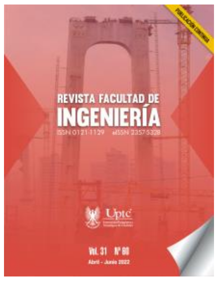
Revista Facultad de Ingeniería
Journal Homepage: https://revistas.uptc.edu.co/index.php/ingenieria



Strategy Based on Computer-Supported Collaborative Learning to Form Workgroups Automatically in an Introductory Programming Course (CS1)

José-Miguel Llanos-Mosquera[1]
Carlos-Giovanny Hidalgo-Suarez[2]
Víctor-Andrés Bucheli-Guerrero[3]

Received: April 18, 2022 **Accepted:** August 02, 2022 **Published:** August 08, 2022

Citation:

J.-M. Llanos-Mosquera, C.-G. Hidalgo-Suarez, V.-A. Bucheli-Guerrero, "Strategy Based on Computer-Supported Collaborative Learning to Form Workgroups Automatically in an Introductory Programming Course (CS1)," *Revista Facultad de Ingeniería*, vol. 31 (61), e14368, 2022. https://doi.org/10.19053/01211129.v31.n61.2022.14368

[1] M. Sc. Universidad del Valle (Cali-Valle del Cauca, Colombia). jose.llanos@correounivalle.edu.co. ORCID: 0000-0003-4642-2770
[2] M. Sc. Universidad del Valle (Cali-Valle del Cauca, Colombia). carlos.hidalgo@correounivalle.edu.co. ORCID: 0000-0003-2308-0720
[3] Ph. D. Universidad del Valle (Cali-Valle del Cauca, Colombia). victor.bucheli@correounivalle.edu.co. ORCID: 0000-0002-0885-8699

Abstract

Students of the Introduction to Programming courses present low grades and this is reflected by the high failure and academic desertion rates. In this sense, looking for ways to improve and support the student's academic performance in the Fundamentals of Object-Oriented Programming (FPOO) course, a strategy based on Computer-Supported Collaborative Learning (CSCL) is proposed. It is supported by an algorithm to form workgroups automatically, which allows motivating students and creates a uniform criterion to develop programming tasks. Under the framework of the quasi-experimental design, the strategy was implemented in different evaluative activities of the FPOO course, which allowed us to answer questions related to the improvement of a student's final grade using the automatic formation of workgroups versus the traditional formation of workgroups. Moreover, we compared the grades obtained when activities are developed with no group formation. The experiments in this paper show that using the collaborative strategy improves students' grades by 22% in the laboratories and by 20% in the final project. In addition, it allows the exchange of knowledge to solve a programming task. This paper concludes that developing strategies that integrate collaboration positively impacts the programming learning process and improves student grades and people skills significantly, which encourage a better learning in programming courses.

Keywords: automatic code assessment; automatic formation of workgroups; collaborative learning; computer programming; technology and education.

Estrategia basada en la metodología Computer-Supported Collaborative Learning para la formación de grupos de trabajo automáticos en un curso de introducción a la programación (CS1)

Resumen

Los cursos de Introducción a la programación presentan bajas calificaciones de los estudiantes, esto se refleja en las altas tasas de mortalidad y deserción académica. En este sentido, buscando formas de mejorar y apoyar el rendimiento académico de los estudiantes del curso CS1 - Fundamentos de Programación Orientada a Objetos (FPOO), este artículo propone una estrategia basada en la metodología

Computer-Supported Collaborative Learning (CSCL) apoyada por un algoritmo para la formación de grupos de trabajo automáticos, que busca motivar a los estudiantes y permite adquirir conocimientos de forma homogénea en el desarrollo de actividades de programación. Bajo el marco del diseño cuasi experimental, se implementó la estrategia para diferentes actividades evaluativas en el curso FPOO, que permitió responder cuestiones relacionadas con la mejora de la calificación final de un estudiante utilizando la formación de grupos de trabajo automáticos en comparación a la formación de grupos de trabajo tradicional, y los resultados que se generan en las calificaciones cuando se desarrollan actividades sin formación de grupos. Los experimentos de este trabajo demuestran que el uso de la estrategia de colaboración mejora las calificaciones de los estudiantes en 22% en laboratorios y 20% en el proyecto final. Además, permite intercambiar conocimientos para resolver una tarea de programación. Finalmente, en este trabajo se concluye que el desarrollo de estrategias que integran la colaboración impacta positivamente en el proceso de aprendizaje de programación, mejorando significativamente las calificaciones del estudiante y las habilidades interpersonales que incentivan a mejorar el aprendizaje en los cursos de programación.

Palabras claves: Aprendizaje colaborativo; educación tecnológica; evaluación automática de código fuente; formación automática de grupos; programación de computadores.

