

## Enhancing Computational Thinking Skills for New Mexico Schools

### Dr. Alfredo J. Perez, Northern New Mexico College

Alfredo J. Perez received his M.S. degree in Computer Science and Ph.D. degree in Computer Science and Engineering from the University of South Florida, in 2009 and 2011 respectively. Since 2011, he has been with Northern New Mexico College, Espanola (NM), where he is currently an Assistant Professor in the Department of Engineering. Prior to coming to USA to pursue graduate studies, he obtained a B.S degree in Systems Engineering from Universidad del Norte in Barranquilla Colombia (2006). His research interests are in the areas of Mobile Computing/Sensing, Data mining, Distributed Systems and STEM education. He has coauthored several journal and conference papers as well as the book "Location Aware Information systems - Developing Real-time Tracking Systems", published by CRC Press.

### Dr. Ivan Lopez Hurtado, Northern New Mexico College

IVAN LOPEZ HURTADO received his B.S. degree in Industrial Physics Engineering from Tec de Monterrey, Monterrey, Mexico, 1995. M.S. degree in Automation from Tec de Monterrey, Monterrey, Mexico, 1998 and Ph.D. in Electrical Engineering from the University of New Mexico, Albuquerque, NM, USA in 2008. He is currently the Department of Engineering, Chair at Northern New Mexico College.

### Dr. Jorge Crichigno, Northern New Mexico College

Jorge Crichigno received the BSc degree in Electrical Engineering from the Catholic University of Asuncion, Paraguay, in 2004, the MSc and PhD degrees in Computer Engineering from the University of New Mexico, Albuquerque, NM, in 2008 and 2009 respectively. In 2007, he was visiting the Wireless Sensor Network Lab in the School of Electronic, Information and Electrical Engineering at Shanghai Jiao Tong University. His research interests include wireless and optical networks, graph theory, mathematical optimization, network security and undergraduate STEM education. He has served as reviewer and TPC member of IEEE journals and conferences and as panelist for NSF STEM undergraduate education initiatives.

### Mr. Raul R Peralta, Northern New Mexico College

Raul Peralta-Meza received the B.Sc. degree in Electronic Engineering from San Agustin National University in Arequipa, Peru, in 1993. He obtained a M.Sc. degree in Computer Science at the Pontifical Catholic University of Peru in 2000 and a M.Sc. degree in Computer Engineering from The University of New Mexico in 2007. Currently, he is a full-time instructor in the Department of Engineering at Northern New Mexico College, Espanola, New Mexico.

### Dr. David Torres, Northern New Mexico College

Dr. David Torres is an Associate Professor of Mathematics at Northern New Mexico College. His research interests include computational fluid dynamics, parallel programming and numerical algorithms.

# Enhancing Computational Thinking Skills for New Mexico Schools

Computational Thinking is a term to group skills for the utilization of computers as problem solving tools that improves college readiness and increases K-12 students' likelihood of attending and graduating from college. It differentiates from Digital Literacy (or Computer Literacy) where a person acquires skills for using the computer for everyday use. Statistics about the percentage of bachelor's degrees in computer science/engineering earned in the United States during the last twenty years shows that less than 20 percent of graduates account for underrepresented minorities which place this population in great disadvantage with other ethnic groups. Given that New Mexico is a state with high percentage of underrepresented minorities, the Department of Engineering at Northern New Mexico College through the support of Google Inc. and the National Science Foundation has organized and developed Computational Thinking (CT) workshops and activities for K-12 teachers and pre-service teachers during the last two years as an approach to increase the awareness of Computer Science among K-12 students in New Mexico. The curriculum developed in the workshops have provided teachers and pre-service teachers a variety of tools to incorporate and enhance their classes in K-12 schools through the utilization of computational thinking activities. The approach utilized for the workshops has not only enhanced CT skills but also has provided ideas to develop lesson plans and activities for the Common Core State Standards. The present paper presents a summary of the activities developed for the workshops as well as results that the participants have shared on the utilization of the acquired skills in their classrooms.

**Keywords:** Computational Thinking, K12 Engineering Education, Computer Science Education, Teacher Preparation, Curriculum Development

## 1. Introduction

The increasing market and demand of computer professionals that have been forecasted by 2018<sup>1</sup> has made almost a priority to prepare the population of the USA to work as computer professionals. The report has stated that there will be about three job opportunities per job applicant in computer-related careers. Not meeting such demand can make USA to be in strategic disadvantage with other economic powers of the world.

In addition, statistics about the percentage of bachelor's degrees in computer science<sup>2</sup> earned in the USA during the last twenty years shows that less than 20 percent of graduates account for underrepresented minorities which put this population in great disadvantage with other ethnic groups of the USA.

Given that New Mexico is a state with high percentage of underrepresented minorities, the Department of Engineering at Northern New Mexico College has started several initiatives to increase recruitment of students from this group through awareness of computer science-related careers to the public. One of these initiatives is the preparation of K12 teachers and pre-service teachers to incorporate computational tools into their teaching assignments as an approach to motivate K12 students from the State of New Mexico to choose Information Engineering Technology, Computer Science and Computer Engineering as an option for College. This paper

presents and summarizes our experience on developing Computational Thinking Workshops for K12 teachers and pre-service as an approach to enhance CT skills in New Mexico Schools.

