Serious game development as a creative learning experience: lessons learnt

Varvara Garneli
Ionian University
Corfu, Greece
c13garn@ionio.gr

Michail N. Giannakos
Norwegian University of Science and Technology
Trondheim, Norway
mgiannakos@acm.org

Konstantinos Chorianopoulos
Ionian University
Norwegian University of Science and Technology
Corfu, Greece
Trondheim, Norway
choko@acm.org

Letizia Jaccheri
Norwegian University of Science and Technology
Trondheim, Norway
letizia@idi.ntnu.no

***Abstract*—Computer programming skills in younger ages seem to be a promising and challenging aspect. Many visual programming tools have been developed in order to assist young students and to improve the current teaching practices and pedagogies. In this paper, we explore the potential effects of a Project Based Learning (PjBL) approach in the field of computer programming. In particular, we try to identify potential differences on students' programming habits/styles, between game development, simulation, and traditional learning strategy; in the context of PjBL. Our sample consisted of 53 middle school students who formed three groups. Introducing programming through a serious game development approach might inspire students towards a creative learning experience. Moreover, parameters like the class formation might affect the development of programming skills. The students in the PjBL treatment were able to complete a project successfully making fewer mistakes. On the other hand, students enrolled in a more traditional top-down approach chose to experiment with more complex curriculum but not always successfully**

***Index Terms*— Computer programming, pedagogy, project based learning, visual programming languages, CS in Schools, educational context, serious game development**

I. Introduction

There are several approaches aiming at motivating learners through creative programming activities. Game design and development [13] and programmable hardware platforms [6] are such examples. The game design and development could be used in order to motivate and enhance learning in Software Engineering and Computer Science topics. Such successful implementations include programming, artificial intelligence, software architecture, or object oriented design concepts and skills [30]. In particular, learning programming concepts in a video game context could be an enjoyable experience [2][13]. Additionally, programming competency could also promote learning in more domains. A science context, for example, could contribute in achieving a better understanding of complex science concepts [23][24]. Exploring the potential effects of a serious game development approach in the school setting could give useful guidelines to the educators.

Digital literacy is not just the ability to use technological tools like a word processor, a spreadsheet program or a presentation manager. An important skill of the digital age could be the ability to "construct" and "create" meaningful things using computers [16][18]. Implementing programming lessons from younger ages in the typical school environment could help on achieving this goal. The benefits for young students would be more than technical skills only. Such examples are Computational Thinking [31], Critical Thinking [32] and Creativity skills [17].

Introducing programming to young children is a very challenging process. The early programming languages for example were very difficult, mostly because of their syntax and the unfriendly user interfaces and editors. Recently, several visual environments have been developed and used in Computer Science Education (CSE) with good results. They can support, for example, a better understanding of programming concepts, like sequences. They also help students to understand the interactivity between the different parts of a program [30]. Alice, Scratch, and Greenfoot [29] are some of the most widely used programming languages in K12 CSE. Additionally, various e - programming tools could also support educators and learners in CSE and especially programming [33]. Their friendly interfaces and features aiming at making programming engaging to everyone [17]. Nevertheless, age, gender, type of activity and goal should be carefully considered when choosing the appropriate tool [29].

The teaching approach could also significantly affect the effectiveness of the introductory programming by preventing potential difficulties / misconceptions and engaging students [21]. There is not just one way that can be used in teaching programming skills successfully. There must be alternative techniques depending on the various students' needs and expectations. The studies that have been conducted on this field suggest different approaches which meet different student needs [21]. Project Based Learning (PjBL) for example is a pedagogical technique that is used very often in CSE. Many educators have designed and successfully applied PjBL approaches and published their experiences. There are many suggestions on parameters like motivation, problem generation or presentation, limitations, expectations etc. in order to achieve better results [20][21]. PjBL is usually chosen due to pedagogical grounds, to foster learning by doing and student independence in knowledge construction. Students expected to think critically and creatively [14].

In this study, we aim at exploring the benefits and limitations of PjBL approach in K-12 CSE. In particular, we suggest different instructional approaches in order to facilitate deep and creative learning in the field of computer programming.

