Assessment of Computational Thinking

Nikolina BUBICA1, Ivica BOLJAT2
1 Mokosica Middle School Dubrovnik, Croatia
2 Faculty of Natural Science Split, Croatia
nikolina.bubica@skole.hr, boljat@pmfst.hr

ABSTRACT
With the new CS Curriculum in the Republic of Croatia, Computational thinking (CT) has finally been introduced in the educational process. In addition to the benefits that CT concepts bring to CS education, the question of evaluating CT and programming learning outcomes is also opening. The purpose of this paper is to present a model of evaluation of CT concepts based on the learning outcomes of the Croatian CS Curriculum using the Evidence-center design approach. The model is independent of the programming tool or environment and is intended for use with students who are CS novices.

KEYWORDS
Computational thinking, evaluation, programming novices, evidence-center design.

1. INTRODUCTION
New trends in technology development have a great impact on our daily lives. Technology enters the fabric of our lives regardless of the occupation area, but also regardless of the age of the user. We hear more demands for changes in K12 education. Also, regardless of the type of technology students use and the occupations they are being educated for, they are increasingly expected to possess some generic competencies such as ability to solve problems in everyday life, disaggregate complex problems to simpler ones, generalize solutions, etc. The fundamental question today is how to respond to such challenges. Leaders of CS education increasingly emphasize the need to modify existing CS curricula and to include the development of these competencies. Jeannette Wing (Wing J. M., 2006) points out that besides the standard types of literacy, such as mathematical, engineering and reading literacy, students are expected to have the ability to solve problems. She defines CT as “…the process of formulating problems and their solutions, but in ways that solutions are presented in a form that enables them to perform effectively with some information processing agent ” (Wing J. M., 2010).

2. COMPUTATIONAL THINKING
There is still a lot of confusion over the very definition of the concept of computational thinking, and many surrounding questions and challenges need to be addressed. It is considered to be the universal competence of every child that would, together with analytical skills, be the foundation for each child’s school learning (Wing J. M., 2006). Denning (Denning P. J., 2009) discusses whether CT belongs exclusively to the field of CS. Guzdial (Guzdial, 2008) describes CT like a 21st century literacy that is necessary to a whole series of faculties. It is often discussed how CT differs from algorithmic thinking, and Denning adds that “… CT means interpreting the problem as an information process for which we are then trying to find an algorithmic solution” (Denning P., 2010). To create an operational definition of CT, the ISTE and CSTA organizations analyzed feedback from about 700 surveyed teachers, scientists and CS researchers. The result was formulated in the operational definition of CT for K12 education as a problem-solving process which includes formulating problems, logically organizing, analyzing and representing data with abstractions, automating solutions through algorithmic thinking and generalizing the problem-solving process (ISTE & CSTA, 2011). When talking about teaching and learning CT, perhaps the most interesting is the role of programming. How much programming, if any, is needed to adopt CT? There is no unique answer, but practice points to different levels of programming involvement. Some define CT as a fundamental ubiquitous problem-solving tool and suggest several activities and projects which address CT (Astrachan, Hambrusch, Peckham, & Settle, 2009). Other approaches suggest various ways of incorporating programming into teaching and learning of CT, from those in which programming is the fundamental CT skill to those that integrate CT through various general education courses.

3. CT IN THE CROATIAN PROPOSAL OF CS CURRICULUM
In May 2016, Croatian Ministry of Education published Proposal of CS Curriculum for K12 education. The proposal was a promising hope for CS teachers since most of them were restrained by the old and outdated curriculums. Moreover, CS curriculum proposal finally accepted CT to be a significant part of the CS education in general. Croatian curriculum subject domains are e-Society, Digital literacy and Communication, Information and Digital technology and CT and Programming (Brođanac, et al., 2016). The role of CT and programming domain in CS Curriculum aims to make students to be involved in logical thinking, modeling, abstracting and problem-solving because solid ICT education, based on CT and creativity, should enable understanding and alteration of the world around us (Brođanac, et al., 2016). CT learning outcomes are created from the beginning of primary education starting with elementary pupils, age 6-7, through middle school pupils, age 11-14, and finally high school students, age 15-18 (Brođanac, et al., 2016)

