COMPUTATIONAL THINKING IN CONNECTION WITH THE ACQUISITION OF KEY SKILLS FOR THE 21ST CENTURY: REFLECTION FROM LITERATURE REVIEW

Takam Djambong

Université de Moncton (CANADA)

Abstract

Computational thinking (CT) is now considered either as an essential skill for the 21st century, either as the fundamental added value of digital literacy of citizens. This literature review aims to present computational thinking as a complex construct that is at the heart of 21st century skills development such as problem solving (PS) and collaboration skills, that can guide students to careers related to the field of science, technology, engineering and mathematics (STEM) disciplines, with high potential of employability and innovation, and, which can have an impact on economic productivity.

Keywords: Computational thinking, 21st century skills, problem solving, STEM.

1 INTRODUCTION

The concept of CT was mentioned for the first time by Seymour Papert[1], who in 1996 suggested a method to more effectively implement new problem-solving approaches. This concept, almost ten years later, will be taken up and popularized by Jeannette Wing through an influential article[2] that traces the origins of CT in the 1950s. Extensive research over the last decade has focused on issues related to teaching and learning skills, concepts and practices relevant to computational thinking. Since then, many studies were interested in this concept, trying to find him a definition and to clarify the concepts, techniques, teaching strategies or learning, assessment methods associated with it, and its articulation with the 21st century skills. This article proposes the synthesis of a number of studies that explored these different aspects of CT.

2 A UNIQUE CONSTRUCT, BUT MULTIPLE DEFINITIONAL APPROACHES

We became interested in studying computational thinking within the work of the CompéTICA (Compétences en TIC en Atlantique, www.competi.ca) project, a new partnership network whose aim is to define digital competences, to describe practices contributing to their development, as well as to initiate cross-disciplinary collaborations facilitating transferability and adaptability of these competences of the life-long continuum1] et [2]. The concept of computation thinking is one of the sets of competences we consider as essential part of digital competences which needs to be analyzed in its complexity through a variety of definitional approaches used in the literature. Among these approaches, four are of particular interest: computational thinking as essential 21st century skill, as a problem-solving process, as part of the STEM (Science, Technology, Engineering and Mathematics) educational movement, and key-element of digital literacy. In the next sub-sections, we analyze each aspect in more details.

2.1 Computational thinking as an essential skill for the 21st century

From this perspective, CT is often seen as a skill set that everyone needs to develop in connection with other 21st Century skills such as critical thinking, productivity or creativity [5]. In addition to reading, writing and arithmetic, CT should be included, to every child's analytical ability as a fundamental skill to be used by everyone, by the middle of the 21st Century [6]. Consider the fact that computers are actually among the most important technical innovations of the modern society, CT and its implementation in the K-12 level is a need for the 21st Century, in order to be successful in a world that is increasingly dependent on complex problem solving tasks [7]. Many in the field of education, particularly in educational technology, agree with computer science education community that CT is

[1] The Connected Family: Bridging the Digital Generation Gap, 1996.

[2] published on 25 November 2005 under the title, "*Computational thinking.*"

ISBN: 978-84-608-8860-4

an important 21st century skill [8]. At the same time, they recognize that the link between solving problems related to computational thinking and solving such problems envisaged in the 21st century skills is not yet clarified. What are the differences and similarities between these two sets of skills?