978-1-4673-7046-2/15 $31.00 © 2015 IEEE
DOI 10.1109/GAS.2015.14

The rest of the paper is structured as follows: in the second section related work and research questions are presented, in the third section is described the methodology used to carry out the didactic intervention and follows a discussion on the results. Finally, we present conclusions of the reported research.

II. BACKGROUND AND RESEARCH QUESTIONS

A. Learning Programming in a Video Game Development Context

Modern technological tools and applications like video games could be incorporated in the typical school environment, in order to stimulate and enrich learning [8]. In particular, video games could be considered as virtual experiences in which players solve problems, and achieve learning and mastery through pleasant activities [5]. When a video game is not primarily used for entertainment then it could be defined as a serious game [26]. Some types of students could be benefit by alternative pedagogies such as serious games [4]. Pex4Fun, for example, is a web-based serious game which is used for educational purposes. In particular, students can edit their text code in any browser in order to check the code execution and be informed about its analysis [27]. While video games and serious games could be significant learning mediums, game design and development is an approach which could be based on a constructivism / constructionism perspective. Computers could be considered as the "construction materials" of video games. While students design and develop their own video games, learning can be achieved through active exploration, experimentation, discussion, and reflection [18]. Students' active improvement is encouraged [9]. Additionally, game design and development could be considered as an enjoyable learning experience which supports the deep learning of computer science concepts [13][1]. Moreover, higher-order thinking, abstraction skills [2] and self-confidence [1] can be enhanced. Similarly in a computer simulation approach, scientific concepts could be represented through programming in order to support profound learning. An approach like this requires some fluency with the programming language and the relevant domain knowledge [23][24]. From this perspective, the development of programming and computational modeling practices could also be supported [24]. Moreover, integrating modeling, programming, and physics for example could promote deep understandings in science concepts [24]. A video game, though, could be considered as a simulation which allows the active participation of the player through the presence of an avatar [5]. When the main goal of a video game has educational nature, it can be considered as a serious game [26]. Under this perspective, more research should be done in the field of the serious game design and development, due to the potential benefits on learners programming habits and skills.

So, the first research question is: *Could a serious game development approach successfully influence the students' programming habits, within a PjBL context?*

B. Programming Lessons Using a PjBL Instructional Approach in a Visual Programming Environment

There are several programming tools which could support a serious video game development approach. Scratch for example is a visual tool that aims at teaching programming skills in a constructionist way. Supporting "tinker ability", it provides a command palette which support exploration. Additionally to various command (move - steps), function (mouse …) and trigger blocks (when – key pressed) which are included in this palette, there is also a number of structures like conditionals (if-else) or loops (repeat, forever) [11]. Programming is done by dragging blocks from the palette into the scripting pane. Children can experiment with different combinations of blocks which can fit together only if they make syntactic sense. A stack of blocks can be triggered by startup or a given key pressed. A project can be consisted of many sprites. Each sprite has its own scripts, costumes, or sounds [3][11][17].

On the other hand, PjBL is a learner–centered instructional approach widely used in CSE. Knowledge is not just transmitted to the students by the teacher but discovered with his help [19]. Practically, firstly a problem is presented to the students. Then, students are critically thinking and working for the solution of the problem. Finally, they report their results. Additionally, a problem can be given periodically in sub problems [14].

Many parameters should be carefully considered while implementing a PjBL approach. According to Richards (2009), project type (e.g. an industry or a made – up), various group features (e.g. size), group management and students motivation are such examples. Moreover, several pedagogic styles could also influence the learning process. An encouraging style, for example, might promote more positive attitudes towards computer programming and more self-confidence than traditional instruction [10]. Additionally, a class could be formed with a traditional top – down approach, in which teacher first gives the theory and then the problem, in order to allow students to put the theory into practice. Alternatively, in a bottom-up approach, teacher introduces the theory within a project framework [19]. Despite the several project based approaches in the field of CSE, there is no clear evidence of the advantages and limitations of class formation, while teaching programming skills, using a visual programming tool. More research should be done in order to assist educators in designing learning activities based on the needs of their students.

