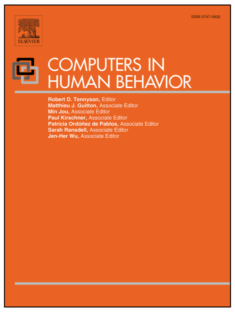


Computational Thinking in pre-university Blended Learning classrooms

Xabier Basogain, Miguel Ángel Olabe, Juan Carlos Olabe, Mauricio Javier Rico Lugo

PII: S0747-5632(17)30301-1

DOI: 10.1016/j.chb.2017.04.058

Reference: CHB 4957

To appear in: *Computers in Human Behavior*

Received Date: 24 February 2017

Revised Date: 18 April 2017

Accepted Date: 30 April 2017

Title: **Computational Thinking in Pre-University Blended Learning Classrooms**

Authors: **Xabier Basogain**
Escuela Superior de Ingeniería
The Basque Country University (EHU)
Bilbao, Spain
xabier.basogain@ehu.eus

Miguel Ángel. Olabe
Escuela Superior de Ingeniería
The Basque Country University (EHU)
Bilbao, Spain
miguelangel.olabe@ehu.eus

Juan Carlos Olabe
Electrical & Computer Engineering
Christian Brothers University, (CBU)
Memphis, TN, USA
jolabe@cbu.edu

Mauricio Javier Rico Lugo
Instituto Colombiano de Aprendizaje (INCAP)
Bogotá, Colombia
mauricio.rico@incap.edu.co

Computational Thinking in Pre-University Blended Learning Classrooms

Abstract—This article describes the implementation of various core elements of Computational Thinking (CT) in the classrooms of schools of Latin America and USA in two specific courses: PC-01 and ECE130. These courses were designed for students of primary and secondary education, as well as for students of high school as part of a dual enrollment program with a local university. Both courses introduce the core concepts and processes of CT aided by the visual programming environments Scratch and Alice. The courses are facilitated by the classroom teacher with the support of a learning platform. This platform is configured to provide innovative pedagogical strategies based on emerging educational technologies. This article describes the concepts integrated under the term CT, and discusses the benefits of learning environments used to incorporate CT in the classroom. It describes as well the syllabi and assessments of both courses, and analyzes their impact of these courses on the educational institutions, the teachers and the students.

Keywords— Computational Thinking; Cognitive Science; Learning Technologies; Scratch; Alice; Educational Technology.

1. Introduction

After a period of transition in which the fundamental ideas promoted by CT were studied and analyzed by the constituencies of public and private education, it appears that a consensus is being reached which can shape future decisions and policies in K-12 education (Hubwieser, Armoni, Giannakos and Mittermeir, 2014). One important consequence of this confluence of ideas is the definition of K-12 curricula with the integration of the core ideas of CT, out of which the particular classroom paradigms will be developed. In addition, it is universally agreed that programming, in its various forms, is a necessary mechanism for the implementation of CT core concepts and best practices. Programming provides the three mechanisms required by a language in the creation of complex systems: a set of primitives; some means of combination; and abstraction. Unlike natural languages, which by design are semi-structured and provide limited ability of multilevel abstraction, programming languages have been designed to realize the core ideas of CT. A mind equipped with the mechanisms of object oriented programming languages is prepared to house the principles of CT and is prepared to develop CT ideas.

The need for the formal study of computational concepts in primary and secondary schools has been recognized by many institutions and administrations. For example, England formally incorporated in 2014 the study of CT and computer programming in the curriculum of primary and secondary education (Department for Education England, 2013).

The organization Code.org (Code.org, 2012; Kalelioğlu, 2014) promotes the idea that all students should have the opportunity to learn programming. This initiative has the support of public figures from Microsoft, Facebook and the world of technology in general. Other initiatives, such as the European Project TACCLE 3 – Coding, provide resources to students and teachers in primary and secondary education in the area of computing (García-Peñalvo, 2016; García-Peñalvo, Reimann, Tuul, Rees, & Jormanainen, 2016).

Programming environments such as Scratch, ScratchJr, and Alice play an essential role in this process. These and other graphic programming environments were designed to address specifically the developmental and learning needs of children (ages 5-7) and youth adults (ages 8-15) (Flannery, Silverman, Kazakoff, Bers, Bontá & Resnick, 2013; Resnick, Maloney, Monroy-Hernández, Rusk, Eastmond, Brennan & Kafai, 2009; Sáez López, González & Cano, 2016).

Alice (Alice, 2017) is a graphical programming environment integrated within a three dimensional world. It is an object oriented programming environment where, in addition to programming with graphic interlocking blocks, the programmer has immediate access to the Java code that has been created behind the scenes. Because Alice hides from the user the syntactical complexity of Java, and because it is integrated within a 3D environment, it provides young students with an ideal environment to learn the fundamental programming concepts in the motivating context of creating movies and video games (Zhang, Liu, Ordónez de Pablos & She, 2014). These movies and games create a perfect platform where students express their interest and creative ideas (Denner, Werner & Ortiz, 2012; Zhang, Ordónez de Pablos & Zhu, 2012). They also offer a rich object oriented world where human cognitive primitives (objects with properties, behaviors, and interfaces with other objects) can express core concepts of CT and object oriented programming.

1.1. Computational Thinking

The history of CT, as it is the case in many developments in Science, reflects the convergence of multiple ideas, often from different areas of study (Cognitive Sciences, Linguistics, Psychology, and Computer Science). After being developed in isolation, these ideas found a synergetic effect when applied to the area of education, and in particular the processes involving generative languages for the creation of novel methods and complex systems.

