A Systematic Mapping Study on Assessing Computational Thinking Abilities

Ana Liz Souto O. de Araujo
Software Practices Laboratory
Federal University of Campina Grande
Departament of Exact Sciences
Federal University of Paraíba
Paraíba, Brazil
analiz@copin.ufcg.edu.br

Wilkerson L. Andrade
Software Practices Laboratory
Federal University of Campina Grande
Paraíba, Brazil
wilkerson@computacao.ufcg.edu.br

Dalton D. Serey Guerrero
Software Practices Laboratory
Federal University of Campina Grande
Paraíba, Brazil
dalton@computacao.ufcg.edu.br

***Abstract*—Several initiatives have been created to promote Computational Thinking (CT) abilities in students. There are multiple approaches of assessing CT and wide abilities and skills involved. However, the evidence on how to assess CT has not yet been systematically grouped or reviewed. The goal of our study is to identify and classify approaches to promote CT and the different ways of assessing CT abilities. To achieve this goal, a systematic mapping study was planned and executed. The results reveal that: *(i)* programming courses are the most common pedagogical approaches to promote CT for K-12 students; *(ii)* multiple skills are involved in CT, but solving problems, algorithms, and abstraction are most common abilities assessed; and *(iii)* codes and multi-choice questionnaires are the most common artifacts for assessing CT abilities. This study points out to the fact that there are open questions for exploring and developing new researches for promoting and assessing CT abilities.**

I. Introduction

According to Wing, in her seminal paper on the subject, computational thinking is about developing "fundamental, not rote skills" for problem solving. She mentioned, among others, the ability to think recursively, use abstraction and decomposition when dealing with complex tasks, and to use heuristics to reason about possible solutions [1]. Thinking like a computer scientist requires being able to think in multiple levels of abstraction.

Since then, several researchers have focused on defining the core set of skills that characterize computational thinking (CT). According to Hu, CT represents a cognitive process and hence, it should be seen as a hybrid paradigm that accommodates different thinking models such as logical, algorithmic, analytic, mathematical, engineering and creative thinking [2]. As a consequence, several different abilities and skills are associated with CT. For instance, while analysis and generalization are frequently employed in discussions of general problem solving skills, abilities like designing systems, programming computers, as well as being able to apply concepts of automation and modeling are more frequently employed in discussions of general computer science concepts [3].

Most research results related to CT skills and abilities can be found in recent academic literatures. Some studies have focused on the relation of CT and teaching in K-12. Grover *et al.* presented environments and tools that foster CT and how they interact with Computing Education in K12 [4]. Voogt *et al.* presented a historical review of CT concepts, and gathered examples of how CT is taught. They also discussed the position of CT in the K-12 curriculum [5]. Taking a different perspective, Selby and Woollard have recently proposed a definition of CT and the main skills involved [3], based on a study of the most frequently terms and skills associated to CT [3].

A related, but different field of investigation focuses on how to assess CT skills and abilities. In the report of the Workshop on Pedagogical Aspects of Computational Thinking from 2011, the National Research Council cites three reasons for assessing computational thinking abilities: *(i)* to judge the curriculum and related materials and pedagogy, *(ii)* to judge the individual's progress, and *(iii)* to manage instructor training and support [6]. However, while a significant amount of research has been developed on skills, the discussion on their assessment is still in its infancy.

In this study we focus on the approaches used by researchers and teachers to observe and assess the development of CT abilities in students. As far as we know, there is no systematic review of the scientific literature on the subject of CT assessment. We identified and classified proposed approaches to promote CT and the different ways used to assess abilities within those approaches. More concretely, the study aimed at *(i)* characterizing the state of the art of research on assessing CT abilities in education, *(ii)* identifying and classifying the different approaches and artifacts used to assess CT abilities in various educational levels, and *(iii)* discussing the results based on approaches, artifacts, and abilities assessed. To achieve this goal, we planned and executed a systematic mapping study based on the guidelines established by Petersen *et al.* for such studies [7].

The rest of this paper is organized as follows. In Section II, we briefly discuss related work. In Section III, we present our research methodology and the research questions that guided the study. In Section IV, we present the data collected along the study. In Section V, we present and discuss the results achieved. Finally, in Section VI, we present our conclusions.

II. Related work

To the best of our knowledge there are no literature reviews about assessing CT abilities. The existing studies about CT involve definition, skills and role of CT in K-12 [4] [5] [3]. These studies are important because they show an overview of essential aspects of CT and indicate the abilities that should be promoted and assessed.

The systematic review presented by Barcelos *et al.* considered studies that describe and evaluate approaches to integrate CT and mathematics [8]. A wide variety of mathematical topics are developed with emphasis on Algebra, Calculus and higher-order thinking skills. Programming is the main approach, followed by robotics and spreadsheets. Mathematical model is not related to CT according to this study. Another systematic review identified tools used for promoting CT [9]. Scratch[1] is the most common tool for promoting CT, followed by App Inventor[2] and Alice [3]. Both systematic reviews are interested in promoting CT with mathematics or tools, but they do not show interest in how to assess the development of CT in those context.

The literature review presented by Lye and Koh focused on teaching CT through programming [10]. The computational concept (i.e. concepts that programmer use) is widely explored in promoting CT through programming. The authors propose that more intervention studies focused on computational practices and computational perspectives could be conducted for the promotion of CT. In addition, they propose that a constructionism-based problem-solving learning environment, with information processing, scaffolding and reflection activities could be designed to foster computational practices and computational perspectives. The authors do not reveal how to assess the proposed approach.