Hence, the second research question is: *which are the benefits of a bottom-up PjBL approach in a computer visual programming context?*

In order to investigate the aforementioned research questions, we formed up two groups which were taught the same programming curricula in a project framework (bottom up). One group was taught through a science simulation project and the second one through a game development project based on science concepts. For the needs of this research, we also formed another group (control) which was taught the same

programming curricula with a more conventional approach (top down).

III. METHODOLOGY

A. Research Design

In this paper, we will explore the advantages and limitations of a serious game development approach in CSE. Moreover, we will investigate the potential effects of the class formation in a PjBL instructional approach. For the needs of this research, we formed up three different groups. All groups were taught the same programming curricula. The same definitions and examples were used in order to explain the various programming concepts. The instructional methods which have been used though were different. One group was taught with a traditional top – down approach. For every new programming concept, a project was given to the students in order to put the theory into practice (control group). The other two groups were taught with a bottom – up approach; students were working on a project that has been given to them from the beginning; periodically, new instructions were helping them in solving various sub problems (experimental groups) [14]. Both experimental groups were working on the same science based project with one difference. One group was simulating a phenomenon while the other group was working on the same phenomenon from a game development perspective.

There are several programming tools which could be used in this didactic intervention. We chose to use a visual programming tool, the Scratch Programming Environment due to its easiness for young students and its connection to the principles of constructionism [17]. All students though had already used scratch to make storytelling applications, the year before. According to their teachers, they were familiar with the usage of sequential code (say/for/sec, switch costume to, move/steps, and turn/degrees).

All lessons were based on the constructionism principals [15]. First, new concepts and examples were presented by the teacher. Then instructions were given to the students in order to complete their projects. Students also had the choice to request knowledge according to their own preferences and take their own decisions in designing or coding. Help and extra knowledge was provided by the teachers [3].

B. Sampling and Processes

We performed a between group experiment with 66 students, 14 to 15 years old. The participants were students of the third grade of the Gymnasium (middle school) and formed three groups. We followed the school's distribution in classes as we wanted to keep students into their ordinary environment. Students were working in pairs of their own choice. Some students, though, did not manage to attend all classes due to the longitudinal nature of the research. Thus, we had to remove them from our data analysis. Our results were based on fifty three students (26 boys and 27 girls) who attended all classes. They formed three groups, the control group and the two experimental ones, which engaged with the three respective approaches. The control group was consisted of 18 students, 13 boys and 5 girls, the simulation group was consisted of 17 students, 7 boys and 10 girls, and the game development group was consisted of 17 students, 6 boys and 12 girls.

While designing this intervention, we decided to base our programming curriculum in the philosophy of the in time pedagogy [12]. Under this perspective, we did not follow a sequential presentation of the various concepts. Instead the knowledge was introduced when needed. In Table 2, there is a quick list of the programming curricula.

TABLE I. PROGRAMMING CURRICULA

1.	Coordination and synchronization
2.	Loops and Pen commands for designing
3.	Conditionals, event handlers, and sensing
4.	Variables
5.	Operators for numerical (and boolean) values

We asked the experimental groups to be engaged with a project concerning the function of an electric circuit. We chose this curriculum as students had already attended relevant lessons, in the physics class, the previous month. Students' projects should include one battery and one switch to turn it on or off. When the switch is on electrons and positive ions move inside the circuit, so an electric lamp turns also on. Both experimental groups should present the function of an electric circuit but through different perspectives, a simulation and a game development one. The simulation group was encouraged to represent the circuit functions in order to help someone to study it. On the other hand, the video game development group was encouraged to copy the circuit's functions to a video game for educational purposes also (Figure 1).

Fig. 1. Screenshots from a simulation and a video game

In the beginning of the project, a small storytelling was giving the description and motivation of it.

At the same time, the control group was taught the same programming curriculum. The projects which have been produced by the students were according the curriculum nature. Additionally, projects' futures were similar with the ones of the experimental groups. For example, while the experimental groups' students were designing their circuits using pen primitives, the control group's students designed geometrical shapes (Figure 2).