Some of the pioneer thinkers in this field include Papert, Wing and Wolfram. One of the earliest references to CT is contained in (Papert, 1996), where Papert describes the value of applying human cognitive primitives to object oriented problems by noticing the relationships between the components of a complex system. Other similar references can be found in (Vee, 2013; Wolfram, 2016), in which there are direct references to the fundamental ideas of dividing a complex task into a set of simpler tasks.

We can find some specific descriptions regarding the core elements of CT in the work that the International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA). These institutions collaborated with industry and K-12 education, and developed tools to address the needs of the educators that will eventually integrate this discipline in the classrooms (Mannila, Dagiene, Demo, Grgurina, Mirolo, Rolandsson & Settle, 2014; Sykora, 2014).

The field of CT is still in its infancy; even if the overall goals and core principles are defined, two important tasks should be the object of research in the near future. The first is to discover the uncharted world of cognitive processes that the human mind can sustain, provided the required tools. Cognitive science has dramatically transformed the model for which the educational system of the past was designed. We know now, better than before, what the limits of the mind are (Pinker, 1995; Pinker, 1999; Kahneman, 2003; Kahneman, 2011). The task ahead is to discover the human computational resources, the cognitive primitives of the mind, in order to mine their power. The second task is to develop a set of experiences that will allow students to develop these primitives, engage in these cognitive processes, and produce rich and innovative portfolios. The educational system, K-12 and beyond, is based on a curriculum characterized by the use of descriptive languages and the study of Type-A problems (Olabe, Basogain, Olabe, Maíz & Castaño, 2014). The curriculum of the future will likely be designed for the use of generative languages (object-oriented languages) and the study of Type-B problems.

In England, there are some initial steps for the integration of CT in the classroom (Department for Education England, 2013). In the United States, the main path for pre-university students to participate in CT activities and curriculum is the Advanced Placement Computer Science courses (Berkeley, 2017; Harvard, 2017).

1.2. Blended Learning

Virtual Learning Environments (VLEs) offer benefits to teachers and students. Teachers encourage collaboration and communication among students, and customize and differentiate progress in the classroom. Students learn to work with their peers on projects, and develop collaborative skills and problem solving strategies.

However, VLEs also require economic investments, technical support, and additional training for teachers (Paddick, 2014). These requirements have created an uneven growth of VLEs in educational systems (Johannesen, 2013).

The educational environment, however, is beginning to change because of several developments:

- public and private schools are recognizing the educational value of this new type of learning;
- there is an evolution towards VLEs and e-learning (Gros and García-Peñalvo, 2016) that are easier to use and are better adapted to the needs of students;
- the MOOC (Massive Open Online Courses) phenomenon (Breslow, Pritchard, DeBoer, Stump, Ho & Seaton, 2013; Vila, Andrés & Guerrero, 2014) and its variations (COOC, NOOC, SPOC) have changed the way we use technology in distance and blended education (Benfield, Roberts & Francis 2006; Bruff, Fisher, McEwen & Smith, 2013).

The two courses described in this paper are designed around Moodle (https://moodle.org/) and the features that it provides as a VLE platform designed to support innovative pedagogical strategies.

The methodology of the course is based on a learner-centered paradigm. This implies that the tasks of the student include: a) studying a series of short video lessons, b) taking interactive quizzes, c) assessing and being assessed through testing and Peer-to-Peer projects (P2P) and d) participating in online forums with classmates and teachers (Glance, Forsey & Riley, 2013).

These courses emphasize the experience of student learning. The core concepts and processes of CT (College Board, 2016) that have been developed in the two courses are evaluated through Test and P2P activities.

The design of the courses presented in this paper is based on a blended structure. One objective is to benefit from the qualities of the local teacher in the classroom (deep knowledge of the students, their learning styles and preferences, the ideal class dynamic, etc.). A second objective is to benefit from the services provided by the VLE (design and delivery of curriculum with multimedia, individual monitoring of the progress of each student, assessment and P2P evaluations, portfolio management, etc.) (Barbour, Brown, Waters, Hoey, Hunt, Kennedy & Trimm, 2011; Grover, Pea & Cooper, 2015).

2. PC-01 Course

2.1. Content

Our research team has focused its teaching and research activities of the last decade in the area of technology and e-learning education. The team was able to analyze, experiment with, and implement systematic studies of courses offered in these platforms (Olabe, Basogain & Olabe, 2016).

One of these courses is 'PC-01: Introduction to Computational Thinking' and it was designed to allow immediate implementation in the school and simple access to contents and tools. Its objective was the introduction to concepts and processes in CT.

The course is organized into 10 sessions, each lasting two hours. The sessions were named according to the families of Scratch blocks being studied: Movement, Event, Control, etc. The VLE selected was Moodle, and the programming environment Scratch.