Estratégia baseada na metodologia Computer-Supported Collaborative Learning para a formação de grupos de trabalho automáticos em um curso introdutório à programação (CS1)

Resumo

Os cursos de Introdução à Programação apresentam baixas notas dos alunos, isso se reflete nas altas taxas de mortalidade e deserção acadêmica. Nesse sentido, buscando formas de melhorar e apoiar o desempenho acadêmico dos alunos do curso CS1 - Fundamentos de Programação Orientada a Objetos (FPOO), este artigo propõe uma estratégia baseada na metodologia Computer-Supported Collaborative Learning (CSCL) apoiada por um algoritmo para a formação de grupos de trabalho automáticos, que procura motivar os alunos e permitir-lhes adquirir conhecimentos de forma homogénea no desenvolvimento das atividades de programação. No quadro do desenho quase-experimental, foi implementada a estratégia de diferentes atividades avaliativas no curso da FPOO, que permitiu responder a questões relacionadas com a melhoria da nota final de um aluno através da formação de grupos de trabalho automáticos em comparação com a formação de grupos de trabalho tradicionais e os resultados que são gerados nas capacitações quando as atividades são desenvolvidas sem a formação de grupos. Os experimentos deste trabalho mostram que o uso da estratégia colaborativa melhora as notas dos alunos em 22% nos laboratórios e 20% no projeto final. Além disso, permite a troca de conhecimento para resolver uma tarefa de programação. Por fim, este artigo conclui que o desenvolvimento de estratégias que integrem a colaboração tem um impacto positivo no processo de aprendizagem de programação, melhorando significativamente as notas dos alunos e as habilidades interpessoais que incentivam um melhor aprendizado nos cursos de programação.

Palavras-chave: Aprendizagem colaborativa; Educação Tecnológica; avaliação automática do código fonte; formação automática de grupos; Programação de computadores.

I. INTRODUCTION

Introduction to Programming (CS1) is a mandatory course for Systems Engineering students [1]. In this course, the students should acquire logical skills and apply them in a programming paradigm to solve computational problems through a programming language [2], [3]. The student demonstrates those skills through learning activities that measure their achievements in the programming course [4]-[7].

In the Systems Engineering program of Universidad del Valle [8], the traditional teaching-learning methodology of a programming course consists of face-to-face classes including theory and practice (laboratories) according to the content of the CS1 course. Students who pass this course continue to CS2; however, between 30% and 50% of students do not pass [9], [10]. Therefore, it is necessary to design learning strategies so that they improve their grades, achieve logical skills, stimulate collaborative work, and increase their motivation [11]-[13].

In recent years, in the educational context, great interest has been aroused in the design of collaborative tools that enhance the development of learning skills in students. In turn, they allow the emergence of other experiences and work dynamics within the classroom. This interest has started early in countries like the United States, England, and Australia, whose educational experiences reflect the benefits of working in groups compared to individual work [14]-[16].

Collaboration as a learning strategy in the classroom has yielded satisfactory results in programming courses, improving academic aspects and personal skills [17], [18]. However, identifying the individual work that each student contributes to a group programming activity is a complicated task for the teacher [19], [20]. Pereira mentions in [21] that one way to contribute to the problem is to develop tools to provide feedback based on the summative and formative evaluation.

In the literature, one of the most popular learning approaches is Computer Supported Collaborative Learning (CSCL) [18], [22], which emerges as support for traditional learning [15], [23] and seeks to control the learning process to ensure that students acquire knowledge collaboratively [24], [25]. This approach is based on the formation of groups and how these can be supported by technology to improve learning and teaching. These processes can be intervened to adapt to specific needs [26], [27].

This paper presents a strategy based on the CSCL methodology to form workgroups automatically in an introductory programming course (CS1). The purpose is to motivate students to develop programming skills and collaborate. Several experiments were conducted to prove that collaborative work improves academic grades compared to individual work.

This paper is organized as follows: Section 1 presents the related work; Section 2 describes the methodology, where the questions of interest, the selection of the CS1 course sample, and the phases of the experiments are presented in detail; Section 3 presents the results of the experiments; Section 4 discusses the results and concludes the work.

II. Related Work

CodeBench [20] is a system intended to include collaboration and evaluation of programming concepts. It is based on three stages: 1) task specification (what to test, how to test, coding plan); 2) distribution of tasks among group members (publish in a public repository, determine approval requirements, define the testers, define inputs and capture outputs, calculate grades; 3) testing the programs using the tool's automatic evaluator.

CourseMaker [21] is an early warning system designed to support the academic process. It employs learning analytics to categorize performance and group effort. Teachers use this tool to provide timely assistance to students performing poorly or at risk of failing in their classes. Groups are formed to provide targeted advice and support for multiple university students.

TRAKLA [22] is an automatic source code evaluation tool that enables group formation based on dynamic evaluation and feedback among students. This tool enables workgroup formation but does not evaluate the group source code.

I-MINDS [23] is a software for the intelligent management of online classrooms or groups. It enables programming activities to be reviewed in real time and offline, thus facilitating student practice. Its technology is based on intelligent agents that interact with users autonomously in a chatbot. Source code is evaluated using the virtual judge DOMJudge [24], where students submit their solutions to the posed programming problems and receive immediate feedback.

UNCode tool [25] has the option of grouping programming students according to three criteria: 1) by similar grades; 2) randomly; 3) applying the “pair programming” concept. The groups are assessed by the evaluation of the submitted source codes, which indicate whether the syntax and efficiency of the program are correct or have errors. Based on this evaluation, a group grade and an individual grade are assigned. Mechanisms such as static analysis and feedback with questionnaires (mini tests) are included; they help students to improve the proposed programming solutions.

III. METHODOLOGY

This paper implements an algorithm to form workgroups automatically from a learning activity in a CS1 programming course. The process was based on questions of interest, population and sample, data collection, implementation of the automatic grouping algorithm, and interpretation of the results.