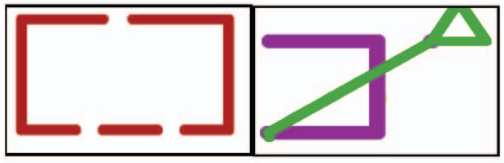

Fig. 2. Experimental and control group designs using pen primitives

C. Procedure

Firstly, the students were informed that they'll attend programming lessons using the Scratch Environment. The experimental groups students were additionally informed that the lessons will be conducted in a project framework and a small description was given to them. Then, they created a project according to their prior programming skills (pre-test). All the groups completed 5 sessions of 90 minutes each. During these sessions, students were taught the same programming curricula and practiced with the three different approaches. In particular, the control group's students were creating projects in order to practice in the several curricula concepts and the experimental groups' students were using the new knowledge in order to complete the given project. All the students were encouraged to freely decide the design and coding of their projects. Extra help and knowledge was given to the ones who needed. In the end of the sessions, students were asked to complete another programming task using the acquired programming knowledge (post test).

D. Measuring Instruments

We employed a pre-test to examine the students' programming habits before the learning process. In particular, we asked students to create a project in scratch based on their prior programming skills. After the end of the teaching intervention, students were asked to create another project (post test). Pre and post tests lasted one didactic hour each. Based on the primitive categories (Table 2), we rated the results in order to be able to identify any potential differences in coding habits between groups.

TABLE II. PRIMITIVES' CATEGORIES

CATEGORIES ***(example code)***
Coordination and synchronization (*broadcast message or when I receive message*) **Event handlers** (*when-clicked*)
Loops*(repeat, forever)*
Conditionals*(if-else)*
Sensing *(touching)*
Variables*(change-by)*
Operators *(=, <, >, and, random etc.)*
Sequence *(move-steps, say etc.)*

Additionally, based on students' work during the didactic intervention, we counted how many students had successfully completed their work.

E. Data Analysis

As we have already mentioned, 53 students participated in this research. We wanted to find potential differences in their programming habits between the pre and post-tests.

In order to assess how the regarding approaches enabled students with certain programming skills and habits, we counted the primitives' usage for certain categories (Table 2) in all pre and post-tests. Then, we computed the number of all primitives that were used in each project. Finally, we did not examine whether primitives were used correctly, just the total number of errors that had been made [3].

A non-parametric Wilcoxon / Exact signed-rank test was applied to the data. After calculating the difference (post – pre), a Mann-Whitney U test was applied in order to find potential differences between the groups.

Moreover, we assessed the students' code during the didactic intervention. Our goal was to distinguish how many students managed to complete their project successfully. Additionally, we gathered information from the teachers' observations in order to validate our results.

F. Results

We used a quantitative method to analyze the results from the pre and post tests. More particularly, our results were the primitives that had been chosen by the students in order to complete a project in scratch. First, we entered in IBM SPSS statistics Version 20 how many times students used each one of the primitive categories (Table 2) in their projects. Then, we entered the number of the total number of primitives used in each project and the total number of errors. After, we conducted a Wilcoxon / Exact signed-rank test. Our results indicated many differences between the groups in the post phase of the experiment.

While working on the post tests projects, the students of the control group experimented with more conditionals, variables, operators, and sequence primitives. Additionally, they wrote more code. The usage of coordination, synchronization, and event handlers' primitives didn't change significantly. Some students had already used an amount of the above in their pre tests. On the other hand, students did not improve the usage of loops and sensing primitives significantly.

The simulation group students chose to use more loops and conditionals in comparison with their pre test. Similarly, some students of this group used sequence, coordination, synchronization, and event handlers' primitives in their pre tests. They did not improve the usage of variables, operators, and sensing primitives significantly, though. In addition, this group didn't write more code in the post tests.

Finally, the game development group students made progress in the usage of loops, coordination, synchronization, event handlers, and sequence primitives. In addition, they showed evidence of improvement in the conditionals and sensing primitives' usage but not in the variables and operators.