2.2 Computational thinking as a problem-solving process

One of the approaches that are becoming popular among researchers and frequently found in the literature is to relate CT to the process of problem-solving. Wing (2006) describes CT as a process that "*involves solving problems, designing systems, and understanding human behavior by drawing on the concepts fundamental to computer science (CS), including a range of mental tools that reflect the breadth of the field of computer science*" [7]. In 2009, a movement towards a need of consensual language surrounding the essential elements of CT emerged from a project funded by the National Science Foundation (NSF) and, jointly led by International Society for Technology in Education (ISTE) and Computer Science Teachers Association (CSTA), during a meeting involving diverse groups of educators with an interest in CT from higher education, PK-12 and industry. From that meeting, the following operational definition was proposed: "*CT is a problem solving process that includes (1) Formulating problems in a way that enables us to use a computer and other tools to help solve them; (2) Logically organizing and analyzing data; (3) Representing data through abstractions, such as models and simulations; (4) Automating solutions through algorithmic thinking (a series of ordered steps); (5) Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient combination of steps and resources; (6) Generalizing and transferring this PS process to a wide variety of problems.*" [6]. Cuny, Synder and Wing have also defined CT as "the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by and information processing agent" [7]. Also in connection with PS, other definitions described CT as: (1) "*thought processes involved in formulating problems so their solutions can be represented as computational steps and algorithms*" [10]; (2) "*a mental orientation to formulating problems as conversions of some input to an output and looking for algorithms to perform the conversion*" [11]; (3) "*a conceptual way to systematically, correctly, and efficiently process information and tasks to solve complex problems*" [12]. It is important to note that, Wing's model of CT is not limited to PS methods derived from CS [13], but CT can be spread from CS to other academic fields which have their own specialized PS methods by adapting existing CT theory and methods to match the need of novices and others non specialists [13].

At the more operational level, the problems (and problem-solving) are addressed from the CT perspective as being sometimes reduced to a set of discrete variables that are likely to be transformed into abstract data, before leading to the formulation of an algorithm. This approach to solving a problem takes into account only one aspect of the nature of a problem and ignores the others. For instance, this definition of computational thinking in connection with problem solving, omits the fact that any aspect of a problem to solve is the result of negotiations between the limited resources on the one hand, and the diversity of sides compete for access to these resources on other hand. Computational thinking, in reality represent a particular vision of solving problems among many, and cannot therefore be general or be able to solve any type of problems [14]. Because not all problems could be formulated in the way to be processed by an automatic data processing tool, an interesting question, therefore, would be to know what could be the link between the different approaches to solving problems in a perspective of hybridization of different approaches based on disciplines and fields of application, and also according to the nature and complexity of the problems to be solved. Another interesting question to investigate would be whether the mental and cognitive processes are mobilized in the same way depending on the angle from which one approaches the issue of solving the problems which are associated or not with computational thinking.

2.3 Computational thinking as complex construct enabling 21st Century and STEM skills acquisition

Because of its nature as a complex construct, CT is often seen as being related to a number of 21st Century competencies such as problem solving, critical thinking, productivity, creativity, collaboration and communication [3]. One of the issues, according to [15], is that CT can foster creativity as one important 21st century competency, by enabling students not only to be consumers of technology, but also to build tools that can have significant impact on the on the abililty to be an innovative and creative citizen.

In other hand, it is important to underline the role and value of CT in STEM preparation. In this perspective, CT considered as tool enabling the development of a variety of skills, and involving the

use of scientific and technological methodologies, helps develops both inventiveness and innovative thinking. CT then, has its roots in STEM and in the synthesis of ideas from all these fields [16]. CT can be also described as a type of analytical thinking that requires STEM skills thinking methodology to understand and solve complex problems within the constraints of the real world [17]. It has been also suggested that, one key approach to support STEM education is to infuse CT elements within STEM topics, in order to , prepare, train and equip future investigators in STEM disciplines with necessary skills to face the challenge of complex problems that would not be solvable unless CT is involved [18].

2.4 Computational thinking as the fundamental aspect of digital literacy and computer science

Computational thinking can be seen as the fundamental aspect of digital literacy to the extent that, it integrates the power of human thinking with the capabilities of computers. In this digital and information age, everyone must be develop the ability to think computationally, in order to move technology projects beyond just using tools and information, toward creating tools and information. And this computing-based creativity requires thought processes about data, using abstractions, and lots of computer science concepts [19]. In this perspective, learning to program for instance involves using computational thinking skills and concepts, in such a way that, CS courses must focus more on computational concepts [12]. However a position paper by ECDL Foundation supported by the Council of European Professional Informatics Societies (http://www.ecdl.org/media/PositionPaper-ComputingandDigitalLiteracy1.pdf) argues that more holistic approach is required while two common misconceptions need to be addressed: one is related to the view that today's students growing in working with computer programs are 'natively' digitally literate so that no formal learning is necessary; the second is grounded in the belief that eveyone who knows programming (or at least coding) would be better positioned for the job market; indeed only few would get a job as IT professionals; rather, nearly everyone would need basic digital literacy skill. Therefore, further clarification of this aspect of CT is needed.