A. Research Questions

This work is based on the course CS1: Fundamentals of Object-Oriented Programming (FPOO). This course has a high academic failure rate [9], [28]; therefore, it is important to integrate learning strategies based on collaboration and tools that help improving student’s academic performance [21], [29]-[31].

The strategy was implemented by answering the questions of interest. **RQ1:** Does the final grade of a student improve using the automatic formation of workgroups compared with traditional group formation? **RQ2:** What are the results in grades when activities are developed without group formation?

B. Population and Sample

The grades of 68 students were collected. They correspond to the laboratories, one exam, and the final project of the FPOO course in the second semester of 2021. This course is offered every semester at the School of Systems and Computing Engineering (EISC) of Universidad del Valle (Cali-Colombia). It consists of 4 hours of face-to-face class per week (2 theoretical and 2 practical), and 8 hours of autonomous work (development outside the classroom), for a total of 192 hours each semester (16 weeks).

In this course, students develop their skills in an object-oriented programming language. Learning outcomes are based on the design, development, documentation, and implementation of solutions that include object-oriented programming concepts (class, object, inheritance, polymorphism). Collaborative learning is based on the strengthening of attitudes toward teamwork and the integral development of the student in the cognitive and sociocognitive dimensions. The 68 students enrolled in the FPOO course were selected as sample for this study. They were randomly grouped in two: 34 students as a control group (CG) and 34 as an experimental group (EG). In the CG, 84.85% of students passed the course, and in the EG, 83.15%. The number of students and percentages achieved show that the groups are homogeneous (Table 1).

Table 1. Description of the groups (CG and EG) in the programming course (FPOO) in the second semester of 2021.

Course	Semester	# Students	Students who passed the course	Programming language	Course weeks
FPOO -CG	August - December	34	84.85%	C++	16
FPOO -EG		34	83.15%		

C. Automatic Formation of Workgroups Using CSCL

FPOO students must solve programming problems as a group. In this course, workgroups of 3 or 4 students were formed. However, the traditional way of grouping students has biases and is not equitable; this is evidenced in the course's final grades [32]. For this reason, an algorithm that allows creating automatic homogeneous groups has been developed from the learning activities. The strategy is described below.

D. Dataset

The data input of the algorithm to form workgroups automatically requires an initial learning activity that must be conducted individually. In this case, the grades obtained in laboratory 1 by the EG were used. 110 submissions were made through the M-IDEA automatic evaluation platform. On average, each student made 2 submissions. Grades are between 3.4 and 4.9 (on a scale from 0.0 to 5.0). Finally, the data array is generated with the grades obtained in laboratory 1 and the code of each student.

E. Algorithm for Automatic Formation of Workgroups

The algorithm takes the previously generated data array, students are then allocated a partner, the highest-performing students are paired with lowest-performing ones. Then, each pair is grouped with another, thus creating a group of four students. If there is an odd number of students in total, the algorithm takes the student who has not been assigned to a group and includes them in one of the previously defined groups randomly. The process uses the uniformity criterion, which consists of forming groups with the same number of students for the completion of activities. If the number of students not assigned to a group is equal to or less than three, the algorithm prompts the user to select whether to create a new group from this selection or distribute them among the existing groups (Figure 1).

Algorithm 1 Group Formation

```
process ← group_formation (corpus_data)
while corpus_data != NULL do
    if corpus_data(current tuple position) > corpus_data(next position in tuple) then:
        corpus_data ← sort records by grade value (descending)
    end if
end while
while corpus_data != NULL do
    if corpus_data == pair then:
        pair_programming ← pairs are created among the total number of students (highest and lowest rating)
    else
        pair_programming ← the student who does not belong to a group is randomly taken and included in a previously defined group (highest and lowest grade)
    end if
end while
if creation_groups == pair then:
    creation_groups ← creation of groups from two pairs (random selection)
else
    creation_groups ← creation of groups from two o more pairs (random selection with uniformity criterion)
end if
```

Fig 1. Pseudocode of the algorithm implemented to form workgroups automatically.

F. Experiments

Four experiments were conducted in this study using the grades obtained by the CG and EG students' submissions. The first experiment integrates laboratory 1; the second includes laboratories 2,3 and 4; the third includes the exam performed; and the latter integrates the final project (see Table 2). Each experiment is described below.

Table 2. Experiments made with the control group (CG) and the experimental group (EG).

Experiment	Tasks	CG	EG
1	Laboratory 1	Individually	Individually
2	Laboratories 2, 3, 4	Traditional group	Automatic group
3	Exam	Individually	Individually
4	Final project	Traditional group	Automatic group

1) Test 1 (lab 1). CG and EG students conduct the Laboratory 1 individually with a time limit of 2 hours (Table 3).

Table 3. Description of laboratory 1 of FPOO course.

Task	Description	Time of assessment	% of course
Lab 1	The learning objectives to be evaluated in this lab are the correct style of coding, the documentation and debugging of the source code in C++	2 hours	$\frac{\sum grade_n}{1} * 0.20$

2) Test 2 (lab 2, 3 and 4). Students conduct 3 laboratory activities with a time limit of 2 hours each (Table 4). The automatic formation of workgroups was used for the EG, and the traditional formation of groups was used for the CG (by affinity between students).

Table 4. Description of laboratories 2 and 3 of the FPOO course.