2.2. Assessments

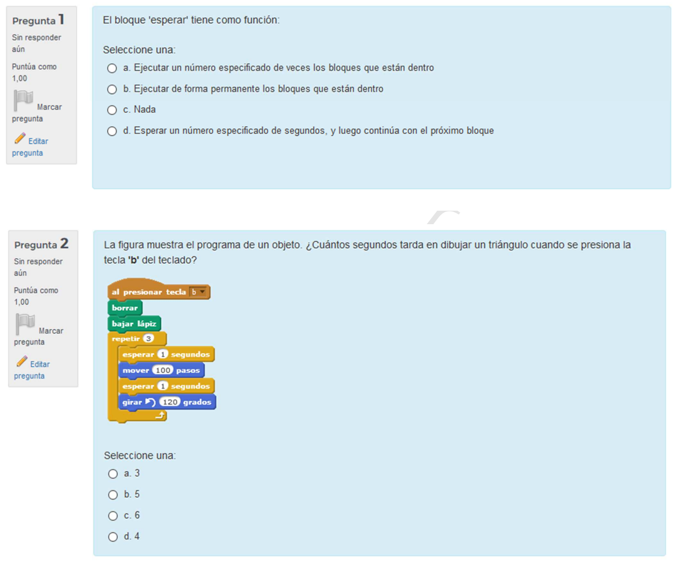
The course PC-01 is based on a continuous student assessment with feedback, the creation of a student portfolio of projects, and the mastery core concepts. It includes ten tests, based on multiple-choice questions of concepts. The students are also required to analyze programming scripts and programming structures.

The construction of the student's portfolio is implemented with the cumulative collection of projects created as part of the P2P assignments. These projects document the abilities acquired by the student, including programming paradigms and the corresponding core ideas of CT. Table 1 lists the collection of ten P2P projects implemented in the course PC-01.

Table 1. PC-01: List of P2P Projects

Project	CT Core Ideas
P2P 1 Design of a project to draw three triangles on the stage.	Sequence of instructions and repetition control.
P2P 2 Design of a project where the protagonist narrates a short personal story.	Use of costumes and text messages.
P2P 3 Design of a project with 4 protagonists playing different sounds.	Use of multiple sprites, play sound procedures.
P2P 4 Design of a project where the protagonist draws on the stage images of creative art with lines and curves of multiple colors.	Controlling the pen resource of the sprite including the color and position.
P2P 5 Design of a project to allow the user to draw on the stage with a pen of multiple colors.	Use of Events and Event handlers with simple control loops.
P2P 6 Design of a project to implement a clock with a moving hand associated with multiple sound and graphic effects.	Control structures and introduction to multi-script programming.
P2P 7 Design of a project with sprites ball, beetle, crab and beach ball interacting with each other using sensors of the environment.	Sensors that report information on the environment.
P2P 8 Design of a project where a penguin exhibits mathematical talents.	Use of numerical and text operators, concatenation.
P2P 9 Design of a project to control the speed and color of a spacecraft by using variables.	Simple data manipulation (setting and changing variable content).
P2P 10 Design of a project for the creation of new, user-defined programming blocks. The new blocks will allow the complex drawing of geometric patterns with variable parameters.	Custom blocks to abstract the functionality of scripts and make programming a modular task.

Figure 1 includes two examples of multiple choice questions presented to students during self-assessment and test sessions. They include a text-based conceptual question and a graphical script-programming question.


Pregunta 1
Sin responder aún
Puntúa como 1,00
Marcar pregunta
Editar pregunta

El bloque 'esperar' tiene como función:

Seleccione una:
- a. Ejecutar un número especificado de veces los bloques que están dentro
- b. Ejecutar de forma permanente los bloques que están dentro
- c. Nada
- d. Esperar un número especificado de segundos, y luego continúa con el próximo bloque

Pregunta 2
Sin responder aún
Puntúa como 1,00
Marcar pregunta
Editar pregunta

La figura muestra el programa de un objeto. ¿Cuántos segundos tarda en dibujar un triángulo cuando se presiona la tecla 'b' del teclado?

```
al presionar tecla b
borrar
bajar lápiz
repetir 3
    esperar 1 segundos
    mover 100 pasos
    esperar 1 segundos
    girar 120 grados
```

Seleccione una:
- a. 3
- b. 5
- c. 6
- d. 4

Fig. 1. Text-based and graphical Scratch script programming questions.

Figure 2 illustrates a solution implemented by students in a P2P project. The figure shows the use of multi-script programming to allow the user to control the velocity and appearance of a spacecraft by using variables. The project studies the powerful paradigm of decomposition of a complex task into simpler goals, each implemented with a custom script.

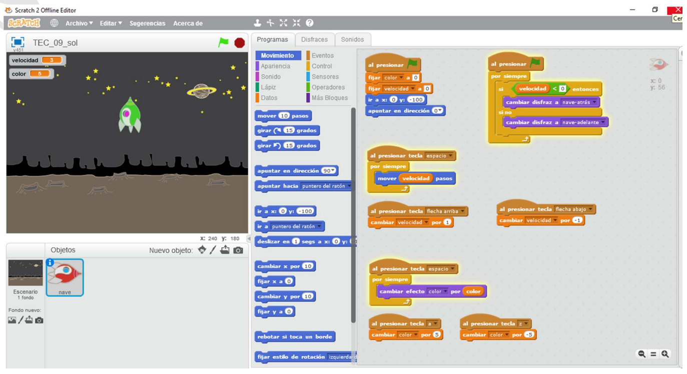


Fig. 2. Solution of P2P 9, multi-script and decomposition.

Students participating in the course PC-01 successfully achieved the academic goals set in the project. In addition, students expressed a sentiment of satisfaction by their participation in activities they considered motivating and where they could express their different creative interests (Basogain, Olabe, Olabe, Ramírez, Del Rosario & Garcia, 2016).

