in a number of CT definitions (e.g. [7, 11, 13]). Even when work with data is not mentioned explicitly in a CT definition (e.g. as in Google's main CT elements: decomposition, pattern recognition, abstraction, and algorithm design; https://youtu.be/sxUJKn6TJOI), it is clear that, for example, pattern recognition, dealing with regularities and trends in data, are based on data practice, which may make use of suitable technology, such as interactive displays.

3.1 Interactive Displays and Their Educational Relevance

Interactive charts are digital devices for the visual presentation of data, whose content updates automatically after changes in considered data or variables. Interactive displays are digital artifacts comprising one or more such charts, possibly coupled with other interactive reports, such as tables or summary measures. Interactive displays composed of two or more interactive reports, usually interactive charts, are called dashboards.

Typically built in a drag-and-drop fashion, interactive charts can be, as a descriptive, exploratory tool, (relatively effortlessly) used to visualise regularities and trends in data, if any. Several interesting interactive charts may, for example, be found at https://www.dur.ac.uk/smart.centre/. The application of these charts, especially for dashboards (typically also built in the drag-and-drop fashion), has increased considerably in recent years (e.g. [20]; visit https://www.idashboards.com/dashboard-examples/, to view dashboards concerning various industries and areas). Learning analytics is, for example, one domain in which dashboards are widely used (e.g. [21]).

Because of such widespread and increasing use of dashboards, as well as possible learning and professional benefits, it is not surprising that there has been noticeable demand recently for the introduction of work with data using interactive displays in secondary education (e.g. [22–26]). Although this work has traditionally been associated with data analysis, possibly based upon complex mathematical and statistical models, it is unlikely that most students would be required to perform such analyses in their future jobs. They would rather do some basic data modelling using dashboards, whether produced by others or resulting from their own modelling, to support their professional claims and actions (e.g. "peer feedback has been used by less than one-third of e-learners"; "another drug dose must be administered to that patient"), which may particularly be relevant to the STEM disciplines [27].[1] This modelling just makes use of simple mathematical models (e.g. frequencies, sums, and means) connecting independent and dependent variables; each model is, after the developer's chart selection, automatically applied by the tool used. Although interactive displays are primarily a means for visualising data, they can also be tools for modelling purposes if used within a modelling cycle.

3.2 Considering Data Practice Through Modelling

Knowing that even simple data preparation (e.g. querying datasets, (re)organising data) may be quite challenging for novice modellers [25], data to model (with just a few variables) should be given to them. In that case, data modelling may only require them to complete three main activities, namely: asking questions; visualising data; and answering questions. In other words, there may be just three key stages in the modelling cycle, usually advanced in a nonlinear way. For experienced data modellers, remaining activities could eventually be added: validating modelling (recommending changes) after answering questions; and preparing data after asking questions [28].

To attain successful realisation of these activities, teachers need to identify their main underlying skills, and provide modellers, especially novices, with support to complete such modelling as easily as possible. Some of these underlying skills are, for example: choosing relations to examine; identifying dependent and independent variables (asking questions); selecting charts to use; selecting measures to apply (visualising data); recognising regularities in charts produced; and connecting regularities observed with corresponding questions asked (answering questions). These and other underlying skills should be fostered through suitable scaffolding, taking into account potential modelling challenges and reasons for these challenges (discussed elsewhere [25], for example). As the three activities depend on each other, many scaffolds would

[1] Job candidates with data practice skills (in particular data science and analytics skills) will soon be preferred by most employers in the United States, for example (see http://www.bhef.com/sites/default/files/bhef_2017_investing_in_dsa.pdf, for reported figures).

connect their underlying skills (e.g. variables selection with charts production; charts production with regularities recognition). Of course, when interactive charts, especially dashboards, are created from scratch, data modelling becomes a design task, whose central activity of problem structuring [29] needs meticulous scaffolding, possibly with (special) attention paid to the role of context in doing so. To ease contextual challenges, the data to model may be coupled with a short description of the underlying context. Furthermore, modellers may be scaffolded to develop problem structuring skills (e.g. selecting variables, measures, table, and charts to use) while improving knowledge of context under scrutiny (e.g. clarifying what other issues to examine, how they have been measured, and for what points in time to use data available), and vice versa (derived from [30]).