4. HOW TO ASSESS CT?
Everyone agrees that learning programming is hard, but it seems that evaluating new knowledge through evaluating new definitions and programming commands is far simpler than evaluating the way students apply computing and programming language to solve problems and to design different computer work. To assess CT, it is necessary to find evidence of a deeper understanding of the problem
solved by the student or to find evidence of understanding how the student created his coded solution. Since CT concepts include, for example, abstraction (ISTE & CSTA, 2011), it means that we must find ways to evaluate how student applied abstraction in his solution while trying to solve a problem. As there is very little agreement about the CT definition, it is even less known about the tools for assessing such thinking. However, there are approaches for evaluating the development of CT that are currently in use or are still in development. They could serve as a solid foundation for developing a general approach for evaluating CT. Brennan and Resnick propose a valuation method that includes project portfolio analysis, document-based interviews, and development of design scenarios (Brennan & Resnick, 2012). Such approach estimates the fluidity of computer-based practice of testing and debugging, experimenting and repetition, abstraction and modulation, and reusing and remixing/scaling. Expertise is assessed through three levels: low, medium and high. The evaluation approach of student's documentation consists of building creative projects from students but also of creating visible traces of their work on the project. Such traces could be achieved in the form of paper or digital diaries. Also, it could be achieved by using Scratch's commentary capabilities for explaining some project's features and screen views that will graphically present the project, its intent or the main advantages and disadvantages. Still, there is not enough research data to validate this approach. Dorling and Walker specifically study the practice of teaching CT in the classroom environment and propose a framework for evaluating the Computing Progression Pathway that recognizes the major areas of CS and offers specific levels of adoption (Dorling, 2014). Within the PACT project (Principled Assessment of CT) general CS practice is represented through some design patterns which emphasize application and reviewing of design skill while solving the computational problem rather than evaluating the knowledge of the concepts necessary to apply such skills (Bienkowski, Snow, Rutstein, & Grover, 2015). This approach is based on Evidence-centered design (ECD) (Hendrickson, Ewing, Kaliski, & Huff, April, 2013) for creating a structured description of the domain evidence argument and highlights knowledge and skills complexity or other features or behaviors that should be valued. The ECD approach is usually represented through five layers: domain analysis, domain modeling, conceptual evaluation framework, evaluation application and delivery. SRI Education group, within the PACT project, proposed application modes for every layer to create the practice of CT assessment. Also, it is possible to find several published computer-based or paper-pencil tests that differ in context, intended for the age of those who are important in testing and reevaluating (Werner, Denner, & Campe, 2012). This paper offers a framework for assessing CT demonstrated on Croatian Learning Outcomes of CT and Programming Domain based on ECD and PACT evaluation proposal.

5. PROPOSAL OF CT ASSESSMENT

Despite the advantages of introducing CT into the new curriculum, we can’t ignore possible difficulties and new problems that arise from this new approach to teaching CS. Evaluation of CT becomes a new challenge in the present CS educational work and requires a more serious approach than finding individual solutions by teachers' practitioners. One proposal of CT evaluation will be presented in the next paragraphs. It uses ECD as an orientation towards multiple activities necessary to create useful documentation like domain analysis, domain modeling, construction of framework and assessment implementation and delivery (Mislevy & Harterel, 2006).

5.1. Domain analysis

Appropriate pedagogical practice, emphasizing the constructivist approach to learning and putting students at the heart of the learning process, should develop the competencies like independence, self-confidence, responsibility, and entrepreneurship. CS curriculum created according to the learning outcomes instead of the prescribed content, enables the realization of learning and teaching directed at each student level and the development of his or her potential. It provides flexibility and gives freedom to the teachers in designing the learning and teaching process. The basic goal of the domain analysis layer is to find and explore all relevant materials concerning the target learning outcomes. In this article, we will use the sixth-grade CT learning outcomes, student age 11-12 (http://bit.ly/2018cte, Table 1). These learning outcomes stem from several documents but mostly Croatian National Educational Standards, CS Teacher Standards and Proposal of Croatian CS Curriculum. Croatian National Educational Standards defines the way in which CS is involved in Croatian primary, secondary and higher education. Croatian CS Curriculum and CS Teacher Standards defines CS learning outcomes at each educational level with its adoption level specification. Every learning outcome is expressed in detail within Bloom taxonomy, through different adoption levels: satisfactory, good, very good and exceptional level (http://bit.ly/2018cte, Table 2). These learning outcomes are a basis for our assessment process. In following sections, we will try to identify more design patterns that will help us create appropriate evaluation.