3. Course ECE130

3.1. Content

In addition to the PC-01 course (and other related courses, which are designed for students 10-15 years old) our team also focused in the needs and challenges of the introduction of CT in the classroom of pre-university age students.

For this project, we benefited from two especial characteristics that are preeminently present in the educational system of the United States. One is the ability of students in high school to earn college credit while in high school by taking Advance Placement courses (College Board, 2017) or by enrolling in dual-enrollment courses that offer credit both in the high school at the University-level. The second is the increasing demand for computer literacy courses required in universities as part of the GER's (General Education Requirements) of all students.

Christian Brothers University (CBU, 2017) has participated for many years with several high schools in dual-enrollment programs. In 2009 the course ECE130 (Introduction to Computational Thinking and Programming) was designed to introduce junior and senior students in high school to the fundamental ideas of CT while studying the core foundations of programming.

The course was designed as an introduction to object oriented programming using graphics and the creation of 3D movies, games and interactive applications. Alice was selected as the programming environment given the special features that it offered to the novice programmer. Alice eliminates all syntactical errors by using a menu driven interface and it is integrated into a 3D world that allows students to assimilate object-oriented principles by association with state values, procedures, and functions of real life objects.

The course is structured into twelve units. Each unit includes a set of core ideas and principles that are illustrated via simple 3D projects that increase in complexity.

3.2. Assessments

After an initial mastery phase, the student needs to complete open-ended projects in which the new ideas are integrated. These projects always take the form of a movie, a game or an interactive application. A comprehensive rubric is provided as a guide for the student. This rubric will then be used in the peer assessment of other students' work.

The weekly assessment is implemented with three separate tools: self-assessment, test, and P2P assessment. The self-assessment is a tool that allows students to review a comprehensive checklist of core ideas, new vocabulary, and formal programming rules that need to be mastered before proceeding further. The tests are independent assessments that document the progress of students and certify the achievement required for a successful completion of the course.

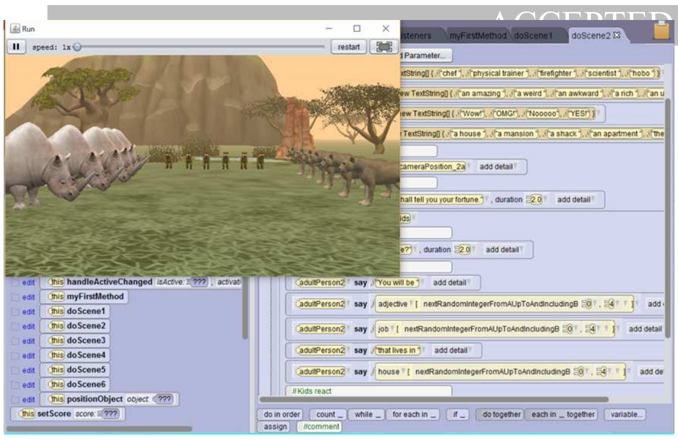
The P2P assessments address several student skills in the process of acquiring the status of a Computational Thinker. One developed quality is the ability to read code, to interpret knowledge represented in the form of programming code, and to assess its complexity and quality. By assessing the work of others, and comparing the assessments with the rubric assessments, the student develops expertise in the skill of reading, interpreting, and evaluating the quality of written code.

A second skill developed by the P2P assessments transforms the traditional passive role of the student, in which his or her work is evaluated by the teacher, into an active role in which the student is given the privilege and the responsibility of evaluating the work of others. This important responsibility conveys the implicit message that educating the student means making the student an expert: an expert that can create quality work, and an expert that can appreciate and evaluate the work of others. Table 2 includes a list of the ten P2P projects of the course ECE130.

Table 2. ECE130: List of P2P Projects.

Project	CT Core Ideas
P2P 1 Design of a project with 4 Scenes and 2 Sub-scenes. Leader object defines action to be implemented.	Top-down design and bottom up implementation. Decomposition.
P2P 2 Design of a project using Class level Methods. Use of Markers to identify physical locations.	Objects contain behaviors that can be expanded with new Class Methods.
P2P 3 Design of a project using user-defined functions. Functions will obtain information directly form the environment and pass it to the main program.	Objects can determine useful information from the environment through the use of user-defined functions.
P2P 4 Design of a project with Class Methods and Functions that use Parameters.	Class methods and functions provide adaptive behavior through the use of parameters.
P2P 5 Design of a project with the use of control structures for individual and collective segments of code.	Control structures add versatility to the code of individual objects and collections of objects.
P2P 6 Design of a project with nested groups of control structures for individual objects and collections of objects.	Complex topologies of nested control structures add functionalities to languages.
P2P 7 Design of a project with data structures including one and two dimensional arrays of objects.	Data Structures allow the manipulation of families of objects efficiently and dynamically.
P2P 8 Design of project using Random strategies for the control of arrays.	The use of Random functions allows the resolution of problems through trial and error strategies.
P2P 9 Design of an interactive project by interfacing with objects through events and event handlers.	Events and event handlers allow the implementation of dynamic systems.
P2P 10 Design of a comprehensive project including all the core elements of the course.	Complex dynamic systems, using arrays of objects and interfacing with the external world can be implemented in modular form.

