electronics

MDPI

Article

Assessment in Software Development for Competitive Environments: An AI Strategy Development Case Study

Manuel Palomo-Duarte [1,*], Antonio García-Domínguez [2] and Antonio Balderas [1]

[1] Departamento de Ingeniería Informática, Universidad de Cádiz, 11519 Puerto Real, Spain; antonio.balderas@uca.es
[2] School of Engineering and Physical Sciences, Aston University, Birmingham B4 7ET, UK; a.garcia-dominguez@aston.ac.uk
* Correspondence: manuel.palomo@uca.es

Abstract: Competitions are being widely used to motivate students in diverse learning processes, including those in computer programming. This paper presents a methodology for designing and assessing competitive learning scenarios that allow students to develop three different coding skills: the ability to compete against unknown competitors, the ability to compete against known competitors and the ability to compete against refined versions of known competitors. The proposal is based on peer code review, implemented as an improvement cycle after the dissemination of the code among participants. A case study evaluating the methodology was conducted with two cohorts of students in an undergraduate course. The analysis of the obtained grades suggests that while performance after our assistance was improved, students could still fail or succeed independently of the assistance. Complementary data from student questionnaires and supervisor observations are aligned with this finding. As a conclusion, the evidence supports the validity of the methodology. Additionally, several guidelines based on the experience are provided to transfer the proposal to other environments.

Keywords: computer science education; programming; competition-based learning; assessment; open source software

Citation: Palomo-Duarte, M.; García-Domíngue, A.; Balderas, A. Assessment in Software Development for Competitive Environments: An AI Strategy Development Case Study. *Electronics* **2021**, *10*, 1566. https://doi.org/10.3390/electronics10131566

Academic Editor: George Angelos Papadopoulos

Received: 6 May 2021
Accepted: 25 June 2021
Published: 29 June 2021

1. Introduction

Higher education has to develop both specific and generic skills in their students, in response to the stringent demands of the current job market [1,2]. These skills will make the difference between their own success and that of their competitors, which may be known or not. Even if those skills are known, rapidly evolving markets may require preempting competitors in order to stay in business. Students can benefit from practicing these competitive scenarios in advance in well-defined learning environments [3]. Among other benefits, competitive learning can boost motivation while avoiding some of the problems (e.g., plagiarism [4]) of other grading methods [5]. Software engineering students in particular have the benefit over other more "tangible" engineering disciplines that programs are highly malleable and new versions can be quickly produced, accelerating these competition cycles [6].

Farr et al. [7] pointed out that the leaderships' attitude is fundamental to facing competitive environments in engineering contexts. Schoemaker et al. [8] identified six skills that allow leaders to think strategically and navigate the unknown effectively: the abilities to anticipate, challenge, interpret, decide, align, and learn. However, according to Siewiorek et al., they are difficult to train and learn in a formal educational context [9].

Competitive methods are usually deployed on games [10,11]. However, the rankings they commonly generate are only comparative and do not fit into regular qualification schemes. They do not actually assess students' learning outcomes, but their performances compared to their peers. Simply turning rankings directly into grades could raise issues,

since students would not be assessed according to what they learned from the experience [12].

The research question derived from this context was: *can a multi-level software development learning method provide evidence of the students' competitive skills?*. To answer the research question, we created a methodology that aligns these abilities with three skills of the participants in competition:

1. The ability to surpass unknown competitors (ASUC): in this case we work on the ability to anticipate the strategies of our rivals may be, in order to beat them.
2. The ability to surpass known competitors (ASKC): once we know the strategies of our rivals, we can interpret them, align them with ours, and decide how to act.
3. The ability to surpass improved competitors (ASIC) or the ability to improve in a new generation of evolving known competitors: this implies learning from one's own mistakes and from the successes and mistakes of one's rivals.

The contribution of this paper is a methodology for designing competitive software development learning scenarios which train and assess the three aforementioned skills using objective data. The proposal consists of a programming competition divided into several rounds, allowing students to refine their strategies between rounds. The learning assessment considers not only competition against peers, but also improvements in the performance of each student's strategy. This way, students are required to find weak points in their own strategies and adopt the best parts of the strategies implemented by their colleagues, thereby rewarding asynchronous and implicit collaboration.

A case study was designed and implemented using the proposed approach as part of a university-based computer science course, based on the principles of design-based research [13] and the existing literature on competition-based learning. The experience was supported by a purpose-built environment, available as open-source software, where students programmed rule-based expert systems to compete in a predefined board game. The results obtained by the students in the case study for the three competitive skills mentioned above were complemented with the students' answers to a final questionnaire on the method. Both sources of information have been analyzed and complemented with the observations of the supervisor compiled during the experience. Insights are discussed in the case study section.

The state of the art is discussed in the following section. Next, the research design section presents the proposed competitive evaluation methodology and a description of the case study's evaluation. Then, the case study is analyzed and discussed in the results section. Finally, some conclusions and future lines of work are outlined.

2. The State of the Art

While the first proposals of challenge-based learning were available in 2008 [14], it has not received significant attention in academia until recent years [15]. Although there is no consensus among the different approaches to challenge-based learning [16], according to Johnson and Adams, it is a multidisciplinary approach to education that encourages students to leverage the technology they use in their daily lives to solve real world problems [17]. It has proved to be an effective approach not only for teaching both specific and soft skills, but also for increasing diversity and inclusiveness in the design of solutions to complex problems [18].

Literature shows that competitive learning structures bring their own challenges. Slavin mentioned two key points to take into account [19]:

- Firstly, competition can be a highly contingent reward structure, to the degree that competitors are evenly matched, and success depends on effort.
- Secondly, within a win–lose competition, one student's success necessitates another student's failure.

These points require looking into ways to structure the competition. Researchers and practitioners have demonstrated that students working in a competitive environment

can get better results than those working in a traditional one [20,21]. However, in such a learning environment, students can lose motivation because of the nature of the competition. If certain students are frequently beaten by most of their competitors, it can be difficult for them to demonstrate their aptitude. The main issues derived from competition found by Muñoz-Merino et al. [22] are disappointment and loss of confidence and interest, lack of efficacy, and a poor attitude toward mistakes. The same authors developed a tool to support competitions that comprised several rounds [23]. By using this tool, students are dynamically paired to compete against peers with similar scores, avoiding negative emotions in them caused by the aforementioned issues. However, the experiment found several students who, despite feeling motivated, had very poor performance.