Task	Description	Time of assessment	% of course
Labs 2 and 3	Learning objectives focus on the use of standard libraries of the C++ programming language, inheritance, and source code refactoring	2 hours	$\frac{\sum grade_n}{1} * 0.10$
Lab 4	Learning objectives assess inheritance, polymorphism, and refactoring of source code	2 hours	$\frac{\sum grade_n}{1} * 0.20$

3) Test 2 (exam). In this experiment, EG and CG students take the exam individually with a time limit of 1 hour (Table 5). The exam for both groups consists of two components: the first one includes multiple-choice questions with a single answer, it allows evaluating the conceptual learning outcomes of the course. The second includes a programming exercise that the student must develop and submit in the INGInious M-IDEA automatic source code evaluation tool, which allows assessing the practical learning results of the course.

Table 5. FPOO Course Exam Description.

Task	Description	Time of assessment	% of course
Exam	Evaluates all the theoretical and practical concepts developed during the semester	1 hour	$\frac{\sum grade_n}{1} * 0.30$

4) Test 3 (final project). In this experiment, students develop the final project in groups. This activity has a deadline of 12 weeks counted from week 4, where the statement is socialized (Table 6). For the EG, the automatic formation of workgroups was used, while in the CG, it was the traditional formation of groups (by affinity between students).

Table 6. Description of the final project of the FPOO course.

Task	Description	Time of assessment	% of course
Final Project	With the development of the final project, we intended to evaluate the use of classes, objects, and relationships.	12 weeks	$\frac{\sum grade_n}{1} * 0.20$

III. RESULTS

This section presents the results obtained by the CG and EG in the experiments described in the methodology. Those elements allow us to answer the defined questions of interest. Subsection 4.1 presents the results obtained by the students in laboratories 2, 3, 4 and the final project, which were carried out with automatic and traditional formation of workgroups. Subsection 4.2 presents the grade obtained by the students in laboratory 1 and exam, developed without group formation. In subsection 4.3 the final grades of the CG and the EG are compared.

In the results, the median of the scores and the Mann-Whitney statistical test were used, which allows for comparing the mean for the variables used in the EG and CG, based on a null hypothesis H0. In the process, the p-value corresponding to the significance level is obtained, if the value found is less than or equal to 0.05, the null hypothesis is rejected because it is concluded that the mean between the EG and CG differs with a level of significance of 5%. But, if the p-value is more significant than 0.05, the null hypothesis is accepted, indicating that the mean value for the two groups does not differ significantly.

A. Qualification of Students Using Group Formation (Automatic and Traditional)

To respond to **RQ1**, the grades students obtained in the learning activities corresponding to laboratories 2, 3, 4 and the final project were analyzed. Activities were conducted with automatic and traditional group formation. Figure 2 shows the results obtained in these laboratories for the CG and EG. In laboratory 2, the CG presents a median of 3.9, and the EG of 4.8. In laboratory 3, the CG reached 3.0 in the median of the grades, and the EG obtained 4.9. Finally, in laboratory 4, the CG reached a median of 3.9, while the EG obtained 4.8.

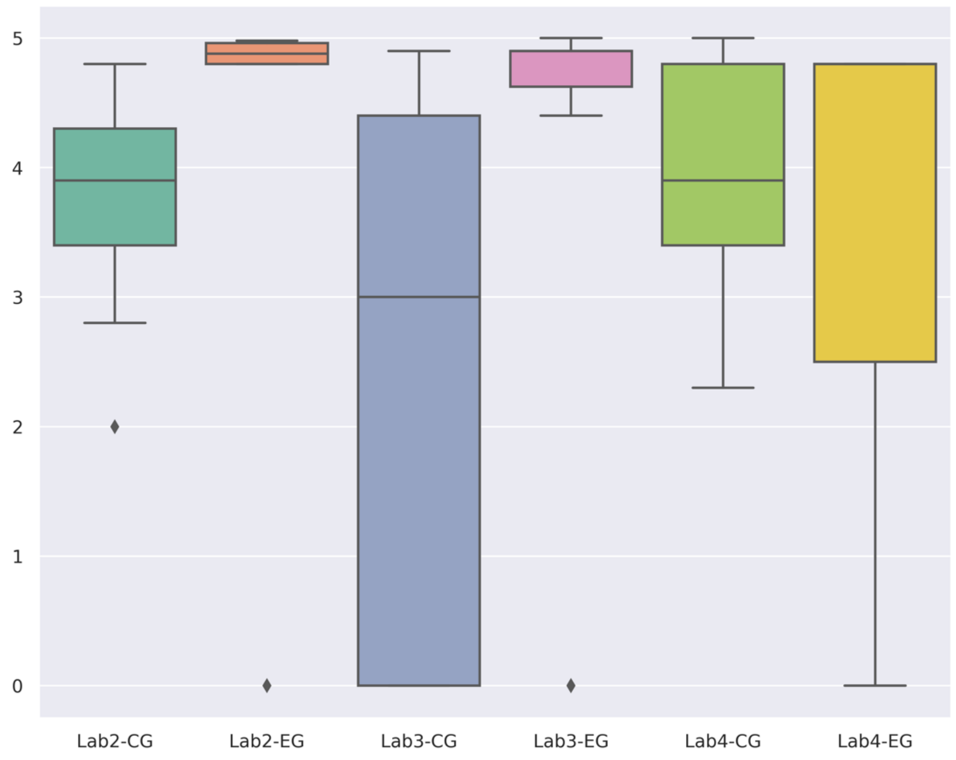


Fig 2. Results obtained in laboratories 2, 3 and 4 by the CG and EG.