## 2. Computational Thinking

Computational Thinking (CT)<sup>3</sup> is a term to group skills for the utilization of computers or any type of information processing agent as problem-solving tools. It differentiates from Digital Literacy<sup>4</sup> (or Computer Literacy) where a person acquires skills for the utilization of a computer for everyday use (e.g. browsing the Internet, typing a letter, preparing slideshows, using a spreadsheet, etc...). As defined by Wing<sup>3</sup>, CT encompasses more than programming computers or a set of skills only for future computer professionals: since CT defines a problem-solving methodology carried out by information processing agents, the concept is applicable to any type of organization, machine and/or living organism that can perform computation.

The concept of computation can be described as the process of transforming input data into output data to solve a particular problem. The transformation is usually described in terms of an algorithm which is a set of steps to accomplish a task. An analogy between computation and cooking is depicted in figure 1. This figure compares the process of baking a cake (output) from the ingredients (input) by using the recipe (algorithm). The chef and the oven (processing agents) produce the transformation of the ingredients into a cake as described by the recipe. In comparison, indexing the web follows a similar type of transformation: analyzing web pages and relations between webpages (input), an indexing algorithm/procedure, a data center/cluster computer (processing agent) and the final index of web pages for a given keyword (output).



**Figure 1.** An analogy between Cooking and Computing

## 2.1 Computational Thinking Skills

Activities in computational thinking are usually organized into four major skills which are *Decomposition*, *Pattern Recognition*, *Pattern Generalization and Abstraction*, and *Algorithm Design* <sup>5</sup>.

### A. Decomposition

Decomposition is the ability to separate/divide an activity or task in smaller activities/task in such a way that the initial activity can be explained/described as a process to an information processing agent. An example of this activity is to decompose the steps taken to go from a place to another (e.g. Leave house, make right, go straight for  $x$  miles, turn left, go straight  $y$  miles, arrive).

### B. Pattern Recognition

Pattern Recognition is the ability to notice common characteristics or differences that will create shortcuts or make predictions. For example, suppose the sequence of numbers: 1, 4, 10, 19, and 31. Can you describe the pattern? What would be the next number in the sequence? If correctly identified, the next number in the sequence should be 46. Even though this is a simple sequence, many algorithms and processes take into account some type of pattern recognition in order to formalize what is happening.

### C. Pattern Generalization and Abstraction

Once the pattern is discovered, the next step is the generalization and abstraction of the pattern. The skill encompasses the ability to filter information to solve a problem and re-defining it in general terms using variables and/or formulas so problems that are similar in nature can be solved in the same way. For the example shown above (the sequence of numbers) the pattern generalization and abstraction step comprises taking the sequence and converting it into a recurrence relation. The formula that abstracts the sequence can be written as  $S(i) = S(i-1) + 3 * i$  where  $i$  is a positive integer and  $S(0) = 1$ .

### D. Algorithm Design

The last skill in Computational Thinking is the development and/or the description of the solution of a problem as a recipe/algorithm. Algorithm Design is based on the three skills described above which help to solve a problem using abstractions and the transformation such abstractions as described by the algorithm. When the algorithm is described through a computer language, then the problem can be solved efficiently by a computer. An example of an algorithm that generates the sequence of numbers above is depicted in figure 1. In the figure, the algorithm is written in the Python computer language.

### Original Sequence

1  
4  
10  
19  
31  
46



### Algorithm that Generates Sequence

```
def generateNextNumber(num):  
    prev = 1  
    for i in range(num):  
        nextNumSeq = prev + 3*i  
        prev = nextNumSeq  
    return prev  
  
for i in range(1,7,1):  
    print generateNextNumber(i)
```

**Figure 2.** A sequence of numbers and a Python code that generates the sequence

## 2.2 Learning Computational Thinking

Lee *et al.*<sup>6</sup> describe the learning of computational thinking in youth through the utilization of modeling and simulations, robotics, and game design and development as an effective way to learn CT through a framework of three progressing phases. In the first phase, called ‘USE’, the students reutilize CT activities already created by somebody else (i.e. playing a game already developed, using a program that controls a robot). After students feel comfortable, then they move forward to the second phase called ‘MODIFY’, on which students begin to modify already developed code by changing small pieces/parts of it (i.e. changing the looks of an avatar in a game, changing the behavior of the robot). As students begin to further improve their skills and feel even more comfortable, they begin developing their own designs and ideas. This phase is called ‘CREATE’ on which the students create their own projects.

As described by Lee *et al.*<sup>6</sup>, the goals of these phases are to learn three major terms called abstraction, automation and analysis that the authors have found as an effective way to describe CT to youth; however, issues and challenges such as the lack of curriculum standards, infrastructure and lack of opportunities for teachers to learn CT make it difficult to implement CT in day-to-day classrooms activities<sup>6</sup>. As such, many programs have been focalized in implementing after hours CT activities. An example of such projects is the Project GUTS (Growing Up Thinking Scientifically)<sup>7</sup>.