III. Research Methodology

This section details the methodology used for executing this systematic mapping study. The objective of a mapping study is to categorize several primary studies, frequently providing a visual summary of results. In addition, clusters and research gaps are evidenced allowing the identification of possible open problems for future works [7].

In a mapping study, papers are categorized based on research questions aiming at providing a visual summary of the field. This methodology has been increasingly used in Software Engineering by Petersen *et al.* [7]. Because of this, we chose the protocol defined by Petersen *et al.* and applied it to the Computer and Education field. As every mapping study has a clear protocol to guide, each step and outcome of the protocol is detailed in the rest of this section.

A. Research Questions

There are multiple approaches for promoting and assessing CT along with several involved abilities. So, the following research questions (RQ) were defined in order to guide this systematic mapping study:

- **RQ1 -** What are the pedagogical approaches for promoting CT and for which kind of audience?

The purpose is to elicit what kind of schemes/activities the researchers apply to promote the acquisition of CT skills and for which audience (students' grade) these studies have been carried out.

- **RQ2 -** What are the skills assessed in CT?

The aim is to bring forth which abilities or skills are assessed by the pedagogical approach. In this work, skill and ability are considered as synonyms.

- **RQ3 -** What are the instruments or artifacts for assessing CT abilities?

The objective is to identify which instruments or artifacts are used in order to measure the CT abilities in each pedagogical approach.

B. Conduct Search

After defining the research questions, a strategy was designed to select the related papers. The following steps were adopted:

1) Analyze the terms to be used in the search string in order to answer the research questions;
2) Decide what the central idea in Computational Thinking and assess abilities is;
3) Choose synonyms of "assessing" in the context of education;
4) Test each synonym of "assessing" in search, analyze results in each digital library, and select the synonyms;
5) Use Boolean OR to connect the selected assessing synonyms;
6) Link "Computational Thinking" using Boolean AND;
7) Execute pilot tests to gauge the search string.

As a result, the following search string was obtained:

"Computational Thinking" AND (assess OR analyze OR evaluation OR measure OR validation)

Six digital libraries were chosen to conduct the research: ACM, ERIC, IEEEXplore, ScienceDirect (Elsevier), Springer, and Scopus. These digital libraries were chosen because they are the main publication vehicles of Computer Science and Education. Table I presents the summary of the search returned by each digital library. The studies were collected based on a title/abstract search. Thus, in this step, the outcome was all papers selected in the initial search.

C. Papers Screening

All inclusion and exclusion criteria applied in this mapping study are described below.

The defined inclusion criteria are:

1) Studies that are about assessing some ability or skill in the context of CT;
2) Studies in any kind of educational level/grade;

[1] https://scratch.mit.edu/
[2] http://appinventor.mit.edu/explore/
[3] http://www.alice.org/

TABLE I
SUMMARY OF PRIMARY SEARCH RESULTS

Digital Library/Database name	Search Results
ACM	69
ERIC	11
IEEEXplore	57
ScienceDirect	12
Scopus	224
Springer	115
Total	488

3) Studies reported in workshop OR conference OR journal.

The defined exclusion criteria are:

1) Studies that do not introduce how to assess or measure CT abilities or skills;
2) Studies that do not introduce "analysis" or "validation" of CT abilities, i.e. papers that analyzed or validated the approach instead of CT abilities;
3) Studies that are not in the education context;
4) Studies that are in Computational Biology;
5) Secondary studies, short papers and chapter of book.

The selection process was conducted in two steps. In the first stage, the title and abstract of all papers selected in the first search were read. In this step, duplicated papers, secondary studies and short papers were removed. 51 studies were judged as potentially relevant after this point. In the second step, all papers were read and the inclusion and exclusion criteria were applied. Most studies were excluded because they did not demonstrate how to assess or measure CT abilities. Accordingly, in this last step, 27 studies were selected as relevant to our goals. To avoid search bias, all stages were discussed by the three authors. The outcome of this step was all studies selected to answer the research questions based on the inclusion and exclusion criteria.

IV. RESULTS

This section details the result of our mapping study. The research was conducted from September 2015 to April 2016. First of all, the general results extracted directly from the headers of the papers are presented, such as countries where the research has been done, publication vehicle (journal or conference), and studies distribution by year of publication. Then, the remaining sections answers each research question defined in Section III.

A. General results

1) Countries distribution: Fig. 1 shows the countries distribution where each research has been done. The United States is the leading country where approaches of assessing CT abilities have been investigated (16 studies). After that, South Africa, Italy and UK have 2 studies and other countries, such as Brazil, China, Germany, Hungary, India, Israel, New Zealand, Romania, Singapore, and South Korea have only 1 study.

The United States is the leading country in number of publication. A possible explanation is the presence of the Computer Science Teacher Association (CSTA) Computational Thinking Task Force. CSTA Computational Thinking Task Force[4] is responsible for current developments in CT and dissemination of teaching and learning resources related to CT.