The median of grades of the EG is higher than the CG in laboratories 2, 3 and 4, reaching an average of 4.8 and 3.6, respectively (on a scale of 0.0 to 5.0). This shows that the implemented automatic formation of workgroups has positive effects on programming activities. The learning objectives are related to correct coding style, documentation, source code debugging, use of standard libraries C++, inheritance, polymorphism, and source code refactoring. However, it is necessary to carry out other experiments to support this idea.

Figure 3 presents the results obtained in the final project. The CG reached a median of 4.0, while the EG obtained 5.0. With the results of the experiment, it was observed that the automatic formation of workgroups generated positive results in the EG, reaching higher grades compared to the CG. This indicates that the strategy can be used to conduct activities where the use of relationships between objects, polymorphism, property changes, controller class and extensibility are evaluated.

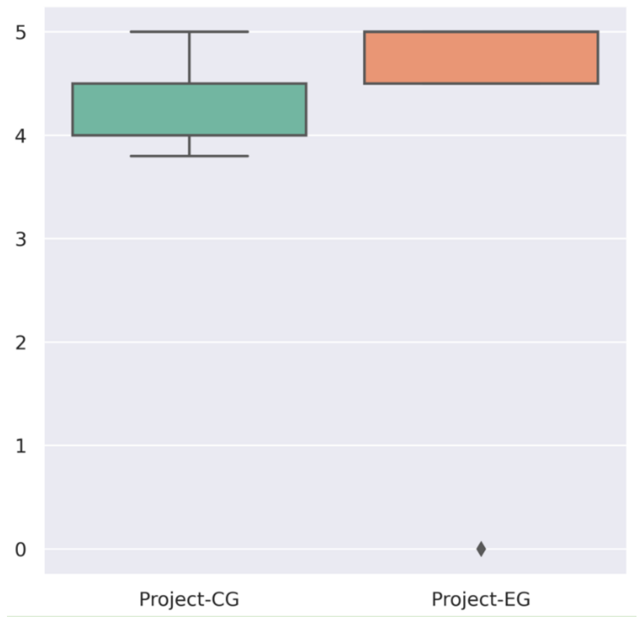


Fig 3. Results obtained in the final project by the CG and EG.

In labs 2, 3 and the final project, the p-value was 1.29e-09, 1.64e-06 and 0.12e-04, respectively. In this case, the null hypothesis is rejected, because the mean of the scores differs between the EG and CG with a significance level of 5%. However, in laboratory 4, the p-value was 0.69. In this case, the null hypothesis is accepted because the resulting value is greater than 0.05. Thus, it is possible to conclude that the mean of the grades is similar for the students of the two groups (EG and CG).

B. Students' Grades in Laboratory 1 and Exam (Without Group Formation)

To respond to **RQ2**, we analyzed the grades of the students in laboratory 1 and exam, conducted without group formation. In laboratory 1, the CG reached a median of 4.6, while that of the EG was 4.0. In this learning activity, where the correct style of coding, documentation, and debugging of the source code in C++ is evaluated, it is observed that the CG students can achieve higher grades by 12% compared to the EG (Figure 4).

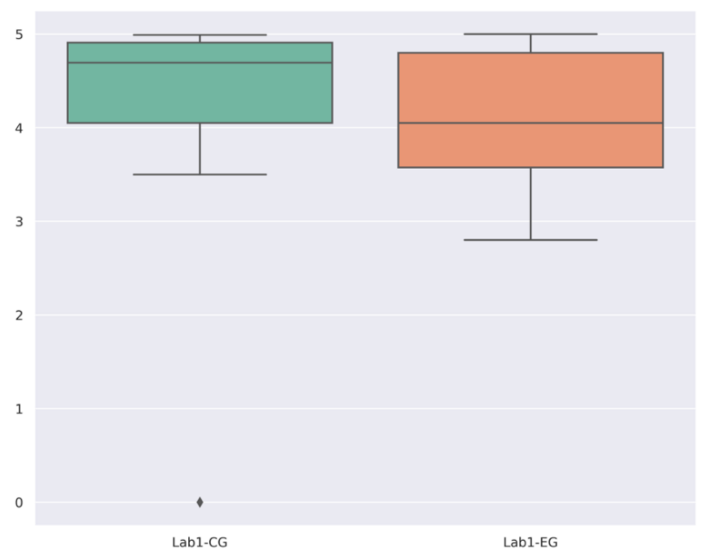


Fig 4. Student grades in lab 1.

In the exam, all the theoretical and practical concepts developed during the academic semester were evaluated. The CG reached a median of 4.2, while the EG obtained 4.0 (Figure 5). It is probable that students of the two groups reach this grade because the activity was developed individually (without group formation). However, it is necessary to conduct new experiments and activities to validate this argument.