With the current implementation of the Common Core Standards (in particular the Mathematical Practice Standards)<sup>8</sup>, there is the need to seek the development of skills that make use of mathematics to solve real-life, everyday problems. The math practice standards are eight outcomes that a student should acquire during K12 and they are related to the CT skills mentioned in the section 2.1 of this paper. Seen in this direction, CT activities can be used to teach concepts for the standards, hence allowing the incorporation and learning of CT in everyday classroom activities. Nevertheless, this requires the preparation of the teachers in CT and guiding of them on how to integrate CT activities into activities for the common core.

## 3. Workshop Development

Through the sponsorship of Google’s Computer Science for High Schools (CS4HS) program, the Department of Engineering organized and developed CT workshops for K12 teachers during the

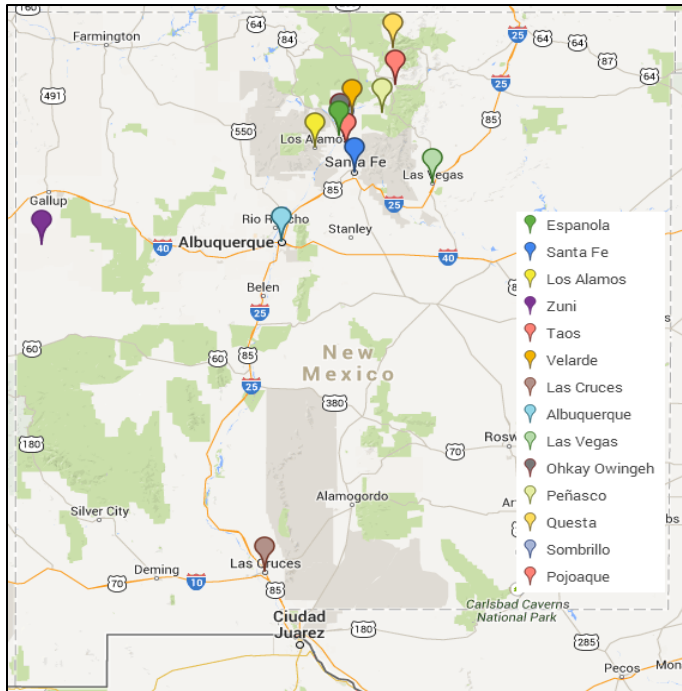
years 2012 and 2013. In these workshops, the participants were introduced to CT through different types of activities with the goal of the integration of these activities into classroom activities. The workshops were held for three days (2012) and two days (2013) with follow-up discussions. In addition, the workshops were presented in collaboration with the Department of Mathematics to pre-service STEM teachers through the NSF-sponsored Noyce Program at the College. NSF also has sponsored the participation of Noyce scholars in the NM Super Computing Challenge Summer Teaching Institute, a teaching institute that focalizes in teaching agent-based simulation and programming.

### **3.1 Demographics and Recruitment**

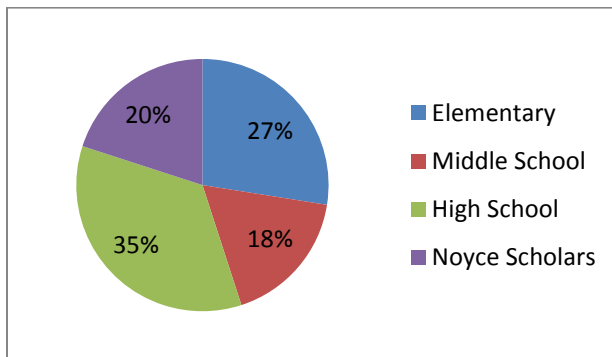
Northern New Mexico College's primary service area is Rio Arriba County, a sparsely populated rural area covering 5,860 square miles. The county has a population of 40,246 persons with only 6.9 people per square mile (less than half of the state ratio) and a per capita income of \$19,134<sup>9</sup> (the total estimated population of the state of New Mexico as of 2012 was 2.85 million with 42% living in the Albuquerque metropolitan area<sup>10</sup>), which makes this region to be a rural area. In addition, NNMC service area stretches to rural communities in southern Taos County and northern Santa Fe County. Within this region there are eight Native American Indian Pueblos (Taos, Ohkay Owingeh, Santa Clara, San Ildefonso, Tesuque, Nambe, Pojoaque, and Picuris) and one Indian Reservation, the Dulce Jicarilla Indian Reservation. The population demographics are as follows: 71.3% Hispanics (a population that resided in the area prior to the Mexican-American War); 16% are Native American, and 12.8% are Anglo based on the 2010 US Census.

Whereas the workshops recruited a total of 40 participants from different parts of the state of NM, most of them were in the service area of the College (see figure 3). From these 40 participants, 12 participated in the 2012 version of the workshops, 20 in the 2013 version, and 8 were Noyce pre-service scholars. Teachers were recruited through mailing lists such as the NM Super Computing Challenge's teacher email list<sup>11</sup> and through the recruitment office of the College. For the first year, only the mailing lists were used for recruitment. For the second year both the mailing lists and the College's recruitment office were used, with the second method proving to be better since after a week that the message was sent through the recruitment office, the spots for the participants were filled.