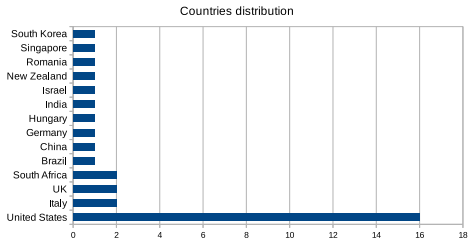

Fig. 1. Countries distribution

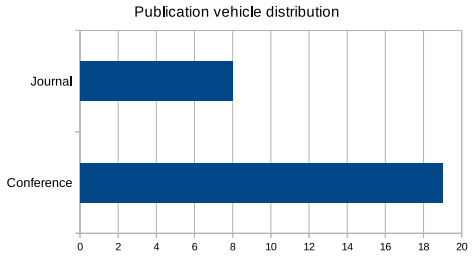

Fig. 2. Publication vehicle distribution

2) Publication vehicle: Fig. 2 presents the publication vehicle adopted by the selected studies. Publication in conferences are the most frequent type with 70% (19 studies from 27). Journals appear with 30% (8 studies from 27).

3) Publication years: Fig. 3 presents the studies distribution by year of publication. We do not delimit time to conduct the research, but the term "Computational Thinking" earned new signification in 2006 by Jannette Wing [1]. However, no work before 2006 was found. The first selected study is from 2009 (1 study). In 2010, 2011, and 2012 few studies were reported about assessing CT abilities (3, 1 and 3 studies, respectively). From 2013, there was an increase in the number of studies: 5 studies in 2013, 6 studies in 2014, 7 studies in 2015, and 1 study in 2016 (until April 2016).

[4]https://csta.acm.org/Curriculum/sub/CompThinking.html

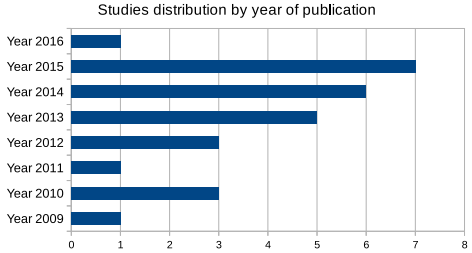

Fig. 3. Studies distribution by year of publication

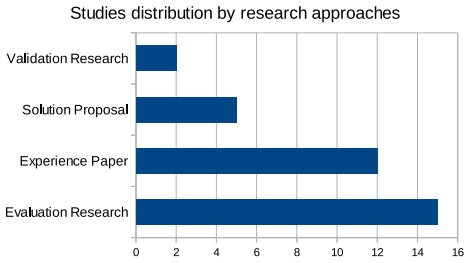

Fig. 4. Studies distribution by research approaches

TABLE II
PEDAGOGICAL APPROACHES FOR PROMOTING CT

Approach	Reference	Frequency
Course	[11] [12] [13] [14] [15] [16] [17] [18] [19] [20] [21] [22] [23]	13
Test	[24] [25] [26] [27] [28] [29]	6
Framework	[30] [31] [32] [33] [34]	5
Tool	[35] [15] [36]	3

4) Classification of research approaches: The studies also were classified based on the research approaches. The research approaches chosen by Petersen *et al* [7] were the following: validation research, evaluation research, solution proposal, experience papers, philosophical papers, and opinion paper (for definitions, see [7]). Fig. 4 shows the result of studies distribution by research approaches. Philosophical papers and opinion papers were not found in the selected studies.

B. RQ1: What are the pedagogical approaches for promoting CT and for which kind of audience?

Table II presents four approaches applied to promote CT: course, test, framework, and tool. In this mapping study, the term course includes courses in general regardless of the duration. Moreover, the term course includes workshops, modules or regular classes. Test includes studies that only applied exams. Framework is a model with the intention of measuring CT abilities. At last, tool comprises game, digital ink and platform (online interactive platform that offers several activities to foster and evaluate CT abilities). From this point of view, course was the leading pedagogical approach for promoting CT (13 studies). Tests were the second most common (6 studies), followed by frameworks (5 studies), and tools (3 studies).

Since course was the most popular pedagogical approach, Table III presents a classification by course content. Programming course was the most common pedagogical approach (9 studies). Scratch was the most common programming environment [11], [20], [23], [31] followed by App Inventor [18], [20]. Two courses were specific about Computational Thinking [12], [13]. The remainder included other programming environments as Alice and AgentClub [17], [19], a game design course with Kodu [37], a computer literacy course with spreadsheets [16], a web design course [14], an introductory programming course with multiple languages [20], and a multidisciplinary collaboration class [22].

TABLE III
COURSES FOR PROMOTING CT

Course Content	Reference	Frequency
Scratch	[11] [31] [20] [23]	4
Computational Thinking	[12] [13]	2
App Inventor	[18] [20]	2
Alice	[17]	1
AgentClub	[19]	1
Game design (Kodu)	[37]	1
Computer Literacy Course	[16]	1
Web design	[14]	1
Introductory programming	[20]	1
Multidisciplinary collaboration class	[22]	1

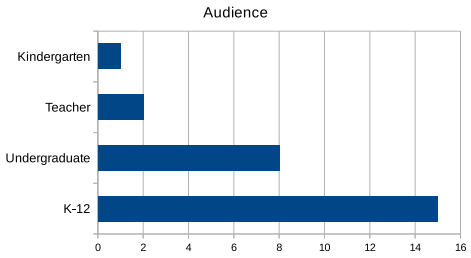

Fig. 5. Audience

In addition, this mapping study aimed at knowing for which kind of audience (students' grade) these studies have been carried out. Fig. 5 grouped the students' grade. K-12 students were the leading audience (15 studies), followed by undergraduate students (8 studies). The undergraduates are Computer Science students and non-Computer Science students. Kindergarten has one study. Teachers and pre-service teachers have two studies.