Another proposal using the same game we used for our case study was introduced in [24]. It proposes a betting system as a solution for students lacking the programming skills necessary for competition. Students review the code of other students before the competition takes place, and are invited to compose their grades from their own strategies and those of colleagues that they consider superior. This is formalized as a single-round competition methodology to specifically work and assess the skill "critical analysis of the code written by peers", which is different from the skills proposed in this paper.

Similar experiences that use competitions to motivate students in programming courses have been found in the literature [25,26], but unfortunately only a few of them considered multiple rounds for assessment. Arevalillo-Herráez et al. conducted an experiment where strategies for the first competitions were programmed in groups and those of the second round were refinements of them by each member. In the second round, students with better programming skills could advise colleagues with limited programming skills, thereby getting extra points for improving the performances of other strategies. While interesting, the proposal raises the question of the actual role of the advisers in the second round, as providing advice to average colleagues and trusting their supposedly improved skills seems less attractive that directly programming for them [27]. A second related study with two serialized competitions, so competitors could refine their systems, was conducted by Paneda et al. [28]. Students' proposals (but not the source code) were shared with the rest of the colleagues after the second round. Unfortunately, both papers include little information on the assessment method and no empirical data. Besides this fact, an important difference from the current proposal is the code sharing between peers that should exercise their code reading skills.

Peer code review has demonstrated improvements in students' learning outcomes regarding several abilities, such as programming skills, collaborative learning, compliance with coding standards, time management, and giving and accepting criticism [29]. Therefore, the application of different phases of coding and peer code review in a competitive scenario could provide teachers with a number of ways to apply in enriched assessments of students' learning outcomes.

3. Research Design

The research design that follows is a combination of a software system development methodology and a research methodology, based on the principles of design-based research [13], which is a "systematic but flexible methodology aimed to improve educational practices through iterative analysis, design, development, and implementation, based on collaboration among researchers and practitioners in real-world settings." Thus, the assessment intervention was inspired by these principles.

To be effective, interventions should be able to migrate from experimental classrooms to average classrooms operated by average students, supported by realistic combinations of technologies. The intervention may be a learning activity, a type of assessment, or a technological intervention. In this paper, the novel intervention is the competitive assessment methodology that was applied in a computer programming course.

3.1. Competitive Assessment Methodology

The design of the competitive assessment intervention includes the following phases (Figure 1):

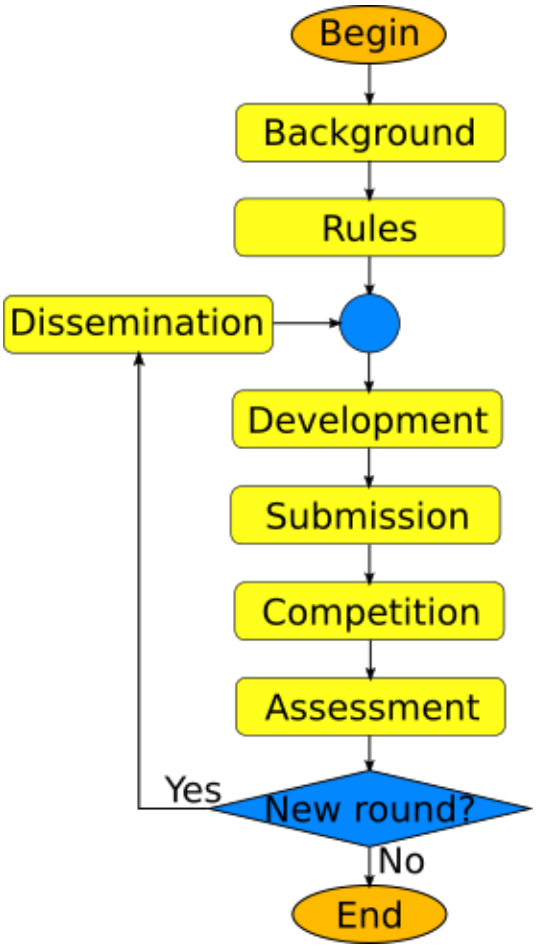

Figure 1. Proposed methodology.

1. Background: First, students need the proper theoretical/practical background to face the competition. Instructors must provide lectures or laboratory sessions depending on the prior skills of the students and the competition settings (programming language, game structure, and so on).
2. Competition rules: Students must know the rules of the competition they will face. This includes the aim of the competition, any support that will be provided (e.g., a simulation environment or some examples), whether they can play in teams, what measures are to be taken to prevent plagiarism between rounds, and so on. The assessment scheme must be also publicly known.
3. Initial development: After analyzing the game, the students design and code their strategies. This can be done in a laboratory (ideally, with the support of a supervisor), at home, or using a mixed approach.
4. Initial submission: Before the deadline, students must submit their strategies to participate in the competition.
5. Initial competition round: The first round of the competition is played. If the kind (and scale) of competition allows for it, providing real-time graphical feedback on the performances of the strategies can be both engaging and formative for the students.
6. Initial assessment: The supervisor collects assessment metrics of the experience and discusses with students the key aspects that set the strategies apart, encouraging self-criticism.

7. Initial dissemination: If desired, the supervisor can take actions so students actively share their approaches. Among other things, students could have a forum to share their code snippets explaining their strategies, they could work in teams to analyze the code of colleagues and write reports, and/or the source code of the programs could be publicly available.
8. Improvement development (first refinement): After playing the first competition, students must identify bugs and weak points in their strategies and refine them.
9. Improvement submission: There is then a new deadline for sending improved strategies.
10. Improvement competition round: When all the refined strategies are submitted, a new competition must take place in two parts.
    (a) Firstly, a customized competition round is allowed for each strategy. In it, the refined strategies are pitted against the original strategies to measure their relative performance improvements. For instance, each refined strategy could take part in a round with the other original strategies. Alternatively, each refined strategy could be pitted only against its own original version, against the strategies that the original version lost to, against a reference strategy, against the best strategies, and so on. This way, different metrics can be obtained for fine-grained assessment. Note that, depending on the settings, this phase can take far longer than previous ones. In that case, it could be performed in batch mode away from the classroom.
    (b) Secondly, a new competition round with all the refined strategies takes place.
11. Improvement assessment: The supervisor collects additional metrics to improve the assessment of the three skills. A final reflection on the experience is conducted with the whole group.