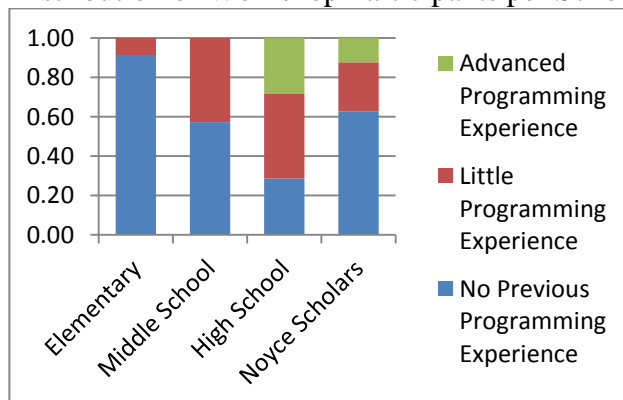
The distribution of teachers based on their school type is shown in figure 4 (a), as well as the normalized programming experience before the workshops per school type in figure 4(b). Participants were not restricted to math and science teachers: their teaching areas ranged from different subjects including social science, languages, special education, and arts. The participants were required to fill out an electronic application submitted through the workshops' website (figure 5). This website also contained the material delivered to the teachers. As incentives for participation, the participants in the 2012 version received Lego Mindstorm robots and LabView software. In the 2013 version, they received monetary incentives and Makey Makey human computer interaction hardware. Funding also supported lunch, coffee breaks, materials for participants, and stipends for the faculty, student helpers, and the administrative assistant in the Department of Engineering that prepared, delivered, and provided the workshop logistics.



**Figure 3.** Participant's School Location



(a) Distribution of Workshop Participants per School Type



(b) Programming Experience per School Type (before workshops, normalized values)

**Figure 4.** Demographics about Workshop Participants



**Figure 5.** Workshop Website (available at <http://tinyurl.com/nnmc-ct4nms> )

CT Activity	Example of Available Tools	Suggested School Level	Suggested Classes	CT Learning Phase
Kinematic Activities	CS-Unplugged	Elementary or greater	Arts, Math, Science	USE
Computer Games	Missing Mechanism, Light bot	Elementary or greater	Math, Technology	USE
Programming Languages	Scratch, LabView, Python	4th Grade or greater	Arts, Language, Science, Math, Technology	USE, MODIFY, CREATE
Robotics and Human Computer Interaction	Lego Robots, Makey Makey	Elementary (HCI), Middle School (robots)	Arts, Science, Math, Physics, Technology	USE, MODIFY, CREATE
Agent Models and Simulation	Netlogo, StarLogo TNG	Middle School/High School	Social Science, Biology, Ecology, Math, Technology	USE, MODIFY, CREATE

**Table 1.** Suggested Curriculum to teach CT in School Settings

### 3.3 Curriculum Development

In the workshops provided by the Department of Engineering, K12 teachers and pre-service teachers were exposed to the concepts mentioned in section 2 of this paper through hands-on activities using kinematic activities, computer games, programming languages, robotics and human-computer interaction interfaces, and agent-based programming and simulation. Most of the materials prepared for the workshops made use of free available software and material that can be obtained from the Internet. This section presents the materials and activities that were used in the development of the curriculum for the workshops which is summarized in table 1.



## A. Kinematic Activities and Computer Games

Kinematic activities teach Computational Thinking and Computer Science using pencil-and-paper activities and games/kinematic activities. These activities are useful to introduce CT and computer science concepts and also are useful when a school does not have a computer lab. In this part of the workshops, the participants were exposed to three activities from the CS-Unplugged book<sup>12</sup>. CS-Unplugged are a set of activities designed for elementary schools to teach computer science concepts, with translation to many languages. From this book, two encoding/decoding activities based on binary codes (teaching abstraction) and a graph theory activity based on the minimum spanning tree problem (teaching abstraction, pattern recognition and algorithmic design) were used to introduce this type of material to the teachers (see figure 6 (a)). This part of the workshop was performed in two hours.

In addition to kinematic activities, certain types of computer games are also useful to introduce CT concepts, in particular the concepts of decomposition and algorithmic design. Puzzle-based games, strategy games and agent-based simulation games, on which a player has to build contraptions, use some type of block-based programming, or develop strategies to achieve a goal are the types of games that can help to develop CT skills. Examples of these games are: the Missing Mechanism<sup>13</sup>, Amazing Alex<sup>14</sup>, Armadillo Run<sup>15</sup>, Light-bot<sup>16</sup>, SimCity<sup>17</sup>, or part of the introductory activities available under code.org initiative. These types of activities can be used in elementary and middle school levels. In addition, games such as Armadillo Run can be used in Physics classes to introduce concepts such as forces, friction, elasticity and others, hence mixing both physics concepts and CT.