The conclusion of the above results is that the control group mostly experimented with more complex programming curricula while the game development group was improved in all the primitive categories except for the more complex ones. Their projects, though, had fewer errors. Finally, the simulation group had the less improvement in the code produced after the didactic intervention by the other two groups.

TABLE III. TESTING THE PRE AND POST TESTS

	Control Group (N=18)						Simulation Group (N=17)						Game development Group (N=18)					
Primitive Categories	Pre Test Mean	Pre Test Std. SD	Post Test Mean	Post Test SD	Z	P (<.05)	Pre Test Mean	Pre Test SD	Post Test Mean	Post Test SD	Z	P (<.05)	Pre Test Mean	Pre Test SD	Post Test Mean	Post Test SD	Z	P (<.05)
Loops	0.61	1.19	1.39	1.68	1.43	**0.227**	1.94	2.79	3.56	4.97	-2.14	**.032***	0.00	0.00	2.00	3.97	-2.21	**.031***
Conditionals	0.67	1.198	4.50	7.52	-2.91	**.004***	1.648	2.83	3.44	4.76	-2.46	**.014***	0.00	0.00	1.82	3.97	-1.83	0.068
Coordination. Synchronization and Event Handlers	11.89	7.75	12.89	8.81	-0.64	**0.522**	9.53	5.82	9.82	8.92	-0.27	**0.549**	12.77	7.22	21.47	13.86	-2.64	**.021***
Variables	0.28	1.18	2.94	2.62	-2.77	**.006***	0.47	1.37	1.25	3.57	-1.60	**0.109**	0.00	0.00	2.06	4.72	-1.60	**0.109**
Sensing	0.72	1.49	3.28	7.64	-1.73	**0.180**	1.71	2.87	2.75	4.01	-1.70	**0.727**	0.00	0.00	1.76	3.80	-1.84	**0.066**
Operators	0.00	0.00	1.67	1.61	-3.21	**.000***	0.00	0.00	0.56	1.41	-1.60	**0.109**	0.00	0.00	0.29	0.77	-1.63	**0.102**
Sequence	11.94	6.14	18.61	11.63	-2.62	**0.009**	12.12	8.15	13.76	14.60	-0.25	**0.806**	12.00	8.08	21.41	15.89	-3.20	**.001***
Total	26.11	13.19	45.28	26.70	-3.36	**.001***	27.41	20.97	34.47	40.13	-0.66	**0.510**	24.76	14.93	50.82	41.45	-3.20	**.001***
Errors	0.06	0.246	1.11	1.08	-2.98	**.003***	0.12	0.33	0.25	0.58	-1.41	**0.157**	0.18	0.53	0.06	0.24	-0.82	**0.414**

*** at 0.05 level of significance; SD, Standard Deviation; Z, Wilcoxon / Exact signed-rank**

Then, we calculated the difference between pre and post tests. We applied a Kruskal-Wallis H test with a post hoc test which is used in order to determine statistically significant differences between two or more groups of an independent variable on a continuous or ordinal dependent variable. By doing that we attempt to investigate which treatment (group) result higher and significant shift on students' programming habits

TABLE IV. DIFFERENCES BETWEEN TREATMENTS (GROUPS)

	Control Group (N=18)	Simulation Group (N=17)	Game Development Group (N=17)		
Primitive Categories	Mean Rank	Mean Rank	Mean Rank	H	P(<.05)
Loops	24,86	27,34	25,94	0,259	0,879
Conditionals	30,06	25,75	21,94	3,020	0,221
Coordination. Synchronization and Event Handlers	23,75	22,47	33,44	5,390	0,068
Variables	**33,17***	21,38	22,76	8,441	**0,015****
Sensing	26,5	23,72	26,16	0,443	0,801
Operators	**34,97***	21,69	20,56	13,588	**0,001****
Sequence	**28,64***	19,03	**31,71***	6,524	**0,038****
Total	**30,11***	18,24	**30,94***	7,550	**0,023****
Errors	**35,47***	22,69	19,09	17,509	**0,000****