Some of the features of the Alice programming environment are illustrated in Figure 3. This environment includes separate areas to edit the scene and the program. It also includes separate areas for the methods, functions, and states of each object.



```
d Parameter...
xtString[] { "chef" , "physical trainer" , "firefighter" , "scientist" , "hobo" }
ew TextString[] { "an amazing" , "a weird" , "an awkward" , "a rich" , "an u
ew TextString[] { "Wow!" , "OMG!" , "Nooooo" , "YES!" }
TextString[] { "a house" , "a mansion" , "a shack" , "an apartment" , "the
cameraPosition_2a   add detail
hall tell you your fortune." , duration 2.0   add detail
ids
e?" , duration 2.0   add detail
adultPerson2 say "You will be"   add detail
adultPerson2 say adjective [ nextRandomIntegerFromAUpToAndIncludingB 0 , 4 ]   add
adultPerson2 say job [ nextRandomIntegerFromAUpToAndIncludingB 0 , 4 ]   add detail
adultPerson2 say "that lives in"   add detail
adultPerson2 say house [ nextRandomIntegerFromAUpToAndIncludingB 0 , 4 ]   add de
//Kids react

edit  this handleActiveChanged isActive: ??? , activati
edit  this myFirstMethod
edit  this doScene1
edit  this doScene2
edit  this doScene3
edit  this doScene4
edit  this doScene5
edit  this doScene6
edit  this positionObject object: ???
this setScore score: ???

do in order | count _ | while _ | for each in _ | if _ | do together | each in _ together | variable... | assign | //comment
```

Fig. 3. Graphical Alice script programming of object arrays.

The programming environment of Alice provides students recently introduced to the world of programming a selected set of tools that, given their simplicity, are easy to learn and manipulate, and at the same time allow the rapid prototyping of complex programs with a multiplicity of interacting objects.

4. Discussion

The integration of CT curriculum and activities in the educational system will require the transformation in some fundamental ways of the relationship between schools and society, the role of the teacher in the classroom, and the students' experience during their period of education.

4.1. Impact on Educational Institutions

The final structure of the courses described includes a set of software resources that are easy to install in a VLE and that offer a comprehensive set of educational services in the area of CT for primary and secondary students. The courses are taught in a blended format in the computer laboratory and with the support of the school VLE platform.

The course PC-01 is hosted in a virtual classroom EDUCANDO online (http://aula.educando.edu.do), which is the portal under the supervision of the Department of Computer Education of the Ministry of Education of the Dominican Republic (MINERD).

The VLE platform serves as the central academic repository of all students' resources, activities, and projects allowing the administrators to evaluate in a comprehensive manner all schools associated with the project (Sancho Gil & Padilla Petry, 2016).

At the same time, the VLE environment provides means for sharing resources among the schools, backup installations, deployment of institutional badges, and other centralized services.

These services are made possible by features present in Moodle, and other environments, that automatically generate reports on students' activity, participation, and learning goals achievements in a wide set of learning analytics. Some of these features are part of the standard distribution of Moodle (Moodleplugins, 2017).

The availability of Learning Analytics is fundamental in the process of improving the learning outcomes of an educational institution. These analytics provide direct feedback and information not only to students and teachers, but to administrators and decision makers as well.

An example of higher level learning analytics is the course overview. This tool allows ranking multiple courses by activity, participation, and other criteria. This type of comparative analysis is important for medium and large size institutions in order to determine the efficiency of different policies and strategies, and it would be almost impossible to implement with traditional mechanisms.

4.2. Impact on Teachers and Teaching Loads

One of the fundamental changes in education with the assistance of VLE environments is the creation of a collaborative mode of education where students receive their knowledge from both the local teacher, present in the classroom, and the multimedia-based knowledge embedded in the VLE. For example, to offer CT courses in an institution, it is not necessary for the local classroom teacher to be an expert in the field. What is important is that the teacher is knowledgeable in the use of the VLE. The content of the course, assessments, projects, and other academic tasks are provided by what we could call the "guest teacher" through the VLE environment. The local teacher is the classroom leader, motivator, and director of activities.

The local teachers in these courses have a set of resources and services that facilitate the delivery and monitoring of an introductory course in CT. These teachers have access to collections of video tutorials, self-tests, tests, and P2P projects covering the scope of the course curriculum. Before starting the course, the teachers receive training on methodology and course content.

To assist fully the local teacher, a comprehensive guide has been created describing the structure and operation of the main parts of the course (http://www.ehu.eus/es/web/gmm/minerd-pc01#2). This documentation includes a description of the topics studied in the course, the pedagogical resources available, and the organization of the course into modular sessions. In the area of assessment, including the individual assessment and the P2P evaluation, the documentation provides guidelines for teachers and students on the best use of these resources. Finally, a comprehensive index includes the available resources (video tutorials, practice projects, self-test, tests, and the solutions to the P2P projects).

Teachers also find a number of integrated analytical tools in the VLE that allow them to implement collective and individual student progress assessments of concepts and process in CT (Singh, 2015).

4.3. Impact on Students

The students construct their own knowledge in a process described by the pedagogical theory of constructionism. This theory, proposed by Seymour Papert (Papert, 1991), proposes that students build their knowledge through the construction of an 'artifact' that motivates them. For the construction of an artifact, these courses use the programming environments Scratch or Alice. Each course unit is contains two constructionist sections: Practice and P2P.

In the Practice section students experiment in their programming environment with the conceptual contents developed by the teacher in the video tutorials; in the P2P section, the students must design and build a project to solve a situation or a proposed problem.

One goal of these courses is to teach students the principles of a computer programming language. Students use Scratch or Alice as tools that allow them to communicate and create ideas to solve problems (Disessa, 2000).