In the workshops, participants were introduced in one hour to the Missing Mechanism (contraption game, figure 6 (b)) and Light-bot (block-based programming game). These two games are free and they are available online. Based on Lee's<sup>6</sup> learning phases, the kinematic activities and games classify under the 'USE' phase.

## B. Programming Languages

In this part of the workshops, participants were exposed to an introduction to programming languages using Scratch and Python (for the 2013 of the workshops, only Scratch was used based on the feedback from the 2012 version of the workshops). Scratch<sup>18</sup> is a graphical, block-based programming language where a programmer develops code by putting together code through figures/images that represent programming constructs (figure 6 (c)). The goal of block-based programming is to alleviate the typing of code which is how programming is performed in most of the programming languages such as Python<sup>19</sup>. This eliminates syntax errors which is one of the most annoying aspects in learning a programming language for beginners<sup>20</sup>.

Using Scratch, the participants were exposed to the three basic concepts that most of the programming languages have: variables, selection statements and loops. In addition, they were exposed to event-based programming and some constructs that are only available to Scratch. Due to the maturity of the language (it was developed in 2003), curriculum has been already developed to introduce Scratch in classroom activities such as the Creative Computing Curriculum available under the ScratchEd community<sup>21</sup>. ScratchEd's Creative Computing

curriculum exposes to CT to students through subjects such as languages and arts. A two-hour introductory workshop for Scratch was developed for the 2012 version of the program; for 2013 it was extended to four hours based on the feedback received.

In addition to Scratch, the 2012 version of the workshops included an introduction to the Python programming language. The approach utilized to introduce Python was to take one of the coding/decoding exercises utilized for the kinematic activities of the workshops and through a step-by-step approach, to develop the code that implemented the activity. This introduction was done in two hours, and the participants suggested extending or performing a full workshop only for Python. Since programming languages can be used in different ways, they can be used in all three of Lee's learning phases ('USE', 'MODIFY' and 'CREATE').

### **C. Robotics and Human-computer Interaction Interfaces**

Lego Mindstorms Robots<sup>22</sup> and Makey-Makey<sup>23</sup> devices were utilized as part of the workshops as an approach to show interaction of the real world with a computing device. In the 2012 version of the workshop, a full workshop day was used to introduce the participants to programming robots using National Instrument's LabView<sup>24</sup> programming environment. LabView is a block-based programming language similar to Scratch, but on which programs are developed using a programming paradigm called dataflow. In dataflow, computations are performed as data for an instruction becomes available. As the name suggests, the flow of data controls the execution of a program.

The focus of this part of the 2012 workshops was on learning how to utilize LabView to interact with the sensors and the NXT controller that are included in the Lego Mindstorms kit. Therefore, in this workshop day the participants focalized on the features of LabView and the dataflow programming model, and a final exercise where the participants developed a robot to demonstrate experimentally the concept of average velocity using the sonar sensor of the mindstorm kit (figure 6 (d) ), hence combining Physics and CT. Since robotics are more involved than kinematic/computer games/programming languages, using robotics as shown in the workshops are more suitable for middle/high school levels; however, they could be used in elementary levels as suggested by Lu *et. al.*<sup>25</sup>, and by using an easier to understand programming language than LabView. A drawback of Lego robot kits are their price as well as the cost of LabView license.

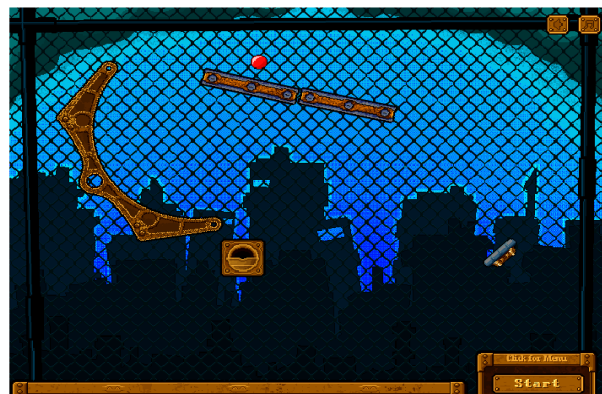
For the 2013 version of the workshops, instead of using the robots, participants were introduced to the Makey-Makey<sup>23</sup> human-computer interaction device. The device is a circuit board that looks like a bare bone joystick and connects through a USB port to a computer. Once connected to the computer, the device replaces certain keys in the keyboard of the computer (up/down/left/right arrows, space) and the click of the mouse. Using conductive materials, students can create surfaces that interact with the computer. In a follow-up meeting, the participants of the 2013 workshops were introduced to this device and they developed a computer interface using paper and pencil that interacted with a Scratch program (figure 6 (e)). Since the Makey - Makey is much simpler to utilize than the Lego robots, they can be introduced into elementary levels. Other advantages are its relative low price (around \$50) and that they can

interact with any computer program and/or language where the keyboard and the mouse can be used.

Robotics can be used at different levels of learning and abstraction; therefore, they can be utilized in the three phases of learning CT<sup>6</sup>. Since Makey – Makey device is a Human-computer interaction device it is more suitable for the ‘USE’ phase, however in combination with Scratch, it could be utilized for all of the phases.