C. RQ2: What are the skills assessed in CT?

The aim is to elicit which abilities or skills are assessed by the selected approaches. The measured skills and abilities were identified, but each study has adopted many different groups of abilities. Table IV shows the skills and abilities assessed by each pedagogical approaches. The most common abilities are emphasized in bold: problem solving, algorithms, abstractions, and decomposition. For the best visualization of the frequency of abilities, Fig. 6 shows a word clouds from Table IV.

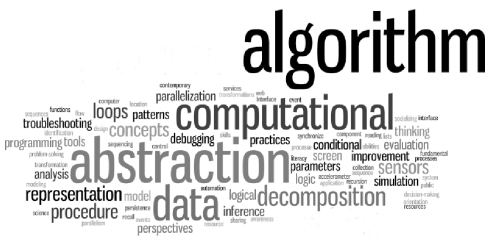

Fig. 6. Word clouds of skills and abilities assessed in CT

TABLE IV
SKILLS AND ABILITIES ASSESSED IN CT

Skills / Abilities	Reference	Frequency
Algorithm	[17] [22] [23] [13] [12] [31] [30] [24] [26] [27] [28] [15] [36] [35]	14
Abstraction	[17] [14] [22] [13] [12] [31] [30] [26] [18]	9
Programming	[23] [11]	2
Logical thinking	[22] [12]	2
Data representation	[23] [13] [31]	3
Data collection, data analysis	[13]	1
Modeling	[17] [15]	2
Decomposition	[13] [12] [11] [31]	4
Procedures	[13] [31] [20]	3
Automation	[13]	1
Parallelization	[13] [31] [20]	3
Simulation	[13] [15]	2
Debugging	[12] [15]	2
Sequences, loops and conditionals	[11] [33] [20]	3
Computational concepts, computational practices and computational perspectives	[32] [34]	2
Sensors	[33] [18] [20]	3
Processes and transformations, models and abstractions, patterns and algorithms, tools and resources, inference and logic, evaluations and improvements	[30] [26]	2
Condicionals	[11]	1
Recall	[16]	1
Synchronize	[21]	1
Troubleshooting	[19] [37]	2
Computer Science fundamental concepts and computational literacy	[25] [36]	2
Screen Interface, events, data persistence, data sharing, lists, public web services, accelerometer, location awareness	[18]	1
Parameters, functions, recursion, event, screen interface	[20]	1

D. RQ3: What are the instruments or artifacts for assessing CT abilities?

The aim is to present the instruments or artifacts used in order to measure CT abilities. Table V shows the instruments or artifacts classified based on approaches. Questionnaire is classified in different types: multiple-choice, open-ended, surveys and interviews. In a multiple-choice questionnaire, the student need to choose the only one correct alternative. In an open-ended questionnaire, the student need to write or draw the correct answer. In a survey there is no right or wrong answer, just the students opinion in a multiple-choice questionnaire. Finally, in a interview the student speaks answers to questions, but there is not necessarily right or wrong answer.

Multiple-choice questionnaire is the most common artifact for assessing CT abilities in eleven studies [24] [11] [25] [26] [27] [37] [23], [28], [34], [36]. Code is the second in ten studies [31] [32] [17] [18] [33] [19] [20] [21] [23] [34], followed by responses in nine studies [35] [14] [26] [16] [28] [29] [36]. Survey, interviews, open-ended questionnaires, lesson plan, video journals, and design scenarios are other instruments for assessing CT abilities. Fig. 7 shows only the frequency of instruments and artifacts. The number of instruments is more than number of selected studies because each study can use more than one instrument for assessing CT abilities. Fig. 8 combines the results of instruments and research approaches in bubble plot.

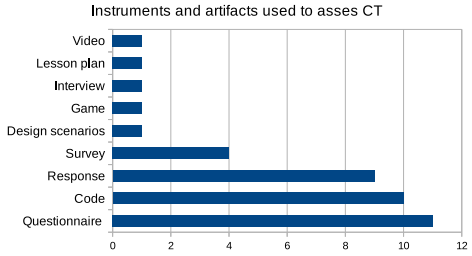

Fig. 7. Instruments and artifacts used to asses CT

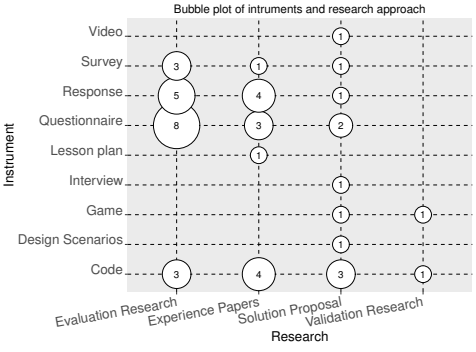

Fig. 8. Instruments and research approaches bubble plot

TABLE V
INSTRUMENTS AND ARTIFACTS USED TO ASSESS CT CATEGORIZED BY APPROACH

Approach	Instrument or artifact	Reference
Course	Code	[17] [18] [19] [20] [21] [23]
	Responses	[14] [16]
	Lesson plan	[13]
	Survey	[22]
	Multiple-choice questionnaire (PISA)	[37]
	Survey and open-ended questionnaire	[12]
	Survey and multiple-choice questionnaire	[11]
Framework	Code (Scratch)	[31]
	Code (Scratch), interviews, and design scenarios	[32]
	Code (Robotic), video and responses	[33]
	Code and questionnaire	[34]
	Game	[30]
Test	Multiple-choice	[24] [25] [26] [27] [28]
	Responses	[26] [28] [29]
Tool	Survey (game)	[15]
	Questionnaire, response, game (platform)	[36]
	Responses (digital ink)	[35]

V. DISCUSSION

This section discusses the data presented in Section IV regarding the concept of CT, pedagogical approaches, assessed abilities, instruments, and artifacts. Finally, the threats to validity are presented.