Phases 7–11 can be repeated as a block to enable more than two competitive rounds if time allows for it. For each new round, new metrics would be obtained by pitting the improved and previous versions of the strategies against each other.

3.2. Evaluation

The research evaluation was carried out through a case study, according to the information systems research classification of Hevner et al. [30]. The experimental setting was an intervention with the competitive assessment strategy proposed in this paper. The intervention was included in the development of a rule-based software module by students, which followed a traditional software development process based on analysis, design, and software construction [31].

The research question was answered in our case using data from complementary sources: the supervisor and the students. The observations of the supervisor were compiled during the whole experience by both person-to-person and virtual interactions. As for the information from the students, a mixed approach was used, combining objective data from the competition, a questionnaire, and notes from informal comments/reactions.

3.3. Participants

The implementation of the competitive assessment intervention was analyzed in two cohorts of students undertaking a video game design course at the University of Cadiz (Spain; 31 students in total). It was offered as an elective course for students of its computer science degree in their third year. The first cohort consisted of 15 students; the second one had 16 students. In the artificial intelligence lessons, students participated in the competition individually as an assessment. A tight restriction was that all work for this assessment had to be done within two weeks (i.e., two hours of lectures and two of laboratory practice each week, plus six hours of independent work), so the game and programming language were carefully chosen to be easy to code for.

3.4. Instruments

The competition involved a simple board game based on Stratego. Two players took turns at moving one piece: a player could win by eliminating the piece of the other player with the lowest value. The game was slightly modified in cohort 2 to discourage direct copying of agents from cohort 1 and provide more depth to the game: an obstacle was placed randomly on the field, and had to be avoided. An open source software tool implementing the game, named Gades Siege (http://drpantera.github.io/gsiege accessed on 28 June 2021), was specifically developed for the competition. It accepts files in the CLIPS/JESS programming language with the players' strategies. It can then run leagues or play-offs with the various strategies, and allows for step-by-step execution of single matches and massive unattended execution of a batch of matches (Figure 2 shows a screenshot of a match).

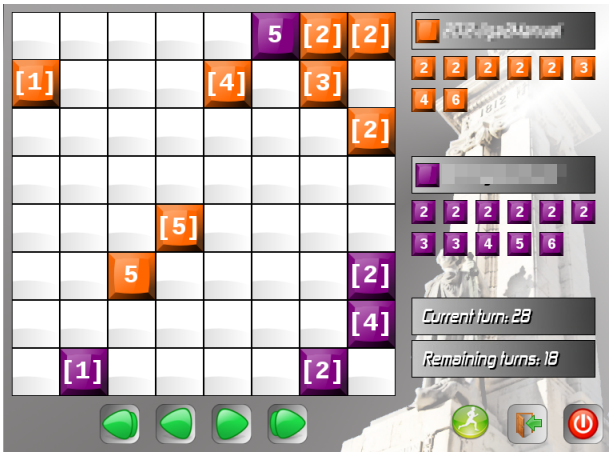

Figure 2. A screenshot of a match.

To keep the programming task simple, each team is controlled by only one agent. This way, the strategy of each player can be easily programmed as a rule-based expert system (also known as production system) [32]. Each expert system is implemented as a set of prioritized if–then conditional rules that are executed according to information from the current status of the game. Coding is as simple as adding, modifying, removing, or re-prioritizing rules. This way, a wide variety of behaviors that can be implemented by making slight changes in a rule set.

The game provides partial knowledge of the world for each player, so there can hardly be unbeatable agents. This way, agents start playing the game with most of the information unavailable. As the game goes on, players collect more information about their opponent and can apply more specific behaviors. Therefore, a good balance between general and specific rules is required (but not sufficient) to create a winning agent.

3.5. Course Design

In order to base the intervention on a realistic situation, a number of introductory lectures about rule-based expert systems programming were provided, and a simulation of the design and development of a game was designed as a scaled-down dramatization of actual computer science design and artificial intelligence issues [33]. The challenge of bounding the temporal scope of the interventions was partially solved by developing a multi-year design [13]. Nevertheless, the strict temporal bounds for each academic year limited the number of iterations available to provide changes and improvements of the learning process. For this reason, the redesign of the process has included only two iterations per year: first designing a competitive strategy and programming the rules for a first version of the software system; and then analyzing the code of their peers, proposing a refined design, and programming a second solution to be delivered. The interventions used an implicit cooperative learning strategy in which students criticize their peers' designs before starting the competitive assessment phase.

3.6. Implementation

The undergraduate course included the competitive assessment intervention according to the proposed design:

1. Background: As the course was elective, students with different backgrounds enrolled. To provide a common ground, a proper theoretical/practical formation on rule-based expert systems programming was provided.
2. Competition rules: The grading system was described in the virtual learning environment. The game platform used to enact the competition was freely available for students, who could install it on their computers and run different competitions with the provided sample strategies. Students were warned that they could only change up to 30% of their code in the refined version.
3. Initial development: Students had two hours to start developing their strategies in the laboratory with the help of the supervisor. Most queries were about programming syntax errors and rule-based programming concepts. Students could continue coding at home for another 5 days.
4. Initial submission: Students uploaded their strategies to a handout activity of the virtual learning environment.
5. Initial competition round: The first round of the competition was played. It consisted of 10 round-robin leagues played with different random seeds because strategies usually include non-deterministic behavior, so results could vary in each league. Some matches were projected in real time on a big screen (Figure 3).
6. Initial assessment: The supervisor collected assessment metrics of the experience and informally commented with students the key aspects that set apart the strategies. While only some students acknowledged the strong points of the best strategies, most of them agreed on the weak points of their own strategies (they usually were quite obvious).
7. Dissemination: A dedicated forum in the virtual learning platform was set up to have students explain their designs to their peers. Those forum posts were complemented with the comments included in the code of the strategies, which usually described low-level design implementation details. This was done to show that students knew the general ideas behind each strategy, and could help them to review code to get winning behavior. Additionally, the lecturer disseminated the source code of the strategies.
8. Second development (refinement): After playing the first round, students had one hour to identify bugs and weak points in their strategies and refine them. They had a limited period of time and could only modify a certain amount of code (30%). Despite the limited time, the behavior of a rule-based expert system could be easily changed by re-prioritizing, adding, or removing rules, so the time was enough to tune the strategies.
9. Second submission: Students uploaded their improved strategies to a new handout activity of the virtual learning environment before the deadline.
10. Improvement competition round: When all the refined strategies were submitted, the new competition took place in two parts.
    (a) Firstly, every improved strategy was pitted versus the original versions of the other strategies in 10 round-robin leagues (using different random seeds)
    (b) Secondly, a new competitive round of 10 round-robin leagues took place, again with different random seeds.
11. Improved assessment: The supervisor collected additional metrics to conduct the assessment of the three skills. A more informal final reflection on the experience was conducted with the whole group. The number of students that acknowledged the strong points of the best strategies significantly increased, and the overall comments about the experience were very positive.