***Significant High performance;**at 0.05 level of significance; H Kruskal-Wallis test**

According to our results (Table 4), there is a statistically significant difference between the game development group and the simulation one, concerning the usage of sequence and total number of primitives. The game development group looks more productive and engaged with the programming activity after the didactic intervention. Statistically significant difference is additionally found in the control group as they used more variables and operators and made more errors in comparison with the simulation and the video game development groups. The control group seems to experiment more with complex curricula like the variables and operators but not always successfully. Moreover, the control group wrote more code than the simulation group but not than the game development one. Finally, the simulation group did not show better programming habits compared with both, the control and the game development group

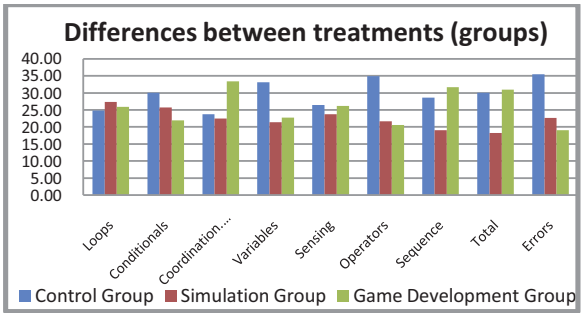

Fig. 3. Differences between treatments (Mean ranks)

The pre and post tests have been used in order to identify possible changes occurred in the students' programming habits. However, we did not measure the students' performance after the didactic intervention. In order to have a better understanding of the findings, we additionally assessed the code which was produced by the students during the intervention. In particular, we triangulated our findings with the number of students who successfully completed their projects and the teachers' observations.

TABLE V. STUDENTS' PROJECTS DURING THE INTERVENTION

Students' project/s	Control Group's Students	Simulation Group's Students	Game Development Group's Students
Successfully completed	89%	100%	65%

According to our findings (Table 5), the simulation group created the most effective code with no errors, during the intervention. On the other hand, 35% of the game development group's students did not manage to complete the requested tasks successfully. It is possible that students focused more on the game and the game development process and not on the programming concepts [30]. Finally, 11% of the control group's students did not manage to complete their work.

Concluding, despite the good performance of the simulation group, the programming habits of the students in the post tests were not very promising. Students used fewer and no complex programming concepts. Video game development group students did not have good performance during the intervention. Nevertheless, the same students chose to develop more code in their post tests. It is possible that the video game approach had positive influence on their programming habits. Moreover, the top down philosophy gave the students of the control group the inspiration to continue practicing with more complicated programming concepts such as the variables and the operators. Their projects, though, were not always without errors.

Based on teachers' observations the experimental groups students were more stressed in the beginning. They were demanding more help, especially, when they had coding errors. They were though satisfied when they managed to complete their projects successfully, but also felt a little tired in the end of the process. The control group also completed successfully the most of the tasks. Many students tried to create original and beautiful projects. They were working more independently and then they wanted to share their work with others

IV. DISCUSSION

A. Learning Programming in a Video Game Development Context

Our research findings suggest that a game programming context might provide an enjoyable experience which stimulates learning [13].This learning setting could additionally promote the creative expression of students [18]. Besides, a learning experience which encourages students to put their ideas into a project could be very positive. Through this creative expression, learners continue to be stakeholders of the project, as they feel that they own a part of it [25]. Overall, combining science and programming concepts in a serious game development context might be a very promising learning setting. Further research should be done in order to confirm and extend parameters like the acquired knowledge, in both domains.

Nevertheless, we should underline the successful completion of all projects made by the simulation group during the intervention. In this learning setting, students had the opportunity to learn and use effectively several computing concepts. They presented a project without errors and according the given instructions. Modeling and simulation could help students to understand both, computing and science concepts [23][24]. Further research could be done in order to make this learning setting more inspiring for the students.