The implementation of a Scratch or Alice project that solves a problem requires the development of higher cognitive functions of the mind. These functions include categorization, decision making, abstraction, insight, problem solving, planning, and execution.

The second part of the P2P tasks, the assessment of programming projects of 3-5 fellow students, according to a specific rubric, is implemented anonymously, and it promotes the development of higher cognitive functions. The students are required to play the role of evaluators, and in turn deepen their knowledge in order to be able to evaluate Scratch or Alice projects created by other students.

In addition to being exposed to other Scratch or Alice projects, these activities open for the student the opportunity to learn and be inspired by new solutions for future problems (Lu & Law, 2012).

The experience of the students in these hybrid courses also allows them to become familiar with the tools and methodologies of the VLE environment. As students of a VLE system, they have experienced different aspects of this form of learning: multimedia format content, collaboration mechanisms, self-assessment and evaluation mechanisms, progress in their knowledge and grades. The courses offer a badge, a prize or medal to recognize accomplishments and provide students with a reward that acknowledges their achievement in CT (Seliskar, 2014).

In the immediate future, these students will see their own knowledge progress with tools and services of educational online platforms offering massive MOOC courses. Lifelong learning will require students the use of VLE environments similar to the environment they experienced in these courses (Attwell & Hughes, 2010).

4.4. Participants & Results

During the last several years, these courses have been implemented in classrooms of high schools in the United States (with continuous evaluation during the academic years, from 2009 until 2016) and in secondary schools in the Dominican Republic (with evaluation in: Study-1, April-June 2016; and Study-2, December-2016/March-2017).

Study-1: Dominican Republic: The PC-01 project was implemented during a six-month period. The first part included the design, implementation, and fine-tuning of the VLE. The course was later replicated on the platform EDUCANDO online of the Ministry of Education (MINEDU). In this platform, virtual classrooms for ten schools in the area of Santo Domingo were created.

The second part of the project involved the delivery of the course in four schools. Previously, the teachers of the schools were trained to facilitate the course. The course was taught by the teacher to his/her students in person in the classroom with the support of the learning platform. The duration of the course varied, and lasted between 6 and 10 weeks. Each classroom included 10 students, with an average of eight active students.

The outcomes of the course were assessed using the portfolio of each student and the number of successful tests. These parameters indicate a high degree of success. Other course results were obtained from personal interviews with students and teachers, and they indicate a high degree of satisfaction on the part of the course participants (some testimonials: 'easy', 'funny', 'entertaining', 'learn more', 'create') (Basogain, Olabe, Olabe, Ramírez, Del Rosario & Garcia, 2016).

The second study in the Dominican Republic included 21 schools distributed across the country in the areas of Santo Domingo, Pedro de Macorís, San Juan de Maguan, San Francisco de Macorís, and Puerto Plata. Each participating school included one or two groups of students, with students of primary and secondary education (10-15 years old).

A critical class management tool offered by the VLE is the ability to process and analyze the numerical data of the students' level of learning and achievement. In particular, in the course PC-01 we analyzed: 1) the number of self-assessment and test attempts by the students; and 2) the corresponding grades obtained in those attempts. It is relevant to note that students could attempt the self-assessment as many times as they wished without direct impact in their grades. The tests, however, could only be attempted twice, with the best grade being recorded.

To illustrate the results obtained in this study, we include Figure 4, which shows the results of two particular Moodle classrooms located in the cities of Las Matas de Farfán and San Pedro Macorís. This figure shows the total number of attempts and the grades obtained in self-assessments and tests in the classrooms of these two schools.

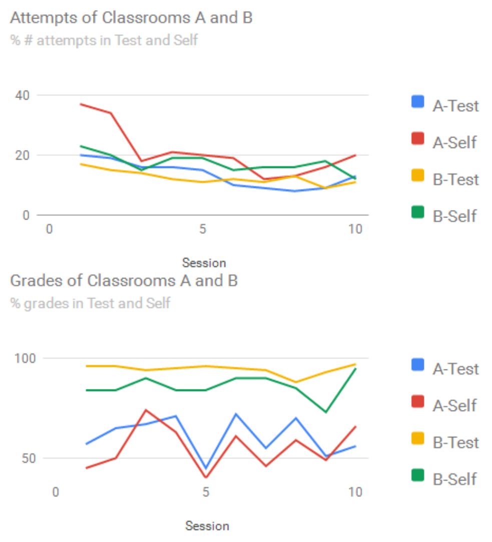


Fig. 4. Attemps and Grades in assessments and tests in the course PC-01.

Analyzing the graph of attempts, one can observe the general descending tendency throughout the ten weekly sessions of the course. This seems to indicate that the students initially chose to practice for longer periods of time, when the subject was still new and unknown. In addition, as the course advanced, the students experienced an increased confidence on their skills and knowledge, and therefore felt less need to complete additional attempts.

Analyzing the graph of the grades one can observe a general pattern of better grades on the tests than on the self-assessments throughout the course. This seems to indicate that the students gradually reinforced their knowledge on core fundamentals as they accumulated experience with time.

The course ECE130 is traditionally offered to several high schools in the Memphis area. Each high school participates with one or two sections of students in their Junior or Senior year, with ages 16 to 18.

Given the open environment in which these courses are offered, special effort is dedicated to guarantee that a large set of randomized tests and assessments are available to prevent the possible data sharing among students.