3.3 Performing and Assessing Data Practice Through Modelling

As underlined above, CT should be cultivated within rich computational environments [17]. Regarding dashboards, Zoho Analytics is, for example, such an environment (https://www.zoho.com/analytics/).[2] Apart from using and combining various interactive reports, it supports data preparation by querying relevant datasets. It also enables collaborative work on dashboard projects, which, if skillfully managed (e.g. in designing, building and combining charts of increased structural complexity), would promote other valuable CT assets, such as computational strategies.

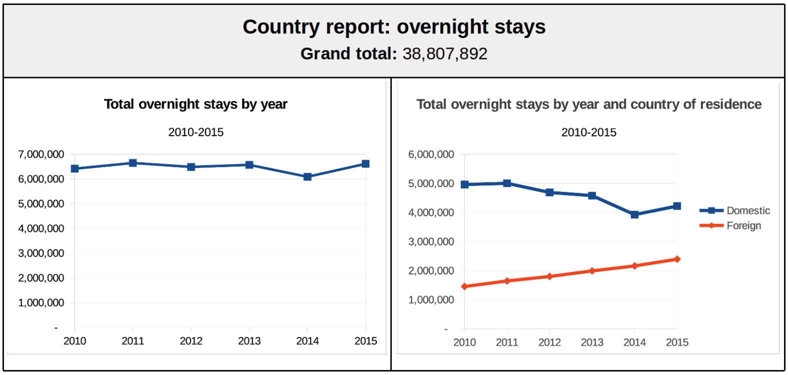

Fig. 1. Dashboard with two interactive charts and one summary measure: although not dominant, overnight stays by foreigners are increasing (modified from [25])

To illustrate some of these strategies, let us consider a dashboard presented in Fig. 1. Its design may be the result of decomposition (i.e. a divide and conquer strategy) of an issue under consideration (e.g. overnight stays in a country). This strategy is clearly applied in building each individual chart concerning its elements,

[2] Contrary to interactive charts, work with dashboards has only been supported by recent versions of some spreadsheet environments. To simplify dashboard creation, on-line publication and use, work with on-line dashboards has usually been realised in specialised, dashboard-tailored environments.

especially variables. Other strategies are top-down and bottom-up approaches. When we go from a dashboard as a whole to its individual reports as parts, a top-down approach is applied. When we start from some individual reports and combine them to create a dashboard, a bottom-up approach is used. (Instead of a single approach, we often apply a combination.) Furthermore, to get feedback and validation from peers and future users, building a chart or dashboard may make use of rapid prototyping, i.e. an iterative process through which we incrementally show what this display will look like. If we apply this interactive process to a chart, we may go from two variables (overnight stays by year) via three variables (overnight stays by year and country of residence) to four variables (overnight stays by year, country of residence and region, with values of the last variable used as a filter). In that way, we can examine whether the pattern of total overnight stays by year (the number of these stays is rather stable between six and seven millions in each year) holds true when we compare domestic and foreign guests, especially when we go from region to region.

As the dashboard development examines calls for simple system engineering, this development would only promote a basic understanding of these computational strategies. For a complex system, a top-down strategy could first be applied to identify its major units, followed by the main components of each unit. This strategy would then be repeated until the system is fully understood from top to bottom [31]. Note that contrary to decomposition, the three other strategies, especially rapid prototyping, have been under-represented in CT-related research (exceptions can be found [32, 33]; cf. CT facet named iteration [7]). For all these strategies to be put into practice, the use of a computing tool is indispensable, although some preparations for applying them may be done in an unplugged, paper-and-pencil environment.

Which learning path could be used to promote data modelling using interactive displays, and how may progression along this path be assessed? For work with digital artifacts in rich computational environments, Lee and colleagues [17] proposed applying the use-modify-create path (e.g. for our data practice, from playing with developed displays via modifying developed displays to creating displays from scratch). If we combine this path with the understand-debug-extend learning trajectory (i.e. from understanding a developed project via debugging this project to extending it [12]), we may arrive at the following learning path: *use displays* to understand or evaluate data modelling done – *modify displays* to debug or extend data modelling realised – *create displays* to perform full data modelling by yourself (cf. Dagstuhl perspectives, [34]). The evaluation of modellers' collaborative work may, as in Brennan and Resnick [12], be based on analysis of students' project portfolios, involving interviews with them about displays they have just evaluated, improved, or fully developed. To achieve this end, a rubric may be used with criteria that evaluate the success of pursuing each data modelling activity, making connections among them (e.g. in terms of major skills underlying these activities and links among them).

4 Closing Remarks

To empower learners for life in the digital age, various digital practices should be mastered. One of them, suitable for many (most) learners, especially in vocational education, is the data practice presented in this paper.