Figure 3. Students watching a match.

The assessment was conducted by using the ratios of points obtained by individual students to those attainable in each league. Let (A_{OO}, P_{OO}) be the pair of attainable and obtained points in the set of leagues with all the original strategies. Likewise, let (A_{RR}, P_{RR}) be the equivalent pair for the leagues with all the refined strategies and let (A_{RO}, P_{RO}) be the pair for the leagues with one refined strategy and the rest of the original strategies. These values can be combined to produce one grade for each scenario:

- Against unknown competitors: $G_{OO} = P_{OO}/A_{OO}$.
- Against known competitors: $G_{RO} = P_{RO}/A_{RO}$.
- Against refined versions of known competitors: $G_{RR} = P_{RR}/A_{RR}$.

3.7. Metrics

The proposed assessment scheme uses three sources of objective information: the first competition with the original strategies, the second one that confronts each refined version with the rest of the original ones, and the last one with all the refined strategies.

Depending on the game context, different metrics can be used to assess each skill. As an example, the grade for each source can be derived from how many points were obtained out of all the attainable credit points in a league. Another option is using certain key performance indicators from the game: the values of the pieces captured in a chess match, the goal difference in a football match, etc. In fact, even internal program metrics can be extracted if considered of interest, such as program size and efficiency. If there is a random factor in the competitions, playing several rounds for each phase can provide a more meaningful result.

4. Results

The research question is answered using data from complementary sources: first, the objective data obtained during the competition are shown. That is followed by an analysis of the answers to the student questionnaire and the comments from students. Finally, the perspective of the supervisor is discussed.

4.1. Data Perspective

Figures 4 (cohort 1) and 5 (cohort 2) show the ratios of achievable points that were obtained by each of the students' strategies in the different settings. Students are ordered by grade in the first round, so the *X* axis shows the position they achieved in the ranking.

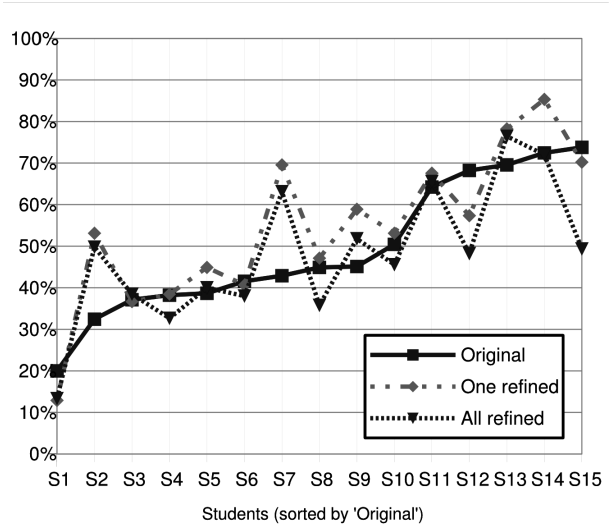

Figure 4. Ratio of achievable points actually obtained by round type for each student in cohort 1.

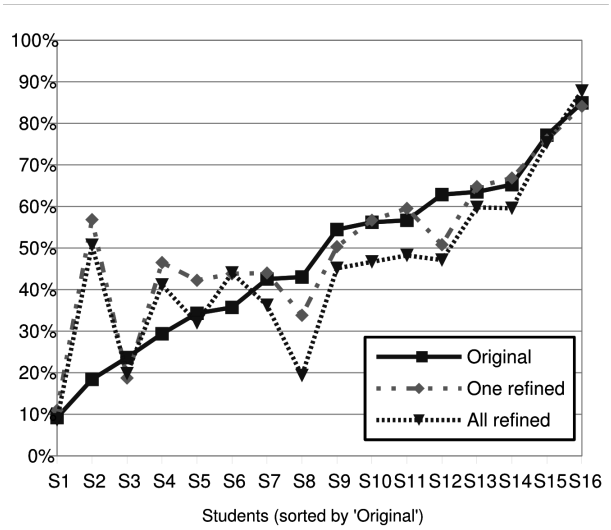

Figure 5. Ratio of achievable points actually obtained by round type for each student in cohort 2.

Additionally, the correlation coefficients of the ratios of achievable points were calculated (Tables 1 and 2). The coefficients between the original agent's performance and the improved agent's performance range between 0.73 and 0.83. This suggests that while performance by the improved agent is related to the performance of the original one, it is not a 1:1 relation and students can still fail or succeed independently of it.

Table 1. Correlation coefficients (cohort 1).

	Original	**One Refined**	**All Refined**
Original	1.00	0.73	0.83
One refined	0.73	1.00	0.95
All refined	0.83	0.95	1.00

Table 2. Correlation coefficients (cohort 2).

	Original	One Refined	All Refined
Original	1.00	0.81	0.83
One refined	0.81	1.00	0.96
All refined	0.83	0.96	1.00

4.1.1. The Ability to Surpass Unknown Competitors

This ability was measured in phase 5, shown as the *Original* rows. At this point, each student had no more information than a sparring strategy that they had to beat to take part in the competition. This required them to build their strategies to surpass their companions' unknown strategies. It is interesting to note that the two cohorts presented notably different features in their results: while cohort 1 (Figure 4) was largely divided into three levels (below 40%, between 40% and 50%, and above 60%) with around one third of students in each, students in cohort 2 (Figure 5) had more linear outcomes.