A. The concept of Computational Thinking

The concept of CT can influence the abilities assessment. Once understanding the concept of CT, the abilities can be elicited and appropriate instruments can be selected to assess theses abilities. The definitions found in literature are broad, making it difficult to find a systematic classification. In addition, some studies propose new definitions instead of analyzing existing ones. Other studies propose activities without exposing their vision about CT. Thus, a large interpretation on CT is seen in the selected studies.

Regarding theoretical references, most studies refer to Wing's seminal papers. Only 5 from 27 studies do not refer to Wing's precursors papers. These studies were published in 2014 and 2015. Although Wing's seminal papers have strong influence on literature, more recent studies are proposing new terms and abilities.

B. Pedagogical approaches

1) Programming courses: Programming courses are the most common intervention to promote CT. Visual programming environments are broadly employed, such as Scratch, App Inventor Alice, and AgentClub [11] [31] [20] [23] [18] [20] [17] [19]. Although these environments are attractive, they seem to be limited to solve programming problems, make digital storytelling, and game design. In spite of that, they are feasible to introduce students to software programming because of the visual programming language. This technical feature allows students to learn programming concepts and practice CT abilities while avoiding the syntax hurdle associated with text-based programming languages. Moreover, they are appointed to be pedagogically appropriate to K-12 students.

Computational Thinking is not an alternative to learn how to program. However, many studies argue that, the CT can be initiated through programming. Many other studies judge that programming is a good approach to promote CT, but few of them worry about how to assess the development of CT, individual's progress, and learning difficulties. Programming can promote other abilities that may or may not be considered by CT, for instance: testing, debugging, looking for optimal solutions. Because of that, the concept of CT and the involved abilities should be discussed in the context of programming courses, specifically for non-computer science students.

2) Computational Thinking workshops: Computational Thinking workshops have been designed to target both teachers [13] and pre-service [12] teachers. In one study, pre-service teachers participated in a one-week module of CT (two 50-minute classes) during the educational psychology course. The module aims at demonstrating probabilistic reasoning, algorithmic thinking, heuristics, hypothesis testing, and problem solving. The assessment was conducted based on a survey and three open-end questions for *(i)* explaining the concept of CT, *(ii)* knowing whether CT can be integrated into the classroom, and *(iii)* whether CT it is related to other disciplines. In another study, teachers participated in a three days workshop that showed them the correlation between their taught subjects and CT. They were stimulated to use CT examples to teach in their respective classrooms. The lesson plan done after the workshop were assessed in order to measure the teacher's ability to synthesize the CT core concepts.

The courses differ about purpose, time duration, content and type of assessment. In the first course, the pre-service teachers were analyzed whether they had understood the content and whether they had intended to apply CT in future classroom. The content involved multiple abilities beyond Computational Thinking but programming was not involved. In the second course, teachers were assessed for the ability to connect the CT practices in their own subjects. An evaluation rubric was created to assess whether teachers effectively had used the computational thinking core concepts in their lesson plans. In this case, the teachers had a programming experience with Scratch and Alice.

3) Other courses: The options to promote CT without programming courses were game design courses, web design courses, computer literacy courses, and multidisciplinary collaboration classes. All these courses are distinguished by purpose, content, audience, and assessed abilities. These approaches were not programming courses but they were related to them and they were dependent on computers. No course promotes CT without computers.

C. Abilities

Problem solving is involved in all papers, except for one [18]. Therefore, problem solving is considered as the core of

Computational Thinking. In addition, other abilities collaborate to solve problems. In the mapping study, two papers treat troubleshooting instead of problem solving [19], [37]. In these papers, troubleshooting can be seen in the most common types of problem solving which usually possess a single fault state, have known solutions, rely most efficiently on experience-based rules for diagnosing, and require learners to make judgments about the nature of the problem. Considering this definition, troubleshooting can fit into a context of CT.

Algorithm is directly related to 20 studies and indirectly to 3 (from 27) in this mapping study. Thereby, algorithmic thinking has been pointed out as one of the most fundamental abilities related to solving problems. Designing an algorithm is a way of producing a solution through a set of steps. Defining and following those steps help us to achieve a solution to a problem. For this reason, algorithmic thinking can be considered essential in CT.

Wing, in her seminal paper, argued that the essence of CT is abstraction [38]. For the author, an algorithm can be seen as abstraction of a step-by-step procedure for taking input and producing some desired output. Moreover, thinking abstractly involves identifying the heart of the problem and visualize different levels of details to help to solve it. Abstraction also helps to simplify complex problems and decomposes complex tasks. Abstraction is related with algorithm and problem-solving. Therefore, abstraction can be considered essential in CT as well.