Although this practice is linked to CT in a mathematical context (and such studies are quite rare [35]), it may, embedded in other contexts with different learning cycles, contribute to the learning of computer science or statistics. For example, an extended context could ask for the preparation of data through a CT-based work with databases [36], whereby this practice becomes more relevant to computer science education where CT has become a critical component [37]. Or, instead of a modelling cycle, a data inquiry cycle could be used [23], making this practice relevant to statistics learning as well.

Apart from its relevance to developing CT, our approach seems to focus on CT pedagogy. Through considering a range of disciplines, it was recently proposed that pedagogical CT environments should primarily focus on interactive visualisations or simulations, modelling and troubleshooting of data sets, and searching for patterns in large data sets [38]. Clearly, our approach aligns with this focus.

To attain successful implementation of our approach in classrooms, professional development may primarily support teachers in realising and making connections between key data modelling activities in terms of their underlying skills (extrapolated from Niess and Gillow-Wiles [39]), being aware of potential challenges in this modelling and reasons for these challenges (e.g. [25]). To avoid some of these challenges (e.g. concerning data preparation and context understanding), teachers may be supported in preparing materials that will be given to students (e.g. data to model and short context descriptions). Detailed support may be provided for them to prepare scaffolds that would help students develop problem-structuring skills while improving context understanding, and vice versa. Professional development may also support teachers in applying particular learning paths and assessing their outcomes (e.g. *use displays* to understand or evaluate data modelling completed – *modify displays* to debug or extend data modelling done – *create displays* to perform full data modelling by yourself) [12, 17].

Apart from teachers and their appropriate professional development, successful implementation of our approach to data practice requires the support of other stakeholders (e.g. school administrators, policy makers, curriculum and assessment developers), who would eventually bring CT into classrooms and have it widely and skillfully applied (e.g. [13]). However, to support the claims that particular CT-interventions are beneficial to students, we need to improve assessment to provide solid evidence that specific CT components/facets were indeed promoted [7].

Acknowledgments. This contribution resulted from the author's work on the project "Improving the quality and accessibility of education in modernization processes in Serbia" (No. 47008), financially supported by the Serbian Ministry of Education, Science, and Technological Development (2011–2018). The author dedicates the contribution to his son Aleksandar.

References

1. Wing, J.M.: Computational thinking. Commun. ACM **49**(3), 33–35 (2006)
2. García-Penalvo, F.J., Mendes, A.J.: Exploring the computational thinking effects in pre-university education. Comput. Hum. Behav. **80**, 407–411 (2018)
3. Yadav, A., Hong, H., Stephenson, C.: Computational thinking for all: pedagogical approaches to embedding 21st century problem solving in K-12 classroom. TechTrends **60** (6), 565–568 (2017)
4. Wing, J.M.: Research notebook: computational thinking—what and why? Link Newsl. **6**, 1–32 (2011)
5. The Royal Society: Shut down or restart? The way forward for computing in UK schools. The Author, London (2012)
6. Voogt, J., Fisser, P., Good, J., Mishra, P., Yadav, A.: Computational thinking in compulsory education: towards an agenda for research and practice. Educ. Inf. Technol. **20**(4), 715–728 (2015)
7. Shute, V.J., Sun, C., Asbell-Clarke, J.: Demystifying computational thinking. Educ. Res. Rev. **22**, 142–158 (2017)
8. Mouza, C., Yang, H., Pan, Y.-C., Ozden, S.Y., Pollock, L.: Resetting educational technology coursework for pre-service teachers: a computational thinking approach to the development of technological pedagogical content knowledge (TPACK). Australas. J. Educ. Technol. **33** (3), 61–76 (2017)
9. Doleck, T., Bazelais, P., Lemay, D.J., Saxena, A., Basnet, R.B.: Algorithmic thinking, cooperativity, creativity, critical thinking, and problem solving: exploring the relationship between computational thinking skills and academic performance. J. Comput. Educ. **4**(4), 355–369 (2017)
10. Barr, V., Stephenson, C.: 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 (2011)
11. Computer Science Teacher Association, International Society for Technology in Education: Computational Thinking. Teacher Resources, 2nd edn. The Authors (2011). http://www.csteachers.org/page/CompThinking
12. Brennan, K., Resnick, M.: 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 (2012). http://scratched.gse.harvard.edu/ct/files/AERA2012.pdf
13. Weintrop, D., et al.: Defining computational thinking for mathematics and science classroom. J. Sci. Educ. Technol. **25**(1), 127–141 (2016)
14. Kotsopoulos, D., et al.: A pedagogical framework for computational thinking. Digit. Exp. Math. Educ. **3**(2), 154–171 (2017)
15. National Research Council: Next Generation Science Standards: For States, by States. The National Academies Press, Washington, DC (2013)
16. Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., Korb, J.T.: Computational thinking in elementary and secondary teacher education. ACM Trans. Comput. Educ. **14**(1), 1–16 (2014)
17. Lee, I., et al.: Computational thinking for youth in practice. ACM Inroads **2**(1), 33–37 (2011)
18. Bell, T., Witten, I.H., Fellows, M.R., Adams, R., McKenzie, J., Jarman, S.: CS unplugged: an enrichment and extension programme for primary-aged students (2015). https://classic.csunplugged.org/