3 THE MAIN CONCEPTS OF COMPUTATIONAL THINKING

In a disciplinary perspective, three types of concepts in computer thought emerge from the analysis of previous research: concepts that can be called general because of their application to a wide enough field of disciplines, specific concepts to computers as a basic discipline of computational thinking and concepts specific to the science disciplines, technology, engineering and mathematics (STEM) with very close ties with the computer science.

3.1 General concepts

Some concepts related to CT can be regarded as mental or cognitive processes (abstraction, algorithmic operations, decomposition, and recognition of forms) and real results (automation, data visualization, modeling, simulation, generalization, etc.) related to problem solving. The core CT concepts according to [20], include: (i) *abstractions* (the mental tools of computing necessary to solve the problem); (ii) *layers* (problems to be solved on different levels); (iii) *relationships* (between layers and abstractions). The eleven following CT concepts[3] has been identified through research (CSTA, 2009): (i) *Abstraction* (identification and extraction of relevant information to define the main ideas); (ii) *algorithm design* (creation of an ordered series of instructions for solving similar problems or for doing tasks); (iii) *Automation* (having computers or machines do repetitive tasks); (iv) *data collection* (gathering information); (v) *data analysis* (making sense of data by finding patterns or developing insights); (vi) *data representation* (depicting and organizing data in appropriate graphs, charts, words, or images); (vii) *decomposition* (breaking down data, processes, or problems into smaller, manageable parts); (viii) *parallelization* (simultaneous processing of smaller tasks from larger task to more efficiently reach a common goal); (ix) *pattern generalization* (creating models, rules, principles, or theories of observed patterns to test predicted outcomes); (x) *pattern recognition* (observing patterns, trends, and regularities in data); (xi) *simulation* (developing a model to imitate real-world processes). Among these eleven concepts, five[4] can be considered as general ones, because they are applicable to almost all disciplines. These are: *abstraction* (AB), *algorithm design* (AL), *decomposition* (DE), *pattern generalization* (GE), *pattern recognition* (PR) and evaluation (EV) [21].

[3] Could be found at https://www.google.com/edu/resources/programs/exploring-computational-thinking/index.html#!ct-materials

[4] Could be found at https://studio.code.org/unplugged/unplug2.pdf

These general concepts rely upon the fact that, CT is not thinking about computers or like computers, but, it is about looking at a problem in such a way that a computer can help someone to solve it.

3.2 Concepts specific to computer science

Seven CT concepts have been identified to be specific to CS: (i) systematic processing of information; (ii) symbol systems and representations; (iii) algorithmic notions of flow and control; (iv) iterative, recursive, and parallel thinking; (v) conditional logic; (vi) efficiency and performance constraints; (vii) debugging and systematic error detection [22]. Seven other concepts viewed as great principles of computing can be considered as a foundation that is useful to recognize, organize and categorize instances of CT and build a framework that can translate CT to contexts outside of CS. [23]. These concepts are: (i) *computation*; (ii) *communication*; (iii) *recollection*; (iv) *evaluation*; (v) *design*; (vi) *automation*; (vii) *coordination*. Another set of CT concepts related to CS can be viewed as techniques or "computational doing". These include: (i) *reflecting* (skill of making judgements); (ii) *coding* (translation of the design into code form and evaluation of the code to ensure that it functions properly under all anticipated conditions); (iii) *designing* (working out the structure, appearance and functionality of artefacts); (iv) *analysing* (using logical thinking both to better understand things and to evaluate them as fit for purpose); and (v) *applying* (adoption of pre-existing solutions to meet the requirements of another context) [24].