4.1.2. The Ability to Surpass Known Competitors

This ability was measured in phase (10.a), shown as *One refined* rows. In this case, cohort 1 presents a more positive picture than cohort 2 in general, as several students significantly improved their strategies as they refined them. Compared to the previous skill, the number of students in the average group increased; the top group remained much the same. Cohort 2 presents more varied changes, with two students sharply rising, one from 20% to almost 60% and another from 30% to nearly 50%. This way, more than half of the students were in the 40–60% range. This led to a more stratified ranking, with a poor group, an average one, and a top one.

4.1.3. The Ability to Improve in a Competitive Environment

This ability was measured in phase 10.b, shown in the *All refined* rows. The results of cohort 1 again show two students sharply improving, one passing from a poor result to 50% of the possible points, and another one improving from an average position to a top one with over 60% of the possible points. Interestingly, two students in the "top" group according to the first round ranked averagely with around 50% of the possible points. In cohort 2, the results show a more compact group in this skill (the first round defined a clear linear rank). This is so because three students from the lower half of the class improved significantly and six of the upper-half decreased.

4.2. Data Analysis

In both cohorts there were initially three groups of students according to their performances in ASUC: a poor group (students under 35%), an average group (between 35% and 50%), and a top group (students over 50%). However, the results obtained in the first competition round (original line) showed a different pattern.

The different patterns in the ASUC (cohort 1 is stratified; cohort 2 is more linear) show a quite different starting point. Students in the poor group (cohort 1) had room for improving, especially after performing a critical review of their own strategies. They acknowledged that their poor results were mainly due to poor overall strategy, rather than being caused by the good ideas of their rivals. Students in the average group tried to fix some issues in their own strategies while incorporating some ideas of some top strategies. Finally, students in the top group had to face the challenge of improving their successful strategies after they became publicly available for classmates. In this case, fine refinement of the strategies was the most sensible option.

Additionally, while students' performances in cohort 1 were stratified, values concentrated between 20% and 73%. This is a rather narrow range compared to that of cohort 2, which was wider at both ends: the lowest score was 10% and the highest one was 84%. These data reflect that students' ASUC in cohort 1 was more uniform than that of students

in cohort 2, and suggests that students in cohort 2 dared to make deeper changes in their strategies. The exception was for students with very high performance, which a priori had less room for improvement: they had to defend their positions once their strategies had been revealed.

In cohort 1, two students showed significantly better performance after refinement, both in the ASKC and in the ASIC: student S2 from the poor group and student S7 from the average one. Conversely, two students fell from the top group to the average one in the second round. These four students had very different performances across the three different skills, showing the advantage of the proposal for multiple skill assessment. Interestingly, another four students had worse performances after the code refinement: while students S8 and S10 kept rather similar average skill performances; students S12 and S15 (top students in the ASUC) performed averagely in the ASIC. This may indicate that some students failed to adequately test whether their changes really improved the performances or not. Their behavior is different from that of S13 and S14, who were also in the top group. They not only remained on top, but increased their performances, obtaining around 10% more of the achievable points (something especially difficult, as their room for improvement was greatly reduced). Finally, most students obtained better grades in ASKC compared to the ASUC, demonstrating being able to learn from experience.

In cohort 2, initial performance in the ASUC followed a linear pattern, but after code refinement the grades became more uniform: the students with average original strategies learned well from the best competitors, thereby improving their performances. In fact, in the ASKC half of the students performed similarly (average), and that average group even increased in the ASIC. Individually, the top four students in the ASUC (S13, S14, S15, and S16) also stayed at the top in the ASKC and ASIC by making small refinements, applying what they learned by reviewing their peers' code. The same applied to the low performance side, students S1 and S3: they retained poor performance for all three skills. However, we can observe that student S2 started by showing very poor performance in ASUC, but was in the top group in the ASKC and performed similarly in the ASIC. Conversely, student S8 worsened a bit in the ASKC and decreased more his performance badly in ASIC.

4.3. Students' Perspectives

In order to evaluate the competitive assessment methodology from the point of view of the participants, a final anonymous questionnaire was passed to all students. In this questionnaire, the students evaluated, using a 5-point Likert scale, the question: *Do you consider this use of competitions to assess skills to be fair?* Table 3 shows the summary of answers.

Table 3. Summary of answers to the question on the competitive assessment methodology.

Answer	Proportion of Students
(1) Strongly disagree	3%
(2) Disagree	0%
(3) Neither agree nor disagree	8%
(4) Agree	25%
(5) Completely agree	65%

The answers to the questionnaire show good reception from the students: 65% and 25% of the them, respectively, fully agreed or agreed with the fulfillment of the objectives by the methodology. Only one student disagreed.

In addition, the students could also make some comments, such as feedback on the experience. It should be noted that most of the comments were along the lines that they really liked the approach of the experience, which they found interesting, entertaining, and enjoyable. Only one student indicated that he would have liked to have more time to refine his code. Finally, it is worth highlighting the final comment by one of the students:

"This course was very useful to improve programming techniques. This is the best programming course I have ever followed. It helped me a lot to improve my programming skills."

4.4. Discussion

On the basis of these results and in line with the research question, the lecturer did consider that the multi-level software development learning method provided evidence of the students' performances in terms of competitive skills. Competition among students provides objective indicators of student performance and prepares students for positions in the workforce [34]. In this paper, we have presented a methodology for designing and evaluating competitive learning scenarios. This methodology has allowed us to obtain evidence to assess our students in three competitive skills.

Selvi and Çosan observed that students developed strategic skills by playing games that provided an environment of competition among themselves, which encouraged the students' decision-making processes [35]. However, the focus was not on measuring the ability to beat their peers' strategies but on understanding student motivation. In our study, each student's strategy aimed to outperform their peers' strategies, which they did not initially know. Therefore, based on the evidence obtained in phase 5 of the game, students designed their strategies and competed against their peers. The students knew what their goal was, but not how their peers would develop their strategies. Thanks to the evidence collected from the games, the lecturers could measure the students' performances in the ASUC.