Computer Science Teachers Association (CSTA) points out the following set of nine CT abilities to K-12 students: data collection, data analysis, data representation, problem decomposition, abstraction, algorithms and procedures, automation, parallelization, and simulation [39]. These abilities are explored in [13]. CSTA argues the role of CT in K-12 is "a problem solving methodology that can be automated, transferred and applied across subjects" [39]. CSTA also designed the CT Teacher Resources and the CT leadership toolkit to help teachers to apply CT in their classes. So, these nine CT abilities can be appropriate to K-12 education.

The CSTA's study already highlighted that CT is not an alternative to learn programming, but CT can be incited through programming. Some papers considered conditional, loop, procedures, parameters, functions, recursion and event as CT abilities. Those terms have correlation to programming, but the CT abilities in the context of programming needs to be more discussed. In another paper, Mobile Computational Thinking (MCT) is the term used to refer the programming aspects of CT plus mobile programming aspects designed for MIT App Inventor [18]. The rubric of assessment MCT includes screen interfaces, events, component abstraction, data persistence, data sharing, lists, public web services, accelerometer, orientation sensors, and location awareness. Although those abilities are very specific to mobile programming, they are considered part of CT abilities.

The most wide set of abilities appeared in [26], [30]. They elicited *(i)* processes and transformations, *(ii)* models and abstractions, *(iii)* patterns and algorithms, *(iv)* tools and resources, *(v)* inference and logic, *(vi)* evaluations and improvements. This set of abilities is wide to encompass CT and can involve activities with and without technology for development and assessing of CT.

D. Instruments and artifacts for assessing CT

Since programming courses were the most common approaches to promote CT, code is already expected as one of the most common artifact for assessing CT abilities. In addition, four frameworks use code to measure CT. The problem in using code to measure development of CT is to limit the assessment to just a checklist, i.e., the presence or absense of some programming structure. In accordance to Lye and Koh [10], computational practices and computational perspectives should be included in the context of programming and computational thinking. Moreover, to measure CT abilities based on code should be more discussed.

Questionnaires (multiple-choise and open-ended questionnaires) also were common instrument for assessing CT. Questionnaires, such as pre-test and post-test, applied before and after courses have advantages and limitations. In most of these cases, it is not possible to analyze the impact of specific activities offered during the course, as highlighted [37]. In the context of programming course, pre-assessment programming tasks could frustrate students who did not have previous knowledge about programming. These students could have a negative impact on their attitudes towards future activities, as highlighted [19].

Multiple-choice questionnaires were one of the most common type of questionnaire. Multiple-choice questionnaires have the advantage of being quicker to reproduce and compile the results. These questionnaires are simpler to summarize for statistical analysis. The challenge is to elaborate an appropriate test in order to measure cognitive processes. Some studies elaborate its own questionnaires. Other studies use a known test to measure students' problem-solving skills, as PISA (Program for International Student Assessment) [37].

On the other hand, responses to exercises also were used for assessing CT abilities. Responses to exercises have the possibility to understand the specific role of individual activities and their impact on students' skills [35]. Content analysis allows us to make inferences about the reasoning that student appear to have been using, and gain further insight into students thought processes. Moreover, content analysis can identify the students' mistakes when they do some exercises during a course or an experiment as in [14]. Besides, it can help to understand the range of errors and identify potential patterns for further investigations and propose a solution for it.

Test is the traditional instrument for assessment. However, the concern is about the planning and content of this instrument. The teacher can underestimated the difficulty of the test or plan their test based on previous experience, not considering appropriate pedagogical foundations for the audience. In the context of CT, there is not consolidated test to apply for measuring CT. However, this is already expected, once the

concept of CT and essential abilities are not well defined in literature.

Considering tools, this mapping study identified three results: one game, one digital ink, and one platform [15], [35], [36]. The game was not tested in a real environment. The digital ink has been undergoing validation process to verify its adequacy as an assessment tool for CT. The platform is a prototype with the purpose of collaboratively assess students by exercises (multiple-choice and open-ended questionnaires, game, puzzle). So, all of them need to be tested for empirical evidences of how the students react and how they could be assessed in practice.

The use of games can be a strategic pedagogical approach to foster CT. Educational games have the benefits to provide fun and help students to learn about subjects or assist them in learning a certain skill while they play. Games can be designed to assess abilities that teacher would like to train and measure as [15].

Tools can be projected in order to capture more information about students' performance during the activity. Specific digital tools can act as an entry mechanism and store data from each subject. This data may be retrieved and processed for future analysis concerning different parameters. Such tools can be a way to assess not only the final result, but also the development process. The digital ink proposed in [35] is an example of this tool. Platforms also can be designed to capture more data about the students' performance. Moreover, platforms can help teachers with automated assessment and repository of questions/tasks as [36].

E. Threats to Validity

This mapping study has some threats to validity. The result may not only be affected by the limitation of the automated search engines of each digital library but also by to human factors during the screening of papers and data extraction steps. So the search could not identify all relevant studies or extract all relevant information. It was limited to peer-reviewed conference papers and journal articles available. This study considers neither book chapters nor short papers.

VI. Conclusion and future works

This research has reinforced in our minds that there is a broad and varied interpretation of CT concepts and consequently of the approaches used to promote it. While these different interpretations are not irreconcilable, they do enforce different views that lead to significantly different assessment approaches.