3.3 Concepts specific to STEM disciplines

Four major categories of STEM practices have been identified as CT concepts related to STEM disciplines. These are: (i) *data analysis*; (ii) *modelling and simulation*; (iii) *computational problem solving*; and (iv) *system thinking* [25]. Each of these categories is composed of a subset to five to seven practices. In addition, all these practices are highly interrelated and dependent on one another [10], in such a way that, by fusing them with STEM disciplines, students can explore and apply CT skills within a more established and accessible STEM context. In this way, STEM can facilitate CT learning [26].

4 THE SKILLS RELATED TO COMPUTATIONAL THINKING

4.1 Skills specific to general concepts of CT

Many cognitive skills have been identified to be associated with general concepts of CT. Each of the CT concepts described previously, is identified with distinct learner behaviour as shown in table 1 below [24].

Table 1: Skills related to CT taxonomy [24].

CT general concept	Skills associated with CT general concept
Abstraction (Ability to think in abstraction, choosing good representation)	• Reducing complexity by removing unnecessary detail • Choosing a way to represent an artefact, to allow it to be manipulated in useful ways • Hiding the full complexity of an artefact • Hiding complexity in data • Identify relationship between abstractions • Filtering information when developing solutions

Algorithmic thinking (Ability to think algorithmically)	• Formulating instructions to achieve a desired effect • Formulating instructions to be followed in a given order (*sequence*) • Formulation instructions that use arithmetic and logical operations • Writing sequences of instructions that store, move and manipulate data (*variables and assignment*) • Writing instructions that choose between different constituent instructions (*selection*) • Writing instructions that repeat groups of constituent instructions (*loops/iteration*) • Grouping and naming a collection of instructions that do a well-defined task to make a new instruction (*subroutines, procedures, functions, methods*) • Writing a set of declarative rules (*coding in Prolog or a data query language*) • Using an appropriate notation to write code to represent any of the above • Creating algorithm to test a hypothesis • Creating algorithmic descriptions of real world processes so as to better understand them (*computational modelling*) • Designing algorithmic solutions that take into account the abilities, limitations and desires of the people who will use them
Decomposition (Ability to think in terms of decomposition)	• Breaking down artefacts into constituent parts to make them easier to work with • Breaking down a problem into simpler versions of the same problem that can be solved in the same way (*recursive and divide and conquer strategies*)
Evaluation **(**Ability to think in terms of evaluation)	• Assessing that an artefact is fit for purpose • Designing and running test plans and interpreting the results (testing) • Comparing the performance of artefacts that do the same thing • Stepping through processes or algorithms/code step-by-step to work out what they do (dry run/tracing) • Using rigorous argument to justify that an algorithm works (proof)
Generalization (patterns) (Ability to think in generalization, identify and make use of pattern)	• Identify patterns and commonalities in artefacts • Adapting solutions, or parts of solutions so they apply to a whole class of similar problems • Transferring ideas and solutions from one area to another

Here the distinction between the concepts and skills is not easy to show. Upon analysis, a concept could have a sense of overall competence and skills that of sub-skills or micro-skills. It would be wise at this point to bring more clarification on the relationship between a concept and skills that are attached to it. .

4.2 21st Century skills developed through CT practices

As already mention before, CT has the potential to foster the development of some 21st essential skills. Four of these 21st Century skills that can be developed through CT practices have been identified. They are: communication (communicate thought processes and results), collaboration (collaborate with peers on computing activities), problem solving and collaborative problem solving

(CPS). Dede et al. (2013)[5] draw attention to the need of delineation of computational thinking's boundaries with other fields to increase its contribution to the 21st competences development.

4.3 Skills related to STEM disciplines

The CT-STEM skills taxonomy has been built up from three main resources: exemplary CT-STEM classroom activities, matching existing CT and STEM inventories of concepts and standards establishing documents, as well as interviews with STEM professionals that use CT in their work. These approaches were chosen to ensure validity of this taxonomy [26], which is broken down into the four main categories (see table 2 below): data and information, modelling and simulation, computational problem-solving skills and system thinking skills.