What happens to students who lose? According to the research of Muñoz-Merino et al. [22], students who performed poorly in competition lost interest in the game. However, the results provided by Ventura et al. [36] show that players become even more engaged in the competition when they are losing. In line with the research of Burleson and Olimpo [37], the design of our methodology allows the student to guess how their peers think, i.e., how their peers will perform in the next phases of the game. It did not matter whether they were high or low performers in the previous phase. With phases 10.a and 10.b, we allow students to improve their strategies and participate in a new round of competition with their peers. In phase 10.a, we measure how they improve their strategies to face competitors whose strategies they may already know, i.e., to surpass known competitors. In phase 11, we measure how they improve their strategies to face competitors whose strategies may also have been improved, i.e., to surpass rivals in a competitive environment. Thus, we have shown how lecturers can use this evidence to measure students' performances in ASKC and ASIC.

Another interesting issue is the approach to refinement. While we have no conclusive evidence about this, the informal comments of the students showed that students mainly followed three different approaches: just fixing bugs, modifying the strategy, or adopting rules from their peers' strategies. Most likely, those who followed the first approach were sure that their strategy was a winning one, but detected some bugs and just fixed them. Others could consider that their strategy could significantly improve with minor modifications, and probably re-priorized rules, deleted undesired ones, or added new rules that they programmed after the first competition. Students in these two situations could have not actually peer-review the code of other students. We cannot blame them for this: if after the first competition they really considered that the other strategies could not provide them what they needed to succeed, we have to respect their opinions. In any case, the lecturer reminded them before the refinement phase that the programming language was chosen to simplify incorporating rules from other strategies (just copying and pasting rules), and that reviewing colleagues' code, even if they finally decided not to adopt any of their rules, could be a source of inspiration. Finally, some students who reviewed peers' strategies acknowledged that they focused on the strategies of colleagues who had brilliantly performed in previous courses about computer programming.

Concerning the approach to designing a strategy, most students admitted that, due to the competition being seen on the big screen, they worked harder than in similar assignments in other courses. The assignment stated that students had to individually program a strategy to play the game; they were not allowed to work in groups or get help from any other source (including students enrolled in previous editions of the course). Breaking this rule would be a violation of the institutional code of conduct. In any case, the game used in the experience was a simplified version of Stratego, a well known board game. It was selected as an easy to play but difficult to master game. This way, if students had previously played the game, or if they checked in the Internet, they would be aware of some required features of a winning strategy for that game. For further research, complementary experiences using other games could be interesting.

The number of participants is another key factor for the proposal: due to the nature of the refinement phase considering peer review, if the number of students is rather high, this process could take too long to be properly conducted. Conversely, if the number of participants is too low, the competition could be easily biased: one student beating all the others, or even worse, a student surpassed by the rest. The alternative for large groups could be dividing them into medium-sized ones, but this could raise issues around bias. This could be mitigated by playing a previous qualifying round before each competition (i.e., playing a short round robin, and then, based on the results, matching leading teams in a competition and those with poor results in another one). This, while possible, would complicate the implementation in the classroom significantly.

Finally, the addition of the random obstacle to the second cohort made the game a bit more difficult, but it was the same game for all participants. Additionally, the second cohort had better resources with which to learn expert systems programming: refined slides, a lecturer aware of the usual problems students had when learning the topic, etc. All in all, we consider that the difficulty ended up being similar for both cohorts.

4.5. Lessons Learned

After reflecting on the experience, we have compiled the following recommendations for transferability:

1. Keeping focus. We learned the importance of remarking all through the experience that in a one versus one competition, there must be a winner and a loser. However, it is the strategy and not the student who coded it that is a winner or loser. a student whose strategy did not perform well usually felt discouraged, and sometimes even blamed the game or the assessment method. After insisting on the differences between the strategy designed, the refinement, and the general programming skills of a person, they usually felt better.
2. Related to the previous reflection, we also insisted on the three skills (especially ASUC). Students appreciated when, after the first assessment, we explained that losing a competition was not a failure if they had tried their best. Confronting unknown competitors is hard, and they are not used to facing those situations in academia. In real life, competition is often faced in teams, and there is not always a clear winner and loser. We insisted that, probably, the key aspect of the learning experience for each of them was the refinement. It was the moment to show what they had learned from the first competition and to demonstrate their ability to learn from their own mistakes and the success of others.
3. Students really enjoyed watching their matches projected on a big screen during the competitions. They shouted as though they were watching a soccer match (the national sport in Spain). Due to the amount of matches, it would not be possible to watch even half of them, but we tried to watch as many as possible. This allowed seeing the strategies live and reflecting on their pros and cons. The supervisor insisted that all strategies had some interesting aspects, and they were discussed informally, providing feedback to the authors.

4. The game and the scoring frame have to be carefully selected according to the context. We used a simplified version of Stratego, a board game for two players, and it proved to be suitable to be programmed (and later modified) in the available time slots. However, there are other games for three, four, or more players (such as [38]) that can be programmed in other programming languages (as seen in [39]) and may be suitable for different environments given different time restrictions.

4.6. Threats to Validity

Although the results have been satisfactory, the validity of this study is subject to several threats—first, students' genders, as the majority of the students enrolled in this course were men, so in order to generalize these findings, it would be interesting to replicate the experiment with gender parity.

Secondly, the addition of the random obstacle to the game in the second cohort could be considered a way to improve the game and make it more similar to the original Stratego (adding a bit more depth to the game). While this led to making the game a little more difficult, it was the same game for all participants. Additionally, the second cohort benefited from better resources to learn expert systems programming: refined slides, a lecturer aware of usual problems of students when learning the topic, etc. Thus, we consider that the difficulty was very similar.

Finally, in regard to time limits, although the time restriction was partially solved using the strategies from previous years, a more complete analysis of the improvement in skill acquisition might be reached by a longer practice period. Unfortunately, the course was discontinued in the undergraduate program.