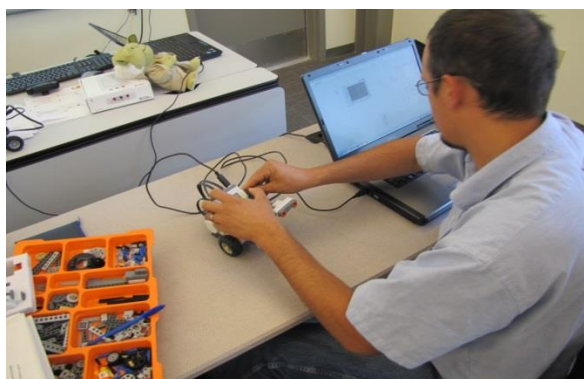
(a) CS Unplugged



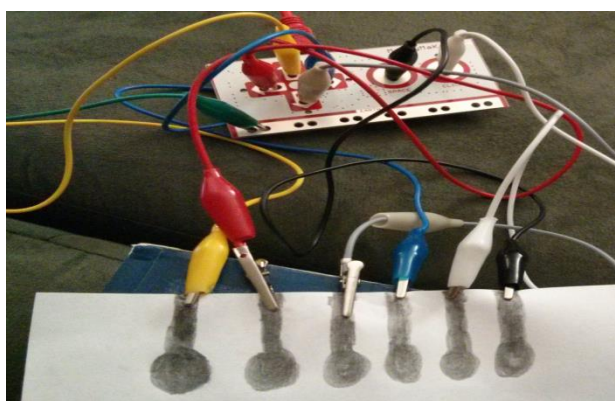
(b) The Missing Mechanism



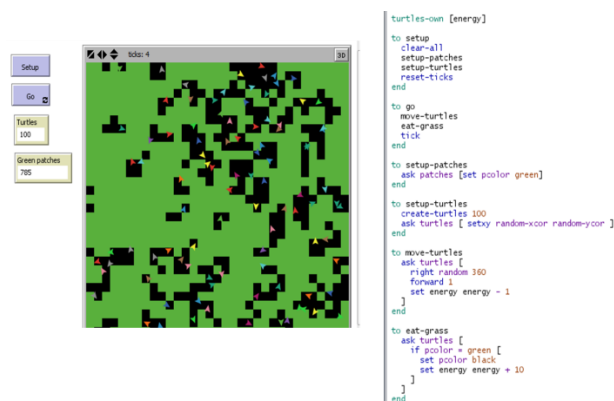
(c) Introduction to Scratch



(d) Lego Mindstorm Robots



(e) Makey-Makey HCI interface



(f) Netlogo

**Figure 6.** Computational Thinking Activities and Tools used in the Workshops

## **D. Agent-based Programming and Simulation**

In agent-based programming<sup>6</sup>, a programmer develops and implements computer models that mimic the behavior of a system through the interaction of autonomous entities called agents. The goal of this type of programming is to find effects and emerging behaviors in the system as a whole. Agent-based Programming is utilized to model and simulate social, economic, epidemiologic, and ecological systems.


Netlogo<sup>26</sup> (figure 6(f)) was utilized as the tool for this part of the workshop. Netlogo's programming environment is available online without cost and uses a dialect of the Logo programming language to implement the behaviors of the agents. In addition to the programming environment, Netlogo includes a complete library of models in different domains (e.g. math, science, biology, ecology, etc...) that a programmer could utilize as baseline models to develop complex systems.

In a complete workshop day (8 hours), teachers received basic training through step-by-step tutorials that taught the basic concepts of the Netlogo and agent-based programming. The workshop was divided into three parts: a first part on the introduction to Agent-based programming, a second part on the basics of Netlogo, and a third part on using the models already implemented in Netlogo to integrate them to classroom activities. A drawback of Netlogo is that it requires the typing of code, however, agent-based programming using block programming (i.e. Scratch) can be accomplished through StarLogoTNG<sup>27</sup>. Agent-based programming and simulation can be used on the three phases ('USE', 'MODIFY' and 'CREATE'), and due to the concepts involved in agent modeling, for the 'MODIFY' and 'CREATE' phases are suggested for grades 6<sup>th</sup> and onwards.

## **4. Results**

Immediate feedback was obtained from the participants through a survey that evaluated the participant's perception about the workshop material on the final day. Participants rated the workshops with an average of 4.7/5 (2012) and 4.5/5 (2013) when asked about the worthiness of the workshops and the material received. Participants also mentioned that workshops could be extended to more days in order to cover better the different topics presented. We recognize this comment as a positive feedback because it means that many of them were interested in expanding parts of the workshop.

After the workshops, follow-up was performed with the participants in order to obtain feedback about the activities that they could have implemented in their classrooms. So far the participants have reported that a total of 320 students have received instruction based on the materials from the workshops. Participants are also developing activities and lesson plans (figure 7) in schools where CT activities had not been implemented before. In addition, as reported by the Rio Grande Sun newspaper<sup>28</sup>, an after-class computing club is currently in development in Española, NM with support from a workshop participant. As reported by the newspaper, this is the first ever created K12 club in the city about computer programming.