Table 2: CT-STEM skills taxonomy [26].

CT-STEM category	Skills associated with the category
Data and information	• Collecting data • Creating data • Manipulating data • Analyzing data • Visualizing data
Modelling and simulation	• Using computational models to understand a concept • Using computational models to find and test solution • Assessing computational models • Designing computational models • Constructing computational models
Computational PS skills	• Preparing problem for computational solutions • Programming • Choosing effective computational tools • Assessing different approaches/solutions to a problem • Developing modular computational solutions • Creating computational abstractions • Troubleshooting and debugging
System thinking skills	• Investigating a complex system as a whole • Understand the relationship within a system • Thinking in level • Communicating information about a system • Defining system and managing complexity

5 HOW TO INTEGRATE COMPUTATIONAL THINKING INTO K-12 CURRICULA

5.1 The main strategies of integration

According Wing (2008), computational thinking should not only be part of the curriculum of undergraduate, but also be integrated into the curricula at the primary and elementary levels. But the question is which concepts of computational thinking must be taught and at what levels during the student's education? Indeed, two main approaches are suggested for the integration of computational thinking in educational curricula [6]:

[5] http://www.curtin.edu.au/edusummit/local/docs/Advancing_computational_thinking_in_21st_century_learning.pdf

1 Computational thinking taught as part of a specific discipline; In the UK, for example, the Royal Society (2012) suggested to integrate computational thinking in primary and secondary computer science curricula as a school discipline;

2 Computational thinking taught as general competence across several disciplines;

And in this context, [22] estimate that beyond the fact that the computer thought was erected as basic scientific practice by the Next Generation Science Standards (NGSS) in the United States, it would be doubly important to integrate computational thinking in educational curricula of STEM disciplines. Google Computational Thinking Repository is an example of such development initiative.

Here, we might wonder about the real impact that the integration of computational thinking strategy in the curricula could have on the implementation of the problem solving process itself. The different possible integration strategies are they mutually exclusive or could there be a possibility of hybridization of different approaches according to the learning contexts and student profile? Another path to explore would be to consider the development of skills related to computational thinking in the context of teaching a school discipline that is not directly related to computer science.

5.2 Learning models of computational thinking in the school context

In the school context, several models of learning in computer thought were listed. Among these models, we can mention:

1 *The learning model without computer* (Computer Sciences Unplugged, Computer Science in-a-box Unplugged your Curriculum, The Value of Computational Thinking Accros Grade Levels)

2 *Learning models using visual environments*. The tools used for this purpose may for example include: *Scratch* [27], *AgentSheets* and *AgentCubes* [28], *Scalable Game Design* [29].

3 *Learning models using the programming environments through the use of games, programming and robotics* as *Alice* [30] or *Greenfoot* [31];

4 *Learning models using modelling and simulation environments* [32];

5 *Learning models by integration with other disciplines*. It may include initiatives such as the Computational Thinking Learning Experience, Exploring Computational Thinking, Musicomputation race [33], Interactive Multimedia, Computational Thinking for Science, Technology, Engineering and Mathematics (CT-STEM).

None of these learning models of computational thinking cannot yet be characterized as stable model given the absence of consensual and clear the definition of the construct of computer thinking and also given the difficulty to identify, caracterize and articulate the different concepts and dimensions associated with this construct. An interesting path would be to identify what particular aspects of computational thinking each of these models seem to favor the most.

6 HOW TO ASSESS COMPUTATIONAL THINKING SKILLS?

6.1 Assess computational thinking as a formative latent variable

CT can be considered as a formative latent variable, because its operational definition has emerged in 2010 through a large consensus among diverse group of educators with an interest in CT from higher education, PK-12, and industry [8]. Then, as a formative latent variable, CT cannot be directly observed or measured. It must be inferred from other directly observable and measurable variables. This fact makes the need to have a working definition of computational thinking in which the variables are not only defined quite explicitly, but are also observable and measurable [28].