5. Conclusion

This paper presents a methodology with which to design and assess competitive learning scenarios consisting of serialized programming competitions using objective information. The strategies programmed by students during the first competitive round were publicly disseminated. Students developed skills while refining their solutions for a second round, by fixing bugs, fine-tuning their strategies, or applying what they learned by reviewing their peers' code. Such refined strategies also faced their previous versions, thereby providing metrics about their actual learning. Taking advantage of this approach, this work assessed the following skills:

1. The ability to surpass unknown competitors (ASUC)
2. The ability to surpass known competitors (ASKC)
3. The ability to surpass improved competitors (ASIC)

A case study based on the principles of design-based research has been shown. The case study was conducted in an elective computer science course, where students developed rule-based expert systems for a simple board game. Complementary sources of information were used: the observations from the supervisor and the students' results in the competitions and their comments.

The different sources of information support the validity of the proposal in context. Firstly, most students showed significantly different performances in the three skills, usually showing what they had learned from the first round. Their opinion about the experience was also very positive, aligned with the observations from the supervisor, providing evidence of the validity of the data produced to objectively assess student performance in the three targeted competitive skills.

We can compile the following conclusions: Firstly, the vast majority of students were faced for the first time in their degree with a competition. While the capitalist world is competitive by nature, the competition seems to be focused on the grades that each student independently gets. The competition was considered positive in general, so we can conclude competitions could be included more often in courses to improve students' skills for the real world. In this case, the proposal was successful, obliging all the students to take part in the competition.

Secondly, the measurement of the ASUC proved to be useful for demonstrating to students their actual ability to learn from mistakes. It helped to encourage students who had performed poorly in ASUC and/or ASIC. We have found no similar approach in the literature.

Thirdly, students acknowledged their limited experience in reviewing peers' code. Despite being a basic skill for actual collaboration in programming real-world projects, academia fails to properly develop it in degrees. Unfortunately, in this case not all the students did actually review code, so the success of the proposal in this aspect was limited.

Finally, positive evaluations were given by both the students and the supervisor. This enjoyment, while subject to limitations due to the limited sample of data and specific context (i.e., specific academic background of students, choice of game, scoring system, time restrictions, and programming language), should help academics successfully transfer the proposal to other contexts.

Two future lines work are planned. The first line of work will be validating the proposal in other settings: changing the game, the programming language, and other aspects. The second line of work will be providing additional contextual information to the expert systems. For example, if in the recent matches of a league a strategy has a rather high performance, perhaps its rival could try to tie a match and incur no risk in trying to win it.

Author Contributions: The courses through which the methodology for designing and assessing competitive learning scenarios were applied were run by M.P.-D. The research was conducted by A.G.-D. and A.B., and supervised by M.P.-D. All authors have approved the manuscript for submission.

Funding: This research was funded by Spanish National Research Agency (AEI), through the project VISAIGLE (TIN2017-85797-R) with ERDF funds.

Institutional Review Board Statement: Ethical review and approval were waived for this study, due to not involving personally identifiable nor sensitive data.

Informed Consent Statement: Student consent was waived due to not involving personally identifiable nor sensitive data.

Data Availability Statement: Data supporting the reported results can be found in https://doi.org/10.5281/zenodo.5036553 (accessed on 28 June 2021).

Acknowledgments: The authors would like to thank the students who contributed to the free/libre software used in this paper.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

DBR	Design-Based Research
ASUC	Ability to Surpass Unknown Competitors
ASKC	Ability to Surpass Known Competitors
ASIC	Ability to Surpass Improved Competitors

References

1. De Los Rios, I.; Cazorla, A.; Díaz-Puente, J.M.; Yagüe, J.L. Project–based learning in engineering higher education: Two decades of teaching competences in real environments. *Procedia-Soc. Behav. Sci.* **2010**, *2*, 1368–1378. [CrossRef]
2. Blömeke, S.; Zlatkin-Troitschanskaia, O.; Kuhn, C.; Fege, J. Modeling and measuring competencies in higher education. In *Modeling and Measuring Competencies in Higher Education*; Springer: Rotterdam, The Netherlands, 2013; pp. 1–10.
3. Burguillo, J.C. Using game theory and Competition-based Learning to stimulate student motivation and performance. *Comput. Educ.* **2010**, *55*, 566–575. [CrossRef]
4. Association, P. Universities Catch Almost 50,000 Student Cheats. Available online: https://www.theguardian.com/education/education+students?page=149 (accessed on 13 October 2018).