B. Programming Lessons Using a PjBL Instructional Approach in a Visual Programming Environment

In this study, we tried to explore the effectiveness of a PjBL teaching approach, in which students were taught programming in a project framework. According to our research results, students who have been taught with a conventional top – down approach (control group) tend to experiment more with complex concepts. This could be explained by the class formation which gave students the opportunity of a better pedagogical explanation. On the other hand, students who have been taught in a project framework (experimental groups) might lose some complex theory components due to the pressure of solving the problem [19]. The control group's students, though, made more mistakes. Despite the small duration of this didactic intervention, the experimental group's students seem to think critically, making the appropriate decisions in order to successfully complete their projects [28]. It is obvious that both approaches could help students in different ways. That is to say, several instructional methods could support various students' needs [21].

V. CONCLUSION

Game development could successfully promote computing concepts in an enjoyable learning environment. Combining science concepts in a game development approach could be an interesting approach, which still motivates students in programming and encourages them towards a creative learning experience.

An approach which is based on a project framework could inspire some students in order to develop further skills. The free exploration might help students to learn how to think critically and to select and implement their knowledge in order to successfully complete a project. On the other hand, a top down instructional approach could be also useful as it could support the learning of complex programming concepts.

Students could benefit from all approaches gaining important skills. The appropriate instructional design of a class setting depends on the learning goals and the students' needs and expectations.

In summary, this study provides evidence for the students' programming habits / styles while implementing strategies like serious game development, science simulation or a more traditional learning. However, there are also some limitations. First, the generalizability of these results must be carefully approached since the field study was conducted in a specific context (e.g., curriculum, age). Extra methods which have been used like the teachers' observations and the number of students who managed to complete their tasks successfully, allow us to have a complimentary picture of the findings. The implications of this research concern the enrichment of the learning process with alternative methods which meet specific learning goals and deal with the various students' needs and background.

Further research should be done in order to confirm and extend the benefits of working in a serious game development context. It is important to identify the advantages and the limitations of this approach for both domains. Moreover, the skills which could be supported by the various instructional approaches should be identified clearly, in order to give useful guidelines to the instructors.

ACKNOWLEDGMENT

The authors would like to thank all of the students and the schools' staff for their participation in the didactic intervention.

REFERENCES

[1] M. Al-Bow, D. Austin, J. Edgington, R. Fajardo, J. Fishburn, C. Lara, and S. Meyer, "Using game creation for teaching computer programming to high school students and teachers," ACM SIGCSE Bulletin, Vol. 41(3), pp.104-108, 2009.

[2] M. Carbonaro, D. Szafron, M. Cutumisu, & J. Schaeffer, "Computer-game construction: A gender-neutral attractor to Computing," Computers& Education, vol. 55(3), pp. 1098-1111, 2010.

[3] A. Dahotre, Y. Zhang, and C. Scaffidi, "A qualitative study of animation programming in the wild," Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ACM, 2010.

[4] B. Garneli, M. N. Giannakos, K. Chorianopoulos, & L. Jaccheri, "Learning by Playing and Learning by Making," In Serious Games Development and Applications, Springer Berlin Heidelberg, pp. 76-85, 2013.

[5] J. P. Gee, "Learning and games. The ecology of games: Connecting youth, games, and learning," vol. 3, pp. 21-40, 2008.

[6] M. N. Giannakos, L. Jaccheri, and R. Proto, "Teaching Computer Science to Young Children through Creativity: Lessons Learned from the Case of Norway," Proceedings of the 3rd Computer Science Education Research Conference on Computer Science Education Research, Open Universiteit, Heerlen, 2013

[7] M. Høiseth, and L. Jaccheri, "Art and technology for young creators," In Entertainment Computing–ICEC 2011, pp. 210-221, Springer Berlin Heidelberg, 2011

[8] H. Y. Hsu, &S. K. Wang, "Using gaming literacies to cultivate new literacies," Simulation & Gaming, Vol. 41(3), pp. 400-417, Wiese Productions, 2010.

[9] Y. B. Kafai, 2006, "Playing and making games for learning instructionist and constructionist perspectives for game studies," Games and culture, Vol. 1(1), pp. 36-40, 2006.