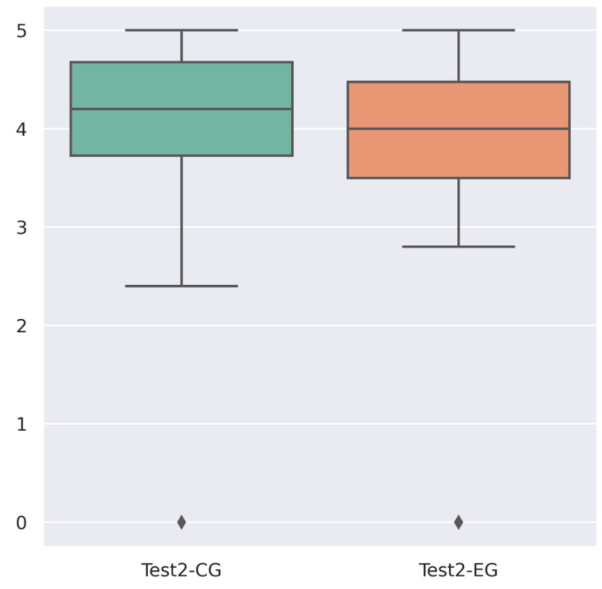


Fig. 5. Results obtained in the exam by the CG and EG.

In laboratory 1, the p-value was 0.50, and in the test, it was 0.20. In this case, the null hypothesis is accepted for the two learning activities because the resulting p-value is greater than 0.05, thus indicating that the mean of the grades is similar for the students of the two groups (EG and CG).

C. GPA Comparison of the CG and EG

Finally, the final median of the grades of all the learning activities defined in the CG and EG were compared. The CG reached a median of 3.1, while the EG reached a median of 4.7 (Figure 6).

The results indicate that the EG obtained better grades than the CG. This may be due to the implementation of the automatic formation of workgroups. It is necessary to conduct other experiments that allow us to discuss this idea.

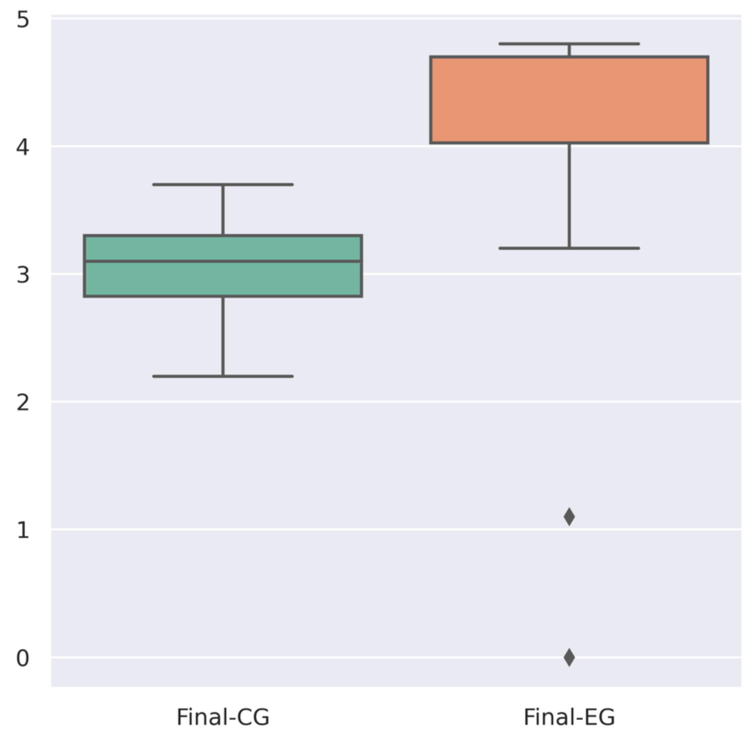


Fig. 6. Median of the grades of the CG and EG

The p-value was 1.34e-07, since the value is less than 0.05, the null hypothesis is rejected because the mean of the grades differs between the EG and CG students with a significance level of 5%.

IV. Discussion and Conclusions

This article presents a strategy based on the CSCL methodology to form workgroups automatically using a learning activity conducted in a programming course (CS1). For **RQ1**, it was determined that EG students improved their grades by 22% compared to the CG in programming activities related to correct coding style, documentation, source code debugging, inheritance, polymorphism, and code refactoring. We also observed that the EG improved their grades by 20% in relation to CG in the development of the final project. There, the use of relationships between objects, polymorphism, property changes, controller class, and extensibility were evaluated.

However, in laboratory 4, the mean grades for the two groups (EG and CG) are similar according to the Mann-Whitney statistical test. This allows us to confirm what was described by Böhne and Kardan in their investigative works [19], [20]. They mentioned that identifying the work each student contributes to the group in a programming activity is a complex task for the teacher.

When analyzing the activities without group formation, for **RQ2**, we observed that the CG achieved better results compared to the EG in laboratory 1 and the exam. In laboratory 1, the CG scores were 12% higher than the EG scores, while in the exam the CG was 4% higher than the EG. However, when comparing the final grade of all the activities in the EG, we observed that the results of laboratories 2, 3, 4 exceeded the results obtained in laboratory 1 and the exam by 16%, while the final project surpassed laboratory 1 and the exam by 20% (see Table 7). This shows that the automatic formation of workgroups implemented in laboratories 2, 3, 4 and the final project is effective for this type of activity.

Table 7. Final grades obtained by the CG and the EG in the experiments.