However, we were able to identify the most commonly used approaches. General problem solving, in particular, is considered in the vast majority of studies (96%) and direct programming courses are the most popular pedagogical approach used to promote CT for K-12 students (62%) (RQ1). Within programming courses, visual programming environments are broadly employed — in particular, tools like Scratch and App Inventor are, currently at least, very popular. These pedagogical approaches, however, are rather limited to programming problem solving, digital storytelling, and game design.

The study also revealed that a broad range of different CT skills and abilities are promoted by researchers and teachers and as a consequence, several artifacts, metrics and dimensions are used in assessment. The abilities assessed in each artifact varies depending on the researcher's conceptualization of CT. Solving problems, developing algorithms, and applying abstraction are the most common abilities assessed among the studies identified (RQ2). Coding and multi-choice questionnaires are the leading artifacts for assessing CT abilities (RQ3). While the artifacts are the same, the method used to actually assess the code produced as well as the answers to questionnaires may vary, according to the pedagogical approach, the purpose of the evaluation and, finally, to the interpretation of the core concepts of CT.

The subjective nature of the definition of CT concepts, skills and abilitites also has a major influence on the set of pedagogical approaches and assessment methods. Despite that, and confirming our expectations, programming courses are the most popular pedagogical approaches. However, this seems to be a consequence of the fact that most studies are either performed by computer scientists or within the context of promoting Computer Science itself. There are, however, different approaches that are not based on programming. In fact, such alternatives are justified and desirable. Since Wing's seminal paper, most researchers accept that CT is about "conceptualizing, not programming". Thus, finding ways to promote CT without the need to explicitly depend on teaching computer programming, be it within visual frameworks or not, is an important contribution to help promote CT more effectively. Unfortunately, we still need to further propose and investigate alternatives.

The broad definition of CT also collaborates to actively encourage it to different audiences: kindergarten, K-12, undergraduate students and teachers. This introduces further challenges with respect to assessment. For each audience, different pedagogical approaches, artifacts and assessing methods are necessary. In particular, a major concern is that teachers have to master both the pedagogical approach as well as methods to assess the development of their students.

Finally, we must comment on the methodological aspects of the papers considered in this study. Perhaps because the subject is difficult and complex, most studies presented simply cannot be replicated. Either because there is no clear presentation of the conditions under which the study occurs, or because there is no precise characterization of the methods used to assess and evaluate the results. We know these studies are difficult, and perhaps impossible, to make them fully replicable, but we believe the whole community of researchers could benefit from a more careful description of the study design, procedures and assessment methods, as well as the general conditions necessary to conduct a similar study. We believe, that the CT research area needs to improve on the research methods adopted to better support claims on the relevance and effectiveness of CT.

REFERENCES

[1] J. M. Wing, "Computational thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33–35, 2006.

[2] C. Hu, "Computational thinking: what it might mean and what we might do about it," in *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*. ACM, 2011, pp. 223–227.

[3] C. Selby and J. Woollard, "Computational thinking: the developing definition," 2013, [Accessed January 20 2016]. [Online]. Available: http://eprints.soton.ac.uk/356481/

[4] S. Grover and R. Pea, "Computational thinking in k–12 a review of the state of the field," *Educational Researcher*, vol. 42, no. 1, pp. 38–43, 2013.

[5] J. Voogt, P. Fisser, J. Good, P. Mishra, and A. Yadav, "Computational thinking in compulsory education: Towards an agenda for research and practice," *Education and Information Technologies*, vol. 20, no. 4, pp. 715–728, 2015.

[6] N. R. Council, *Report of a Workshop on the Pedagogical Aspects of Computational Thinking*. Washington, DC: The National Academies Press, 2011, [Accessed January 15 2016]. [Online]. Available: http://www.nap.edu/catalog/13170/report-of-a-workshop-on-the-pedagogical-aspects-of-computational-thinking

[7] K. Petersen, R. Feldt, S. Mujtaba, and M. Mattsson, "Systematic mapping studies in software engineering," in *12th international conference on evaluation and assessment in software engineering*, vol. 17, no. 1. sn, 2008, pp. 1–10.

[8] T. Barcelos, R. Muñoz, R. V. Acevedo, and I. F. Silveira, "Relações entre o pensamento computacional e a matemática: uma revisão sistemática da literatura," in *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*, vol. 4, no. 1, 2015, p. 1369.

[9] J. Bombasar, A. Raabe, E. M. de Miranda, and R. Santiago, "Ferramentas para o ensino-aprendizagem do pensamento computacional: onde está alan turing?" in *Anais do Simpósio Brasileiro de Informática na Educação*, vol. 26, no. 1, 2015, p. 81.

[10] S. Y. Lye and J. H. L. Koh, "Review on teaching and learning of computational thinking through programming: What is next for k-12?" *Computers in Human Behavior*, vol. 41, pp. 51–61, 2014.

[11] S. Grover, S. Cooper, and R. Pea, "Assessing computational learning in k-12," in *Proceedings of the 2014 Conference on Innovation ; Technology in Computer Science Education*, ser. ITiCSE '14. New York, NY, USA: ACM, 2014, pp. 57–62.

[12] A. Yadav, C. Mayfield, N. Zhou, S. Hambrusch, and J. T. Korb, "Computational thinking in elementary and secondary teacher education," *ACM Transactions on Computing Education (TOCE)*, vol. 14, no. 1, p. 5, 2014.

[13] H. Bort and D. Brylow, "Cs4impact: measuring computational thinking concepts present in cs4hs participant lesson plans," in *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 2013, pp. 427–432.