[10] D. Makris, K. Euaggelopoulos, K. Chorianopoulos, & M. N. Giannakos, "Could you help me to change the variables?: comparing instruction to encouragement for teaching programming," In Proceedings of the 8th Workshop in Primary and Secondary Computing Education, pp. 79-82, ACM, 2013.

[11] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond, "The scratch programming language and environment," ACM Transactions on Computing Education (TOCE), Vol. 10(4), pp. 16, 2010.

[12] O. Meerbaum-Salant, M. Armoni, and M. Ben-Ari, "Learning computer science concepts with scratch," Computer Science Education, vol. 23(3), pp. 239-264, 2013.

[13] C.C. Navarrete, "Creative thinking in digital game design and development: A case study," Computers& Education Vol. 69, pp. 320-331, 2013.

[14] M. J. O'Grady, "Practical problem-based learning in computing education," ACM Transactions on Computing Education (TOCE) Vol.12.3: 10, 2012.

[15] S. Papert, and I. Harel, "Situating constructionism," Constructionism Vol. 36 : 1-11, 1991.

[16] S. Papert, "The Children's Machine: Rethinking School in the Age of the Computer," 1993.

[17] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond,K. Brennan, and Y. Kafai, "Scratch: programming for all," Communications of the ACM, Vol. 52(11), pp. 60-67, 2009.

[18] M. Resnick, "Rethinking learning in the digital age", 2002.

[19] D. Richards, "Designing project-based courses with a focus on group formation and assessment," ACM Transactions on Computing Education (TOCE) Vol. 9.1: 2, 2009.

[20] R. Romeike, "What's my challenge? The forgotten part of problem solving in computer science education," Informatics Education-Supporting Computational Thinking, Springer Berlin Heidelberg, pp.122-133, 2008.

[21] M. Saeli, J. Perrenet, W. M. Jochems, and B. Zwaneveld, "Teaching programming in secondary school: a pedagogical content knowledge perspective," Informatics in Education-An International Journal, Vol. 10 1, pp. 73-88, 2011.

[22] J. R. Savery, and Th. M. Duffy, "Problem based learning: An instructional model," Vol. 35 5 pp. 31-38, 1995.

[23] P. Sengupta, J. S. Kinnebrew, S. Basu, G. Biswas, and D. Clark, "Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework," Education and Information Technologies, vol. 18(2), pp. 351-380, 2013.

[24] P. Sengupta, and A. V. Farris, "Learning kinematics in elementary grades using agent-based computational modeling: a visual programming-based approach," In Proceedings of the 11th International Conference on Interaction Design and Children, pp. 78-87, ACM, June 2012.

[25] T. Smith, K. M. Cooper, and C. S. Longstreet, "Software engineering senior design course: experiences with agile game development in a capstone project ," In Proceedings of the 1st International Workshop on Games and Software Engineering, pp. 9-12, ACM, May 2011.

[26] T. Susi, M. Johannesson, and P. Backlund (2007). Serious games: An overview.

[27] N. Tillmann, J. De Halleux, T. Xie, & J. Bishop, "Pex4Fun: Teaching and learning computer science via social gaming," In Software Engineering Education and Training (CSEE&T), 2012 IEEE 25th Conference on pp. 90-91, IEEE, April 2012.

[28] A. Tiwari, P. Lai, M. So, and K. Yuen, "A comparison of the effects of problem-based learning and lecturing on the development of students' critical thinking," Medical education, Vol. 40(6), pp. 547-554, 2006.

[29] I. Utting, S. Cooper, M. Kölling, J. Maloney, and M. Resnick, "Alice, Greenfoot, and Scratch--a discussion," ACM Transactions on Computing Education (TOCE), Vol.10(4), pp. 17, 2010.

[30] A. I. Wang, & B. Wu, "The use of game development in comouter science and software engineering education,"

[31] J. M. Wing, "Computational thinking," Communications of the ACM, Vol. 49(3), pp. 33-35, 2006.

[32] http://www.criticalthinking.org/pages/the-national-council-for-excellence-in-critical-thinking/406.

[33] http://code.org/ .