5.2. Domain modeling

Domain modeling has the task to identify elements for describing the domain we want to evaluate. According to ECD approach, Domain modeling is organized into five categories: fundamental and additional knowledge, skills and features, possible working products, variable feature and possible observations (Bienkowski, Snow, Rutstein, & Grover, 2015). An example of domain modeling for CT sixth-grade learning outcomes can be found on author's personal page (http://bit.ly/2018cte, Table 3).

5.3. Assessment framework

CT evaluation is highly dependent on the context within which the evaluation is performed. Is it necessary to conduct CT assessment using some programming tool or environment? The question of the connection between CT and programming must be defined regarding the context of the applied evaluation. There are different approaches to incorporating programming into the process of teaching and thus the process of CT assessment. We differentiate them according to the role of programming and CT in the course curriculum (Astrachan, Hambrusch, Peckham, & Settle, 2009). In this paper, assessment of CT is achieved through
the approach that is not dependent on the programming tool or environment. This approach could serve for evaluation of adopted learning outcomes in real classroom situations at some stage of education. Precisely, the independence of the programming tool or environment enables wider application of the evaluation tool and highlights the concepts of evaluation rather than the syntax of a programming tool or environment possibilities. For the same reason, such a tool could be used with students that have no programming background. According to ECD (Hendrickson, Ewing, Kaliski, & Huff, April, 2013), evaluation framework aims to assist assessment designers while they validate their task model. Every assessment designer should validate his work with questions regarding construct relevance, specificity, and scalability and questions related to item statistics and item complexity. This evaluation framework should provide information about evidence, students model and task model, observable characteristics, measurement models and test specifications. For testing this model of evaluation, a similar measuring instrument adapted to Python programming language was conducted during 2016/2017 school year. Evaluation instrument was applied after 12 weeks (6th grade) or 14 weeks (7th grade) of learning and teaching process on a sample of 15 students of 6th grade (8 female) or 10 students of 7th grade (3 female). The positive and promising results of probe evaluation encouraged the creation of this evaluation model, independent of the programming tool and the programming environment.

Model of students

Given that the evaluation is intended for use in middle and secondary schools in the Republic of Croatia where there is a big diversity in applied programming tools and languages, an evaluation that is not dependent on the programming tool could be widely applicable. Programming tool or environment independence emphasizes on CT concepts rather than the ability to work with specific tool or environment. Also, if it is crucial for the actual CS curriculum to use certain programming tool or an environment, these tasks could be easily customized and constructed in it.

Model of tasks

Evaluation tasks are created for students with little or no programming knowledge. Each represents one puzzle used to help the main character in solving problems. Puzzles are supposed to assess one or more CT concepts. CT concepts, concealed in puzzles, have been selected and aligned with the expected learning outcomes (Brodatuc, et al., 2016) and detailed domain analysis (http://bit.ly/2018cte, Fig. 2). Assessment tool should be implemented in the form of online knowledge test consisted of 10 questions. The types of questions that will appear in the evaluation tool are: multiple choice questions (mostly used for identification of some fundamental misconceptions or unsustainable mental model), short answer questions; essay questions (used for student’s authentic algorithmic solutions). Feedback for multiple choice questions should be defined automatically, while short answer questions and essay questions should be manually evaluated by the researcher or teacher.