19. Román-González, M., Pérez-González, J.-C., Jiménez-Fernández, C.: Which cognitive abilities underlie computational thinking? Criterion validity of the computational thinking test. Comput. Hum. Behav. **72**, 678–691 (2017)
20. Wexler, S., Shaffer, J., Cotgreave, A.: The Big Book of Dashboards: Visualizing Your Data Using Real-World Business Scenarios, 1st edn. Wiley, Hoboken (2017)
21. Webb, M.E., et al.: Challenges for IT-enabled formative assessment of complex 21st century skills. Technol. Knowl. Learn. **23**(3), 441–456 (2018)
22. Davison, C.B.: Addressing the challenges of teaching big data in technical education. CTE J. **3**(1), 43–50 (2015)
23. Gould, R., Bargagliotti, A., Johnson, T.: An analysis of secondary teachers' reasoning with participatory sensing data. Stat. Educ. Res. J. **16**(2), 305–334 (2017)
24. Ridgway, J.: Implications of the data revolution for statistics education. Int. Stat. Rev. **84**(3), 528–549 (2016)
25. Kadijevich, D.M.: Data modelling with dashboards: opportunities and challenges. In: Engel, J. (ed.) Promoting Understanding of Statistics About Society. Proceedings of the Roundtable Conference of the International Association of Statistics Education (IASE), Berlin, Germany, July 2016. ISI/IASE, The Haag (2016). https://iase-web.org/documents/papers/rt2016/Kadijevich.pdf
26. Metz, S.: Editor's corner: Big data. Sci. Teach. **82**(5), 6 (2015)
27. English, L.D.: STEM education K-12: perspectives on integration. Int. J. STEM Educ. **3**(3), 1–8 (2016)
28. Kadijevich, D.M.: Data modelling using interactive charts. Teach. Math. **21**(2), 55–72 (2018)
29. Jonassen, D.H.: Toward a design theory of problem solving. Education Tech. Res. Dev. **48**(4), 63–85 (2000)
30. Restrepo, J., Christiaans, H.: Problem structuring and information access in design. J. Des. Res. **4**(2) (2004).
31. Faulconbridge, R.I., Ryan, M.J.: Systems Engineering Practice. Argos Press, Canberra (2014)
32. Kong, S.-C.: A framework of curriculum design for computational thinking development in K-12 education. J. Comput. Educ. **3**(4), 377–394 (2016)
33. Ozcinar, H., Wong, G., Ozturk, T.H. (eds.): Teaching Computational Thinking in Primary Education. IGI Global, Hershey (2018)
34. Brinda, T., Diethelm, I.: Education in the digital networked world. In: Tatnall, A., Webb, M. (eds.) WCCE 2017. IAICT, vol. 515, pp. 653–657. Springer, Cham (2017).
35. Hickmott, D., Prieto-Rodriguez, E., Holmes, K.: A scoping review of studies on computational thinking in K–12 mathematics classrooms. Digit. Exp. Math. Educ. **4**(1), 48–69 (2018)
36. Juxiang, R., Zhihong, N.: Taking database design as trunk line of database courses. In: Proceedings of the Fourth International Conference on Computational and Information Sciences, pp. 767–769. IEEE, Washington, DC (2012)
37. Webb, M., et al.: Computer science in K-12 school curricula of the 21st century: why, what and when? Educ. Inf. Technol. **22**(2), 445–468 (2017)
38. National Research Council: Report of a Workshop of Pedagogical Aspects of Computational Thinking. The National Academies Press, Washington, DC (2011)
39. Niess, M.L., Gillow-Wiles, H.: Expanding teachers' technological pedagogical reasoning with a systems pedagogical approach. Australas. J. Educ. Technol. **33**(3), 77–95 (2017)