Throughout the years, the course ECE130 has experienced an increase in the complexity of the P2P projects as students adapt ever more rapidly to new technologies and evaluation methods using computers.

4.5. Future directions

PC-01 in Colombia: A new project of the course PC-01 has been created between the University of the Basque Country and the Corporation for the National Academic Network for Advance Technology (RENATA) in Colombia via an agreement of collaboration with the title "Introduction of Computational Thinking in the Schools of Bogota and Colombia". In addition, the Ministry of Information Technologies and Communications (MINTIC) of Colombia has joined the collaboration agreement, providing resources for the implementation of the project in 10 schools distributed across the country. The project is being developed during the second semester of the academic year 2016-17, and it is supported by a Moodle-based platform (Renata, 2017).

5. Conclusion

In this paper, we present two courses on CT for primary, secondary education, and high school. These courses introduce the concepts and processes of CT in a hybrid classroom format and online education.

These courses represent real examples of cooperation between computer science and educational technology. The courses use the educational services of VLEs systems to train students in the new curriculum subject of CT. The courses use analytical learning tools and help teachers perform successfully the tasks of teaching, monitoring, and grading the students. On their part, the students have a creative learning experience in an environment of collaboration, and are able to monitor and track their own progress.

References

Alice, (2017). An Educational Software that teaches students computer programming in a 3D environment. Retrieved from: http://www.alice.org

Attwell, G., & Hughes, J. (2010). Pedagogic approaches to using technology for learning: *Literature review.*

Barbour, M., Brown, R., Waters, L. H., Hoey, R., Hunt, J. L., Kennedy, K., & Trimm, T. (2011). Online and Blended Learning: A Survey of Policy and Practice from K-12 Schools around the World. *International Association for K-12 Online Learning*.

Basogain, X., Olabe, M. A., Olabe, J. C., Ramírez, R., Del Rosario, M. and Garcia, J. (2016). PC-01: Introduction to Computational Thinking. Educational Technology in Primary and Secondary Education. 2016 *International Symposium on Computers in Education (SIIE).* pp. 1-5. doi:10.1109/SIIE.2016.7751816

Benfield, G., Roberts, G., & Francis, R. (2006). The undergraduate experience of blended e-learning: a review of UK literature and practice. *London: Higher Education Academy.*

Berkeley, (2017). The Beauty and Joy of Computing. Retrieved from: http://bjc.berkeley.edu/

Breslow, L., Pritchard, D. E., DeBoer, J., Stump, G. S., Ho, A. D., & Seaton, D. T. (2013). Studying learning in the worldwide classroom: Research into edX's first MOOC. *Research & Practice in Assessment*, 8.

Bruff, D. O., Fisher, D. H., McEwen, K. E., & Smith, B. E. (2013). Wrapping a MOOC: Student perceptions of an experiment in blended learning. *Journal of Online Learning and Teaching*, 9(2), 187.

CBU, (2017). Christian Brothers University. (2017). Retrieved from: https://www.cbu.edu/

Code.org, (2012). Anybody can learn. Retrieved from: http://code.org

College Board, (2016). AP Computer Science Principles. Course and Exam Description. Retrieved from: https://secure-media.collegeboard.org/digitalServices/pdf/ap/ap-computer-science-principles-course-and-exam-description.pdf

College Board, (2017). AP Courses. Retrieved from: https://apstudent.collegeboard.org/apcourse

Denner, J., Werner, L., & Ortiz, E. (2012). Computer games created by middle school girls: Can they be used to measure understanding of computer science concepts? *Computers & Education*, 58, 240–249.

Department for Education England, (2013). "National curriculum in England: computing programmes of study - key stages 1 and 2". Ref: DFE-00171-2013. Retrieved from: https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study

Disessa, A. (2000). Changing minds: Computers, learning, and literacy. *Cambridge: MIT Press*.

Flannery, L. P., Silverman, B., Kazakoff, E. R., Bers, M.U., Bontá, P., & Resnick, M.(2013). Designing scratchjr: Support for early childhood learning through computer programming. *In Proceedings of the 12th International Conference on Interaction Design and Children ACM*. pp. 1-10.

García-Peñalvo, F. J. (2016). A brief introduction to TACCLE 3 - Coding European Project. In F. J. García-Peñalvo & J. A. Mendes (Eds.), 2016 International Symposium on Computers in Education (SIIE 16). USA: IEEE.

García-Peñalvo, F. J., Reimann, D., Tuul, M., Rees, A., & Jormanainen, I. (2016). An overview of the most relevant literature on coding and computational thinking with emphasis on the relevant issues for teachers. Belgium. doi:10.5281/zenodo.165123

Glance, D. G., Forsey, M. & Riley, M. (2013). The pedagogical foundations of massive open online courses. *First Monday*, 18 (5). Retrieved from: http://firstmonday.org/ojs/index.php/fm/article/view/4350/3673. doi: 10.5210/fm.v18i5.4350

Gros, B. and García-Peñalvo, F.J. (2016): Future trends in the design strategies and technological affordances of e-learning. In book: Learning, Design, and Technology, pp.1-23. doi:10.1007/978-3-319-17727-4_67-1

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, 25(2), 199-237. doi: 10.1080/08993408.2015.1033142

Harvard, (2017). CS50: Introduction to Computer Science. Retrieved from: https://cs50.harvard.edu/

Hubwieser. P., Armoni, M., Giannakos, M. N., & Mittermeir,R. T. (2014). Perspectives and visions of computer science education in primary and secondary (k-12) schools. *Transactions on Computing Education*, 14(2):7:1{7:9

Johannesen, M. (2013). The role of virtual learning environments in a primary school context: An analysis of inscription of assessment practices. *British Journal of Educational Technology*, 44: 302–313. doi: 10.1111/j.1467-8535.2012.01296.x

Kahneman, D. (2003). Maps of Bounded Rationality: Psychology for Behavioral Economics. *The American Economic Review*, vol. 93 no. 5, pp. 1449-1475. doi: 10.1257/000282803322655392

Kahneman, D. (2011). Thinking, fast and slow. *Macmillan*.