6.2 Examples of assessment tools used to measure CT

One of the many challenges which educators and researchers face today, in addition to develop a definition of the concept of computational thinking which is consensual, is the scarcity of tests validated and widely available to measure the skills acquisition and mobilization of skills related to computational thinking. Some research has yet addressed this issue. Here, are some of them that have been reported in the literature and listed below:

- Tests based on standardized exercises as a collection tool [33];
- Tests based on the traces of activities in an IT environment as collection tool [34];
- Tests based on a combination of standardized exercise and activity traces in an IT environment as collection tool [26];
- Using the already validated cognitive tests [9];
- Other various measurement tests identified such as *Brebas*[6] *International Challenge on Informatics and Computational Thinking*, the *Google for Education Exploring CT*[7] problems, and Computer Science Unplugged[8] tests activities.

Despite the absence of a consensual conceptual and operational definition, the results of a recent study on the different tests measuring the computational thinking seem to conclude that [35]:

- The use of error-correction tasks is an appropriate approach to measure and evaluate the skills associated with computational thinking;
- Nested tasks in partially completed projects not only assess the real level of skills related to computational thinking, but can also be implemented on a large scale;
- Learning environments for solving problems on the constructionist model integrating the treatment of information, reflection and construction activities could be developed to promote and measure the acquisition of skills associated with computer thought.

Note that very few of these different tests have yet been empirically validated in a large scale school setting and could therefore constitute a useful field of CT to investigative in the future.

7 CONCLUSION

The study we conducted under this article shows that computational thinking is constructed which as yet imprecise still in its conceptualization operationalization, and implementation as learning object in schools. Teaching practices to promote the development of digital skills among young people in schools are still far from being stabilized, as is the use of different teaching tools supporting their implementation. The case study we are conducting within the CompéTICA project could provide the opportunity to develop or test such tools, which in the context of science education at the elementary or secondary could help to improve practice of CT development, along with problem solving skills in a technology-rich environment.

AKNOWLEDGEMENTS

This ongoing study is being conducted with the help of the Canadian Social Sciences and Humanities Research Council (Partnership Development Grant #890-2013-0062), New Brunswick Innovation Foundation (2016 Research Assistantship Program) and le Secrétariat aux Affaires Intergouvernementales Canadiennes du Québec (Programme de soutien à la Francophonie canadienne). CRSH (Social Sciences and Humanities Research Council of Canada).

REFERENCES

[1] Freiman, V., Godin, J., Larose, F., Bourgeois, Y., LeBlanc, M., Léger, M. T., Robichaud, X., Chukalovskyy, R., Cormier, D. & Pelletier, W. (2015). Towards the construction of a theoretical and methodological framework to study partnership development for digital competences within the CompéTICA network. Dans *Proceedings of Global Learn* (pp. 306-311): Association for the Advancement of Computing in Education (AACE). DOI: 10.13140/RG.2.1.3977.6807

[2] Gauvin, S., Paquet, M. & Freiman, V. (2015). Vizwik – visual data flow programming and its educational implications. In S. Carliner, C. Fulford & N. Ostashewski (Eds.), Proceedings of EdMedia: World Conference on Educational Media and Technology 2015 (pp. 1602-1608). Association for the Advancement of Computing in Education (AACE).

[6] http://www.bebras.org/

[7] https://www.google.com/edu/resources/programs/exploring-computational-thinking/

[8] http://tabs.chalifour.fr/la-science-informatique-a-lecole/cs-unplugged/

[3] Dede, C., Mishra, P., Voogt, J. (2013). Advancing computational thinking in 21st century learning. EduSummit.

[4] Wing, J. (2005). Computational thinking. A vision for the 21st Century. p.1.

[5] Shailaja, J., Dean, S. (2014). Computational thinking the Intellectual Thinking for the 21st century. p.39.

[6] Voogt, J., and al. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. p.720.

[7] Wing, J. (2006). Computational thinking. Viewpoint. p.33.

[8] Barr, D., Harrison, J., Conery, L. (2011). Computational Thinking: A Digital Age. ISTE. p.21.