Model of evidence

Design and application of high-quality assessment are very demanding and also time-consuming. According to ECD approach (Hendrickson, Ewing, Kaliski, & Huff, April, 2013) our assumptions and hypothesis represent evidence about the way student’s abilities are represented in his work. Such evidence should reveal student’s adoption of learning outcomes. Each algorithm solution is always difficult to evaluate automatically. Evidence analysis helps us in creation of evidence model for similar tasks. While analyzing possible student’s answers, it is crucial to know which computational concepts are evaluated with the default task (http://bit.ly/2018cte, Table 4). Evidence of student work varies from the situation where the student doesn’t even try to do anything, further through several partial solutions and finally to a fully correct solution (http://bit.ly/2018cte, Table 5).

Model of measurement

To complete domain analysis and modeling, it is necessary to define the model of measurement. For the task example in Fig. 1, the possible evidence is presented according to its complexity. If the student does not offer any response or his answer has no links to the task itself, such answer should be rewarded with zero points. With each of the following evidence, it has been recognized a higher level of adoption from the previous one (http://bit.ly/2018cte, Table 5).

5.4. Assessment implementation/delivery

The realization of the test assessment, adopted for Python programming tool and conducted during the 2016/2017 school year, was performed as online assessment within Loomen Learning Management System (LMS). The assessment consisted of eight tasks (one pairing task, four multiple choice tasks, three essay tasks) and was conducted during a 45-minute regular school class. The students showed great satisfaction by conducting online assessment instead of standard paper-pen assessment even though it was their first real encounter with such form of evaluation. The assessment task discrimination analysis showed that as many as six tasks proved to be excellent (task discrimination

![Figure 1: Example of essay task question](http://bit.ly/2018cte)
index > 0.35) while two of them were discarded from further modeling due to the negative index of discrimination. As for the task difficulty, two of them have proved to be extremely simple (0.93), but they have already been dismissed from the further modeling because of their extremely low discrimination. Two tasks had recommended difficulty (0.5-0.6) and four task acceptable difficulty index (0.3-0.7). Further application of the assessment tool will be used to test the validity and reliability of the measuring instrument and will help in creating this new CT assessment model. CT assessment model proposed in this paper will also be organized in the form of online testing within the Loomen LMS. Students’ access to Loomen LMS will be enabled through their unique user data, provided to every middle and secondary student in the Republic of Croatia. In that way, the authenticity of the research participants’ data will be preserved. In the phase of pilot research, it is expected the involvement of 50-60 students with the purpose of testing the clarity of task texts and detecting potential ambiguities or some other problems. Also, several CS teachers will be invited to evaluate the assessment tool as valuable practitioners with attention on measurement model. After defining the final version, the assessment tool will be applied to as many 11-12 year old students as possible who are just encountering fundamental concepts of CS. In addition to the evaluation tool, the students will be previously asked to fill out a questionnaire that aims to collect some personal information interesting to the research like general data such as gender, general academic achievement or some data related to programming knowledge. Also, evaluation tool will be applied even with some number of high school students. The results of the research should reveal the power of the tool itself, but also could explore if there is a difference in the results among participants who have some programming experience from those who have none, and further investigate whether there are differences in gender related to results and so on.

6. CONCLUSION

Many teachers are increasingly emphasizing the need for a stronger involvement of the CT concepts in CS courses, but it is also noticed within some other sciences such as biology, physics, mathematics, chemistry (Interdisciplinary Computational Thinking, 2017). The purpose of this paper was to present one approach to the assessing CT adapted to the actual classroom situation. The proposed assessment tool was developed knowing that there are several programming tools and environments used in CS education in the Republic of Croatia, but also accepting the fact that CS is an elective subject in elementary/middle schools where programming is only minor part of the subject curriculum. The new CS curriculum proposal introduces the concept of computational thinking, and thus opens the question of evaluating its concepts. The proposed evaluation model is based on defined learning outcomes from the CT and programming domain of the new CS curriculum proposal and offers the possibility of assessing CT concepts independently of applied programming tools and environments in the teaching process. Also, it could serve as the basis for making similar assessment tools. The real power of the tool, its validity, and reliability, but also its weaknesses will be able to reveal through its application, which is our next step.

7. REFERENCE