Kalelioğlu, F. (2014). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior*, Volume 52,Pages 200-210, ISSN 0747-5632, http://dx.doi.org/10.1016/j.chb.2015.05.047.

Lu, J., & Law, N. W. Y. (2012). Understanding collaborative learning behavior from Moodle log data. *Interactive Learning Environments*, 20(5), 451-466.

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L. & Settle, A. (2014). Computational Thinking in K-9 Education. In ITiCSE '14 *Proceedings of the 2014 conference on Innovation & technology in computer science education*. (pp. 1-29). doi: 10.1145/2713609.2713610

Moodleplugins, (2017). Moodle plugins. Retrieved from: https://moodle.org/plugins/

Olabe, J.C., Basogain, X., Olabe, M.A, Maíz, I. & Castaño, C. (2014). Solving Math and Science Problems in the Real World with a Computational Mind. *Journal of New Approaches in Educational Research*, vol.3 no. 2, pp. 75-82. doi: 10.7821/naer.3.2.75-82

Olabe, J. C., Basogain, X. & Olabe, M. A. (2016). Developing New Educational Frontiers through Breakthroughs in Cognitive Computation and New Dimensions in Pedagogical Technology. *International Journal of Social Science and Humanity*, vol. 6, no. 11, pp. 813-820. doi:10.18178/ijssh.2016.V6.755

Paddick, R. (2014). As easy as VLE. Education Technology. Retrieved from: http://edtechnology.co.uk/Article/as_easy_as_vle

Papert, S. (1991). Situating constructionism. In I. Harel & S. Papert (Eds.), Constructionism. 1-11. *Norwood, NJ: Ablex*

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1):95{123,1996.

Pinker, S. (1995). The language instinct: The new science of language and mind. *Penguin UK*, vol. 7529.

Pinker, S. (1999). How the mind works. *Annals of the New York Academy of Sciences*, 882.1, pp.119-127.

Renata, (2017). Red Nacional Académica de Tecnología Avanzada, Aula Virtual - Curso PC-01. Retrieved from: http://168.227.244.48/moodle/

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11), 60-67.

Sáez López, J.M., González, M.R. & Cano, E.V. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using "scratch" in five schools. *Computers & Education*, doi: 10.1016/j.compedu.2016.03.003.

Sancho Gil, J., & Padilla Petry, P. (2016). Promoting digital competence in secondary education: are schools there? Insights from a case study. *Journal Of New Approaches In Educational Research*, 5(1), 57-63. doi:10.7821/naer.2016.1.157

Seliskar, H. V. (2014). Using Badges in the Classroom to Motivate Learning. Faculty Focus. *Magna Publications*. Retrieved from: http://www.facultyfocus.com/articles/teaching-with-technology-articles/using-badges-classroom-motivate-learning/

Singh, J. (2015). Learning Analytics tools available in Moodle. Retrieved from: http://www.moodleworld.com/learning-analytics-tools-available-in-moodle-moodleresearch-moodleworld/

Sykora, C. (2014). Computational thinking. Operational Definition of Computational Thinking for K–12 Education. *ISTE website*. https://www.iste.org/explore/articleDetail?articleid=152

Vee, A. (2013). Understanding computer programming as a literacy. *Literacy in Composition Studies*, 1(2):42{64,2013.

Vila, R. R., Andrés, S. M., & Guerrero, C. S. (2014). Evaluación de la calidad pedagógica de los MOOC. *Profesorado: Revista de curriculum y formación del profesorado*, 18(1), 27-41.

Wolfram, S. (2016). How to Teach Computational Thinking. *Blog Stephen Wolfram.* Retrieved from: http://blog.stephenwolfram.com/2016/09/how-to-teach-computational-thinking/

Zhang, J. X., Liu, L., Ordónez de Pablos, P., & She, J. (2014). The auxiliary role of information technology in teaching: Enhancing programming course using Alice. *International Journal of Engineering Education*, 30(3), 560–565.

Zhang, J. X., Ordónez de Pablos, P., & Zhu, H. (2012). The impact of second life on team learning outcomes from the perspective of IT capabilities. *International Journal of Engineering Education*, 28(6), 1388–1392.

Acknowledgment

The authors would like to express their gratitude to the members of Department of Computer Education of the Ministry of Education of the Dominican Republic (MINERD) for their collaboration, and to the members of RENATA and Ministry of Communications and Technologies (MINTIC) of Colombia.

This work was supported in part by the Research Development Grants of the University Basque System (2016-18), Department of Education, Universities and Research – Basque Government, Spain

- Computational Thinking trough object-based programming
- Human Cognitive Primitives: in search of new computational tools
- Blended Learning: added value for teachers and students
- Student Peer Review: new roles and responsibilities for students
- Mastery Learning through student portfolios and peer review