[9] Ambrosio, P.A., Almeida, L., Macedo, J., Franco, A. (2014). Exploring Core Cognitive Skills of Computational Thinking. PPIG, University of Sussex. p.1.

[10] Aho, A. V. (2012). Computation and computational thinking. Computer Journal. 55 (7), pp.832-835.

[11] Denning, P. (2009). The profession of IT – Beyond computational thinking. Communication of the ACM. 52 (6), pp.28-30.

[12] Weintrop and al. (2015). Defining Computational Thinking for Mathematics and Science Classrooms. Springer.

[13] Czerkawski, B.C., Lyman III, E.W. (2015). Exploring Issues about Computational Thinking in Higher Education. University of Arizona. p.59.

[14] Easterbrook, S. (2014). From Computational Thinking to System Thinking. A conceptual toolkit for sustainability computing. ICT4S'2014.

[15] The College Board (2012). Computational thinking practices and big ideas, key concepts, and supporting concepts.

[16] Einhorn, S. (2012). MicroWorlds, Computational Thinking, and 21st Century Learning. LCSI White Paper.

[17] Voskoglou, M., Buckley, S. (2012). Problem Solving and Computers in a Learning Environment. ECS, Vol. 36 N°4. p.32.

[18] Swaid, S.I. (2015). Bringing computational thinking to STEM education. Procedia Manufacturing. p. 3657.

[19] Phillips, P. (2009). Computational Thinking. A Problem Solving Tool for Every Classroom. CSTA.

[20] Wing, J. (2008). Computational thinking and thinking about computing. Phil. Trans. R. Soc. A. (2008) 366, pp.3717-3725.

[21] Pontelli, E. (2011). Computational thinking. Department of Computer Science. New Mexico State University.

[22] Grover, S., Pea, R. (2013). Computational Thinking in K-12: A Review of the State of the Field. Educational Researcher, Vol.42 N°1, pp. 38-43.

[23] Chande, S.V. (2015). A Conceptual Framework for Computational Thinking as a Pedagogical Device. IJIRCCE. Vol.3, Issue 11, p.11683.

[24] Csizmadia and al. (2015). Computational thinking. A guide for teachers. Computing at School.

[25] Bienkowski, M., Snow, E., Rutstein, D., Grover, S. (2015). Assessment Design Patterns for Computational Thinking Practices in Secondary Computer Science: A First Look. SRI Education.

[26] Weintrop and al. (2014). Interactive Assessment Tools for Computational thinking in High School STEM Classrooms. LNICST 136, p.22-25.

[27] Tchounikine, P. (2016). Initier les élèves à la pensée informatique et à la programmation avec Scratch.

[28] Repenning and al. (2014). Scalable Game Design: A Strategy to Bring systemic Computer Science to Schools through Game Design and Simulation Creation. ACM Trans. Computers Education.

[29] Ioannidou, A. (2011). Computatutional Thinking Patterns. AERA.

[30] Conway, M.; Audia, S.; Burnette, T.; Cosgrove, D.; & Christiansen, T. (2000). Alice: lessons learned from building a 3D system for novice. *In Proceedings of the SIGCHI conference on Human factors in computing systems* pp. 483-493. ACM.

[31] Rick, D.; Ludwig, J.; Meyers, S.; Rehder, C.; & Schirmer, I. (2010). Introduction to business Informatics with Greenfoot using the example of airport baggage handling. *In Computing Educational Research*. pp. 335-340. Dallas.

[32] Allan and al. (2012). Computational Thinking for Youth. *In The ITEST Small Group on Computational Thinking.*

[33] Dee Miller et al. (2014). Integrating Computational and Creative Thinking to Improve Learning and Performance in CS1. University of Nebraska, Lincoln.

[34] Koh, H.; Basawapatna, A.; Nickerson, H.; Repenning, A. (2014). Real Time Assessment of Computational Thinking. University of Colorado, Boulder.

[35] Zhon, J., Liao, H. (2013). The study on Features of Computational thinking and its common Operation Mode Methods.