[14] C. S. Miller, L. Perković, and A. Settle, "File references, trees, and computational thinking," in *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*. ACM, 2010, pp. 132–136.

[15] C. Kazimoglu, M. Kiernan, L. Bacon, and L. MacKinnon, "Learning programming at the computational thinking level via digital game-play," *Procedia Computer Science*, vol. 9, pp. 522–531, 2012.

[16] K.-C. Yeh, Y. Xie, and F. Ke, "Teaching computational thinking to non-computing majors using spreadsheet functions," in *Frontiers in Education Conference (FIE), 2011*. IEEE, 2011, pp. F3J–1.

[17] L. Werner, J. Denner, S. Campe, and D. C. Kawamoto, "The fairy performance assessment: measuring computational thinking in middle school," in *Proceedings of the 43rd ACM technical symposium on Computer Science Education*. ACM, 2012, pp. 215–220.

[18] M. Sherman and F. Martin, "The assessment of mobile computational thinking," *Journal of Computing Sciences in Colleges*, vol. 30, no. 6, pp. 53–59, 2015.

[19] D. C. Webb, "Troubleshooting assessment: an authentic problem solving activity for it education," *Procedia-Social and Behavioral Sciences*, vol. 9, pp. 903–907, 2010.

[20] D. Giordano and F. Maiorana, "Use of cutting edge educational tools for an initial programming course," in *Global Engineering Education Conference (EDUCON), 2014 IEEE*. IEEE, 2014, pp. 556–563.

[21] L. Seiter, "Using solo to classify the programming responses of primary grade students," in *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*. ACM, 2015, pp. 540–545.

[22] S. M. Pulimood, K. Pearson, and D. C. Bates, "A study on the impact of multidisciplinary collaboration on computational thinking," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. ACM, 2016, pp. 30–35.

[23] C. Duncan and T. Bell, "A pilot computer science and programming course for primary school students," in *Proceedings of the Workshop in Primary and Secondary Computing Education*. ACM, 2015, pp. 39–48.

[24] I. Zur-Bargury, B. Pârv, and D. Lanzberg, "A nationwide exam as a tool for improving a new curriculum," in *Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education*, ser. ITiCSE '13. New York, NY, USA: ACM, 2013, pp. 267–272. [Online]. Available: http://doi.acm.org/10.1145/2462476.2462479

[25] S. Jun, S. Han, H. Kim, and W. Lee, "Assessing the computational literacy of elementary students on a national level in korea," *Educational Assessment, Evaluation and Accountability*, vol. 26, no. 4, pp. 319–332, 2014.

[26] L. Gouws, K. Bradshaw, and P. Wentworth, "First year student performance in a test for computational thinking," in *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*. ACM, 2013, pp. 271–277.

[27] P. Hubwieser and A. Muhling, "Investigating the psychometric structure of bebras contest: Towards mesuring computational thinking skills," in *Learning and Teaching in Computing and Engineering (LaTiCE), 2015 International Conference on*. IEEE, 2015, pp. 62–69.

[28] M. Csernoch, P. Biró, J. Máth, and K. Abari, "Testing algorithmic skills in traditional and non-traditional programming environments," *Informatics in Education*, vol. 14, no. 2, pp. 175–197, 2015.

[29] J. A. Joines, D. Raubenheimer, and A. Craig, "Using computational tools to enhance problem solving," *Computers in Education Journal*, vol. 20, no. 4, pp. 101–112, 2010.

[30] L. A. Gouws, K. Bradshaw, and P. Wentworth, "Computational thinking in educational activities: an evaluation of the educational game lightbot," in *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*. ACM, 2013, pp. 10–15.

[31] L. Seiter and B. Foreman, "Modeling the learning progressions of computational thinking of primary grade students," in *Proceedings of the ninth annual international ACM conference on International computing education research*. ACM, 2013, pp. 59–66.

[32] K. Brennan and M. Resnick, "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.

[33] M. U. Bers, "The tangiblek robotics program: Applied computational thinking for young children." *Early Childhood Research & Practice*, vol. 12, no. 2, p. n2, 2010.

[34] B. Zhong, Q. Wang, J. Chen, and Y. Li, "An exploration of three-dimensional integrated assessment for computational thinking," *Journal of Educational Computing Research*, p. 0735633115608444, 2015.

[35] A. P. Ambrosio, C. Xavier, and F. Georges, "Digital ink for cognitive assessment of computational thinking," in *Frontiers in Education Conference (FIE), 2014 IEEE*. IEEE, 2014, pp. 1–7.

[36] D. Giordano, F. Maiorana, A. P. Csizmadia, S. Marsden, C. Riedesel, S. Mishra, and L. Vinikienė, "New horizons in the assessment of computer science at school and beyond: Leveraging on the viva platform," in *Proceedings of the 2015 ITiCSE on Working Group Reports*. ACM, 2015, pp. 117–147.

[37] M. Akcaoglu, "Learning problem-solving through making games at the game design and learning summer program," *Educational Technology Research and Development*, vol. 62, no. 5, pp. 583–600, 2014.

[38] J. M. Wing, "Computational thinking and thinking about computing," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1881, pp. 3717–3725, 2008.

[39] V. Barr and C. Stephenson, "Bringing computational thinking to k-12: what is involved and what is the role of the computer science education community?" *Acm Inroads*, vol. 2, no. 1, pp. 48–54, 2011.