Tasks	CG Grade	EG Grade
Laboratory 1	4.6	4.0
Laboratories 2, 3, 4	3.7	4.8
Final project	4.0	5.0
Exam	4.2	4.0

Authors such as Pereira, Oliveira, and Fonseca in [21], [29]-[31], mention that it is important to integrate learning strategies in automatic source code evaluation tools because they can improve the academic performance. In the INGInious M-IDEA source code evaluation tool used in our experiments, automatic feedback was generated on each release. This made it possible to monitor the learning process, and to identify whether the students applied the programming concepts in the source code and corrected errors. Likewise, the learning results of the course were more homogeneous. Group programming skills were also stimulated, which helped improve final grades through the automatic formation of workgroups.

The design of strategies that integrate collaboration, learning analytics, and technological tools significantly improve student grades, thus cancelling the possibility that only one student ends up doing the work of the whole group. In addition, it improves people skills that encourage sharing and enjoying learning computer programming.

Authors' Contribution

José-Miguel Llanos-Mosquera: Research, results, writing - review and editing.
Carlos-Giovanny Hidalgo-Suarez: Research, methodology, writing - review and editing.
Víctor-Andrés Bucheli-Guerrero: Research, validation.

Funding

We especially thank MinCiencias and Fundación CEIBA-Nariño for financing the doctoral studies of José Miguel Llanos Mosquera and Carlos Giovanny Hidalgo Suarez.

References

[1] M. Sahami , S. Roach, E. Cuadros-Vargas, A. Danyluk, R. Dodge, K. Fisher, A. Thompson, Computer Science Curricula 2013, vol. 1. IEEE Computer Society, 2013.

[2] A. Ali, D. Smith, "Teaching an Introductory Programming Language in a General Education Course," *Journal of Information Technology Education: Innovations in Practice*, vol. 13, pp. 57–67, 2014.

[3] G. Alexandron, M. Armoni, M. Gordon, D. Harel, "The effect of previous programming experience on the learning of scenario-based programming," in *Proceedings of the 12th Koli Calling International Conference on Computing Education Research*, New York, NY, USA, Nov. 2012, pp. 151–159. https://doi.org/10.1145/2401796.2401821

[4] M. McCracken , V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. Kolikant, T. Wilusz, "A Multi-national, Multi-institutional Study of Assessment of Programming Skills of First-year CS Students," in *Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, New York, NY, USA, 2001, pp. 125–180. https://doi.org/10.1145/572133.572137

[5] S. Billis, O. Cubenas, "Assessing Collaborative Learning with E-Tools in Engineering and Computer Science Programs," *Advances in Intelligent Systems and Computing*, vol. 1070, pp. 848–854, 2020. https://doi.org/10.1007/978-3-030-32523-7_62

[6] S. I. Malik, "Improvements in Introductory Programming Course: Action Research Insights and Outcomes,"

Systemic Practice and Action Research, vol. 31, no. 6, pp. 637–656, 2018. https://doi.org/10.1007/s11213-018-9446-y

[7] L. Carvajal-Ortiz, B. Florian-Gaviria, J. F. Díaz, “Models, methods and software prototype to support the design, evaluation, and analysis in the curriculum management of competency-based for higher education,” in *XLV Latin American Computing Conference (CLEI)*, Panama, 2019, pp. 1-10. https://doi.org/10.1109/CLEI47609.2019.235114

[8] Universidad del Valle, *Reforma Currícular - Facultad de Ingeniería*, 2020. http://ingenieria.univalle.edu.co/reforma-curricular

[9] C. Watson, F. W. B. Li, “Failure rates in introductory programming revisited,” in *Proceedings of the Conference on Innovation & Technology in Computer Science Education*, New York, NY, USA, 2014, pp. 39–44. https://doi.org/10.1145/2591708.2591749

[10] J. Bennedsen, M. E. Caspersen, “Failure rates in introductory programming: 12 years later,” in *ACM inroads*, 2019, pp. 30–36. https://doi.org/10.1145/3324888 .

[11] L.-K. Soh, N. Khandaker, X. Liu, H. Jiang, “A computer-supported cooperative learning system with multiagent intelligence,” in *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, New York, New York, USA, 2006, e1556. https://doi.org/10.1145/1160633.1160933

[12] M. L. Séin-Echaluce, Á. Fidalgo Blanco, F. J. García-Peñalvo, M. Á. Conde, “A Knowledge Management System to Classify Social Educational Resources Within a Subject Using Teamwork Techniques”, in *International Conference on Learning and Collaboration Technologies*, 2015, pp. 510-519. https://doi.org/10.1007/978-3-319-20609-7_48 .

[13] C. Alvarado, C. B. Lee, G. Gillespie, “New CS1 pedagogies and curriculum, the same success factors?,” in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, New York, New York, USA, 2014, pp. 379–384. https://doi.org/10.1145/2538862.2538897

[14] A. R. Denham, R. Mayben, T. Boman, “Integrating Game-Based Learning Initiative: Increasing the Usage of Game-Based Learning Within K-12 Classrooms Through Professional Learning Groups,” *TechTrends*, vol. 60, no. 1, pp. 70–76, 2016. https://doi.org/10.1007/s11528-015-0019-y

[15] G. Lee, W. W. Fong, J. Gordon, “Blended learning: The view is different from student, teacher, or institution perspective,” *Lecture Notes in Computer Science*, vol. 8038, pp. 356–363, 2013. https://doi.org/10.1007/978-3-642-39750-9_33