		<b><i>Española Public School Effective Common Lesson Plan</i></b>		
<b>Teacher:</b> <b>Mr. Jimmy Lara</b>		<b>Grade Level:</b> <b>Sixth Grade</b>	<b>School:</b> <b>JHR Elementary</b>	<b>Content Area:</b>  <i>Technology/ Computer Science</i>
<b>Unit of Study:</b> Computational Thinking		<b>Sessions:</b> See Appendix 1. <i>Planning: Defining computational design processes</i> 2. <i>Reflecting: Design Notebook question</i> 3. <i>Connecting: My favorite song</i>		<b>Time Frame (Period):</b> Fall Semester <b>Date(s) of Instruction:</b> August- September October-November-December
<b>Launch</b>	<p>Substantive/Progressive Learning Tasks (SMART goals – <u>S</u>pecific, <u>M</u>easurable, <u>A</u>ttainable, <u>R</u>ealistic and <u>T</u>imely)</p> <p>In this unit:</p> <p><b>Students will</b> listen, read, and write about how their actions can help others by using grade level standards for writing.</p> <p><b>Students will</b> apply Cornell three column note taking.</p> <p><b>Students will</b> develop an understanding of S.T.E.M. (Science, Technology, Engineering, Mathematics) by using mathematics and technology.</p> <p><b>Students will</b> apply creative computing and Scratch, through sample projects and hands-on experiences</p> <p><b>Students will</b> explore the arts by creating projects that include elements of music, design, drawing, and dance.</p> <p><b>Students will</b> use the computational concepts of sequence and loops, and the computational practices of being iterative and incremental in the development of projects that include characters, scenes and narrative.</p> <p><b>Students will</b> explore games by creating projects that define goals and rules.</p> <p><b>Students will</b> develop independent projects by defining a project to work on, collaborating with others to improve the project, and presenting the project and its development process.</p>			
	<p>Aligned to CCSS/NM Standards</p> <ul style="list-style-type: none"><li>• <i>ISTE NETS Student Standards 2007</i> <a href="http://www.iste.org/standards/nets-for-students/nets-student-standards-2007.aspx">http://www.iste.org/standards/nets-for-students/nets-student-standards-2007.aspx</a><ul style="list-style-type: none"><li>○ Creativity and Innovation – Students demonstrate creative thinking, construct knowledge, and develop innovative products and processes using technology. Students:<ul style="list-style-type: none"><li>▪ apply existing knowledge to generate new ideas, products, or processes.</li><li>▪ create original works as a means of personal or group expression.</li></ul></li><li>○ Communication and Collaboration – Students use digital media and environments to communicate and work collaboratively, including at a distance, to support individual learning and contribute to the learning of others. Students:<ul style="list-style-type: none"><li>▪ interact, collaborate, and publish with peers, experts, or others employing a variety of digital environments and media.</li><li>▪ communicate information and ideas effectively to multiple audiences using a variety of media and formats.</li><li>▪ contribute to project teams to produce original works or solve problems.</li></ul></li><li>○ Research and Information Fluency – Students apply digital tools to gather, evaluate, and use information. Students:<ul style="list-style-type: none"><li>▪ plan strategies to guide inquiry.</li><li>▪ locate, organize, analyze, evaluate, synthesize, and ethically use information from a variety of sources and media.</li><li>▪ evaluate and select information sources and digital tools based on the appropriateness to specific tasks.</li></ul></li><li>○ Critical Thinking, Problem Solving, and Decision Making – Students use critical thinking skills to plan and conduct research, manage projects, solve problems, and make informed decisions using appropriate digital tools and resources. Students:<ul style="list-style-type: none"><li>▪ identify and define authentic problems and significant questions for investigation.</li><li>▪ plan and manage activities to develop a solution or complete a project.</li><li>▪ collect and analyze data to identify solutions and/or make informed decisions.</li><li>▪ use multiple processes and diverse perspectives to explore alternative solutions.</li></ul></li><li>○ Digital Citizenship – Students understand human, cultural, and societal issues related to technology and practice legal and ethical behavior. Students:<ul style="list-style-type: none"><li>▪ advocate and practice safe, legal, and responsible use of information and technology.</li><li>▪ exhibit a positive attitude toward using technology that supports collaboration, learning, and productivity.</li><li>▪ demonstrate personal responsibility for lifelong learning.</li></ul></li><li>○ Technology Operations and Concepts – Students demonstrate a sound understanding of technology concepts, systems, and operations. Students:<ul style="list-style-type: none"><li>▪ understand and use technology systems.</li><li>▪ select and use applications effectively and productively.</li><li>▪ troubleshoot systems and applications.</li></ul></li></ul></li></ul>			
	<p>Building Connections/Background/Relevance</p> <p>Creative computing is about <i>creativity</i>. Computer science and computing-related fields have long been perceived as being disconnected from young people's interests and values. Creative computing supports the development of personal connections to computing, by drawing upon creativity, imagination, and interests.</p> <p><b>Develop an understanding of science and technology in society.</b> Science and technology have advanced through the contributions of many different people, in different cultures, at different times in history. Technology influences society through its products and processes. What are the different ways you interact with computers? How many of those ways involve you <i>creating</i> with computers?</p>			