5. Arevalillo-Herráez, M.; Benavent, X.; Ferris, R. A competitive learning strategy in teaching programming to students of mathematics. In Proceedings of the 3rd International Technology, Education and Development Conference (INTED); International Association of Technology, Education and Development (IATED), Valencia, Spain, 9–11 March 2009; pp. 1768–1772.
6. Thielscher, M. General game playing in AI research and education. In Proceedings of the 34th Annual German conference on Advances in artificial intelligence, Berlin, Heidelberg, 4–7 October 2011; pp. 26–37.
7. Farr, J.V.; Brazil, D.M. Leadership skills development for engineers. *Environ. Eng. Manage. J.* **2009**, *21*, 3–8. [CrossRef]
8. Schoemaker, P.J.; Krupp, S.; Howland, S. Strategic leadership: The essential skills. *Harv. Bus. Rev.* **2013**, *91*, 131–134.
9. Siewiorek, A.; Saarinen, E.; Lainema, T.; Lehtinen, E. Learning leadership skills in a simulated business environment. *Comput. Educ.* **2012**, *58*, 121–135. [CrossRef]
10. Griffiths, R.P.; Eastin, M.S.; Cicchirillo, V. Competitive Video Game Play An Investigation of Identification and Competition. *Commun. Res.* **2016**, *43*, 468–486. [CrossRef]
11. Murphy, S. Video Games, Competition and Exercise: A New Opportunity for Sport Psychologists? *Sport Psychol.* **2009**, *23*, 487. [CrossRef]
12. Palomo-Duarte, M.; Dodero, J.M.; Tocino, J.T.; García-Domínguez, A.; Balderas, A. Competitive evaluation in a video game development course. In Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education, Haifa, Israel, 3–5 July 2012; pp. 321–326. [CrossRef]
13. Anderson, T.; Shattuck, J. Design-Based Research: A Decade of Progress in Education Research? *Educ. Res.* **2012**, *41*, 16–25. [CrossRef]
14. Nichols, M.H.; Cator, K. *Challenge Based Learning White Paper*; Technical report for Apple Inc.: Cupertino, CA, USA, December 2008.
15. Conde, M.Á.; Sedano, F.J.R.; Fernández-Llamas, C.; Gonçalves, J.; Lima, J.; García-Peñalvo, F.J. RoboSTEAM Project Systematic Mapping: Challenge Based Learning and Robotics. In Proceedings of the 2020 IEEE Global Engineering Education Conference (EDUCON), Online, 27–30 April 2020; pp. 214–221. [CrossRef]
16. Gallagher, S.E.; Savage, T. Challenge-based learning in higher education: An exploratory literature review. *Teach. High. Educ.* **2020**, 1–23. [CrossRef]
17. Johnson, L.; Brown, S. *Challenge Based Learning: The Report from the Implementation Project*; Technical report for The New Media Consortium; The New Media Consortium: Austin, TX, USA, December 2011.
18. Willis, S.; Byrd, G.; Johnson, B.D. Challenge-Based Learning. *Computer* **2017**, *50*, 13–16. [CrossRef]
19. Slavin, R.E. Cooperative Learning. *Rev. Educ. Res.* **1980**, *50*, 315–342. [CrossRef]
20. Regueras, L.; Verdú, E.; Munoz, M.; Perez, M.; de Castro, J.; Verdú, M. Effects of Competitive E-Learning Tools on Higher Education Students: A Case Study. *IEEE Trans. Educ.* **2009**, *52*, 279–285. [CrossRef]
21. Gomes, A.; Mendes, A. A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations. In Proceedings of the 2014 IEEE Frontiers in Education Conference (FIE), Madrid, Spain, 22–25 October 2014; pp. 1–8.
22. Muñoz-Merino, P.J.; Molina, M.F.; Muñoz-Organero, M.; Kloos, C.D. Motivation and emotions in competition systems for education: An empirical study. *IEEE Trans. Educ.* **2014**, *57*, 182–187. [CrossRef]
23. Muñoz-Merino, P.J.; Molina, M.F.; Munoz-Organero, M.; Kloos, C.D. An adaptive and innovative question-driven competition-based intelligent tutoring system for learning. *Expert Syst. Appl.* **2012**, *39*, 6932–6948. [CrossRef]
24. Palomo-Duarte, M.; Dodero, J.M.; García-Domínguez, A. Betting system for formative code review in educational competitions. *Expert Syst. Appl.* **2014**, *41*, 2222–2230. [CrossRef]
25. Carpio Cañada, J.; Mateo Sanguino, T.; Merelo Guervós, J.; Rivas Santos, V. Open classroom: Enhancing student achievement on artificial intelligence through an international online competition. *J. Comput. Assist. Learn.* **2015**, *31*, 14–31. [CrossRef]
26. Grotz, N. Improving programming skills using computer based feedback and peer group competition. In Proceedings of the 2016 IEEE Global Engineering Education Conference (EDUCON), Abu Dhabi, United Arab Emirates, 10–13 April 2016; pp. 585–591.
27. Arevalillo-Herráez, M.; Claver, J. Combining Cooperative and Competitive Learning to Facilitate Knowledge Transfer Between Peers. In Proceedings of the 2008 International Conference of Education, Research and Innovation (ICERI), Madrid, Spain, 17–19 November 2008.
28. Paneda, X.; Melendi, D.; Cabrero, S.; Blanco, R.; Garcia, R.; Rionda, A. Three techniques for competitive lab activities based on project-oriented learning in ICT. *Learn. Technol. IEEE J. Lat. Am.* **2013**, *8*, 39–46. [CrossRef]
29. Wang, Y.; Li, H.; Feng, Y.; Jiang, Y.; Liu, Y. Assessment of programming language learning based on peer code review model: Implementation and experience report. *Comput. Educ.* **2012**, *59*, 412–422. [CrossRef]
30. Hevner, A.R.; March, S.T.; Park, J.; Ram, S. Design Science in Information Systems Research. *MIS Q.* **2004**, *28*, 75–105. [CrossRef]
31. Chiang, R.; Siau, K.; Hardgrave, B. Advances in management information systems. In *Systems Analysis and Design: Techniques, Methodologies, Approaches, and Architectures;* M.E. Sharpe: Armonk, NY, USA, 2009.
32. Giarratano, J.C.; Riley, G. *Expert Systems*, 3rd ed.; PWS Publishing Co.: Boston, MA, USA, 1998.
33. Liao, S.H. Expert system methodologies and applications—A decade review from 1995 to 2004. *Expert Syst. Appl.* **2005**, *28*, 93–103. [CrossRef]
34. Cretsinger, M.A. Academic competitiveness among graduate students. Master's Thesis, University of Wisconsin-Stout, WI, USA, July 2005.

35. Selvi, M.; Çosan, A.Ö. The Effect of Using Educational Games in Teaching Kingdoms of Living Things. *Univers. J. Educ. Res.* **2018**, *6*, 2019–2028. [CrossRef]
36. Ventura, R.B.; Richmond, S.; Nadini, M.; Nakayama, S.; Porfiri, M. Does Winning or Losing Change Players' Engagement in Competitive Games? Experiments in Virtual Reality. *IEEE Trans. Games* **2019**, *13*, 23–34. [CrossRef]
37. Burleson, K.M.; Olimpo, J.T. ClueConnect: A word array game to promote student comprehension of key terminology in an introductory anatomy and physiology course. *Adv. Physiol. Educ.* **2016**, *40*, 223–228. [CrossRef] [PubMed]
38. Hensman, A. Evaluation of Robocode as a Teaching Tool for Computer Programming. In Proceedings of the 2007 National Digital Learning Repository (NDLR) Symposium, Trinity College Dublin, Ireland, 14 December 2007.
39. Gorbachev, R.; Semendyaev, S.; Khokhlov, I.; Litvinenko, V.; Ryakin, I. The Robosoccer as a Modern Educational Platform in the Field of Artificial Intelligence. In Proceedings of the 2019 International Conference on Artificial Intelligence: Applications and Innovations (IC-AIAI), Neos Marmaras, Greece, 5–7 June 2019; pp. 59–61. [CrossRef]