[16] A. J. Lakanen, V. Isomöttönen, “High school students’ perspective to university CS1,” in *Annual Conference on Innovation and Technology in Computer Science Education*, New York, N. Y., USA, 2013, pp. 261–266. https://doi.org/10.1145/2462476.2465585

[17] Y.-H. Chen, W.-C. Lee, C.-H. Tseng, L. Y. Deng, C.-Y. Chang, L.-H. Lee, “Cognitive learning performance assessment and analysis with CSCL applied on the NetGuru platform and CSPL applied on the TAoD platform for the network experiment class,” *Journal of Supercomputing*, vol. 76, no. 1, pp. 16–46, 2020. https://doi.org/10.1007/s11227-019-02836-3

[18] Z. Mehennaoui, Y. Lafifi, H. Seridi, A. Boudria, “A new approach for grouping learners in CSCL systems,” in *Proceedings International Conference on Multimedia Computing and Systems*, 2014, pp. 628–632. https://doi.org/10.1109/ICMCS.2014.6911143

[19] A. Böhne, N. Faltin, B. Wagner, “Distributed group work in a remote programming laboratory - A comparative study,” *International Journal of Engineering Education*, vol. 23, no. 1, pp. 162–170, 2007

[20] A. Kardan, H. Sadeghi, "Modeling the learner group formation problem in computer-supported collaborative learning using mathematical programming," in *8th National and the 5th International Conference on e-Learning and e-Teaching,* IEEE, 2014, pp. 1-5. https://doi.org/10.1109/ICELET.2014.7040616 .

[21] F. D. Pereira , E. Oliveira, A. Cristea, D. Fernandes, L. Silva, G. Aguiar, M. Alshehri, "Early Dropout Prediction for Programming Courses Supported by Online Judges," in *Artificial Intelligence in Education*, 2019, pp. 67–72. https://doi.org/10.1007/978-3-030-23207-8_13 .

[22] G. Stahl, T. Koschmann, D. Suthers, *Computer-supported collaborative learning: An historical perspective*, 2006. https://www.semanticscholar.org/paper/Computer-supported-collaborative-learning-%3A-An-Stahl-Koschmann/00c0825352eded96cdc99fefcbfb52be4c6a796e

[23] M. N. Demaidi, M. Qamhieh, A. Afeefi, "Applying Blended Learning in Programming Courses," *IEEE Access*, vol. 7, pp. 156824–156833, 2019. https://doi.org/10.1109/ACCESS.2019.2949927

[24] G. Ayala, M. Ortíz, M. Osorio, "Agent modelling for CSCL environments using answer sets programming," in *Proceedings of the Mexican International Conference on Computer Science*, 2005, pp. 214–221. https://doi.org/10.1109/ENC.2005.9

[25] J. Lämsä, P. Uribe, A. Jiménez, D. Caballero, R. Hämäläinen, R. Araya, "Deep Networks for Collaboration Analytics: Promoting Automatic Analysis of Face-to-Face Interaction in the Context of Inquiry-Based Learning," *Journal of Learning Analytics*, vol. 8, no. 1, e1, 2021. https://doi.org/10.18608/jla.2021.7118

[26] J. Chen, M. Wang, P. A. Kirschner, C.-C. Tsai, "The Role of Collaboration, Computer Use, Learning Environments, and Supporting Strategies in CSCL: A Meta-Analysis," *Review of Educational Research*, vol. 88, no. 6, pp. 799–843, 2018. https://doi.org/10.3102/0034654318791584

[27] M. Coto, S. Mora, C. Collazos, "Evaluation of the collaboration process from an individual and collaborative perspective," in *ACM International Conference Proceeding Series*, 2014, pp. 1–9. https://doi.org/10.1145/2662253.2662342

[28] J. Bennedsen, M. E. Caspersen, "Failure Rates in Introductory Programming," in *AcM SIGcSE Bulletin*, 2007, pp. 32-36. https://doi.org/10.1145/1272848.1272879.

[29] F. D. Pereira, E. H. T. Oliveira, D. Fernandes, A. Cristea, "Early Performance Prediction for CS1 Course Students using a Combination of Machine Learning and an Evolutionary Algorithm," in IEEE 19th International Conference on Advanced Learning Technologies, Brazil, 2019, pp. 183–184. https://doi.org/10.1109/ICALT.2019.00066

[30] F. D. Pereira, S. C. Fonseca, E. H. T. Oliveira, D. B. F. Oliveira, A. I. Cristea, L. S. G. Carvalho, "Deep learning for early performance prediction of introductory programming students: a comparative and explanatory study," *RBIE*, vol. 28, pp. 723–748, 2020. https://doi.org/10.5753/rbie.2020.28.0.723

[31] F. D. Pereira, S. C. Fonseca, E. H. Oliveira, A. I. Cristea, H. Bellhäuser, L. Rodrigues, L. S. Carvalho , "Explaining Individual and Collective Programming Students' Behavior by Interpreting a Black-Box Predictive Model," *IEEE Access*, vol. 9, pp. 117097–117119, 2021. https://doi.org/10.1109/ACCESS.2021.3105956

[32] Y. S. Lin, Y. C. Chang, C. P. Chu, "Novel Approach to Facilitating Tradeoff Multi-Objective Grouping Optimization," in *IEEE Transactions on Learning Technologies*, 2015, pp. 107-119. https://doi.org/10.1109/TLT.2015.2471995.