**Figure 7.** Excerpt from a Lesson Plan created by a participant of the CT workshops

In robotics, participants from the 2012 version that received Lego robots have shared the utilization of the kits to teach concepts in science, physics and technology. Other participants have applied and received grants to create robotic clubs in their schools. In regards to the other activities presented in the workshops, some elementary teachers have also reported the utilization of computer games and kinematic activities to introduce CT in their classrooms, and language teachers are using Scratch and the curriculum for creative computing to combine storytelling and computer programming. Agent-based programming has also been used to teach probability, and organize groups to compete in the NM Supercomputing challenge. However, other teachers have shared that due to school load they have not been able to implement any activity in their classrooms.

## **5. Conclusions**

This paper has presented the development of a curriculum to teach and enhance computational thinking skills for K12 schools through the utilization of different tools including kinematic activities, computer games, programming languages, robotics and human-computer interaction, and agent-based programming. Feedback provided by the participants has shown the success of the workshops as they are bringing CT activities into classrooms where these activities were not implemented before. As a long-term goal the workshops is to increase the recruitment of students into Computer Science/Engineering/CS-related engineering programs, we will continue to monitor further the participants so they can provide feedback of these activities in their classrooms and the perception about their students on choosing a CS-related career for College studies.

## **Acknowledgements**

The authors would like to acknowledge Google Inc. through the Google Computer Science for High School (CS4HS) program for their generous support in the development of the workshops. This material is based upon work supported by the National Science Foundation under Grant No. 1035465. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## **Bibliography**

1. CSTA Voice, Volume 6, Num 5. Available on [http://csta.acm.org/Communications/sub/CSTAVoice\\_Files/csta\\_voice\\_10\\_2010.pdf](http://csta.acm.org/Communications/sub/CSTAVoice_Files/csta_voice_10_2010.pdf)
2. Women, Minorities, and Persons with Disabilities in Science and Engineering: 2013. National Center of Science and Engineering Statistics. National Science Foundation. 2013.
3. Computational thinking. J. Wing. Communications of the ACM No. 49, Vol. 3, 2006
4. K-12 Digital Literacy & Citizenship Curriculum. Commonsensemedia.org. Available online on <http://www.commonsensemedia.org/educators/curriculum>

5. Google Exploring Computational Thinking. <http://www.google.com/edu/computational-thinking>
6. Computational Thinking for Youth in Practice, I. Lee, F. Martin, J. Denner, B. Coulter, W. Allan, J. Erickson, J. Malyn-Smith, L. Werner, ACM Inroads, vol 2. Issue 1, 2011
7. Project GUTS. <http://www.projectguts.org/>
8. Common Core Standards <http://www.corestandards.org/Math/Practice>
9. Rio Arriba Comprehensive Plan. Available under [http://www.rio-arriba.org/pdf/departments\\_and\\_divisions/comprehensive\\_plan.pdf](http://www.rio-arriba.org/pdf/departments_and_divisions/comprehensive_plan.pdf)
10. U.S. Census Bureau. Available under <http://www.census.gov/popest/data/national/totals/2012/index.html> and <http://www.census.gov/population/www/cen2010/cph-t/CPH-T-2.pdf>
11. Super Computing Challenge. <http://www.challenge.nm.org/>
12. CS-Unplugged. [www.csunplugged.org](http://www.csunplugged.org)
13. The Missing Mechanism. Available under [http://www.physicsgames.net/game/Missing\\_Mechanism.html](http://www.physicsgames.net/game/Missing_Mechanism.html) .
14. Amazing Alex. Available under <http://www.amazingalex.com/>
15. Armadillo Run. Available under <http://www.armadillorun.com/>
16. Light-bot. Available under <http://light-bot.com/>
17. SimCity. Available under <http://www.simcity.com/>
18. Scratch. Available under <http://scratch.mit.edu/>
19. Python. Available under <http://www.python.org/>
20. The neglected battle fields of syntax errors. S. K. Kummerfeld and J. Kay. Proceedings of the fifth Australasian conference on Computing Education - Volume 20 (ACE '03). 2003.
21. ScratchEd. Available under <http://scratched.media.mit.edu/>
22. Lego Mindstorms. Available under <http://mindstorms.lego.com/>
23. Makey Makey. Available under <http://www.makeymakey.com/>
24. National Instruments LabView. Available under <http://www.ni.com/labview/>
25. Using Embodiment with LEGO Robotics to Enhance Physics Understanding in Elementary School Students. C. Lu, S. Kang, S.C Huang, J. Black. Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, 2010.
26. Netlogo. Available under <http://ccl.northwestern.edu/netlogo/>
27. Starlogo. Available under <http://education.mit.edu/projects/starlogo-tng>
28. Learning the Code. Published by Rio Grande Sun Newspaper. Published on Dec. 26<sup>th</sup> 2013.